IBM® General Information Manual

1401 Data Processing System

MINOR REVISION (February, 1960)

This edition D24-1401-1 is a minor revision of the preceding edition but does not obsolete D24-1401-0. The principal changes in this edition are:

PAGE	SUBJECT
20	Input-Output Storage Assignments
23, 24	Input-Output Branch Codes
28	B (I) Unconditional Branch, Figure 24
30	Z (A) (B) Move and Zero Suppress
32	• (I) Stop and Branch
35	Figure 30 Cycles 1 through 8
41	Auxiliary Console
42	Figure 37
43	Printer Controls
51	A and B Auxiliary Registers
55	Figures 54 and 55 Multiply Subroutine
57	Figure 57 Divide-Subroutine
58, 59	Character Code Chart
60	1401 Timing Formulas
62	Magnetic Tape Timings

Contents

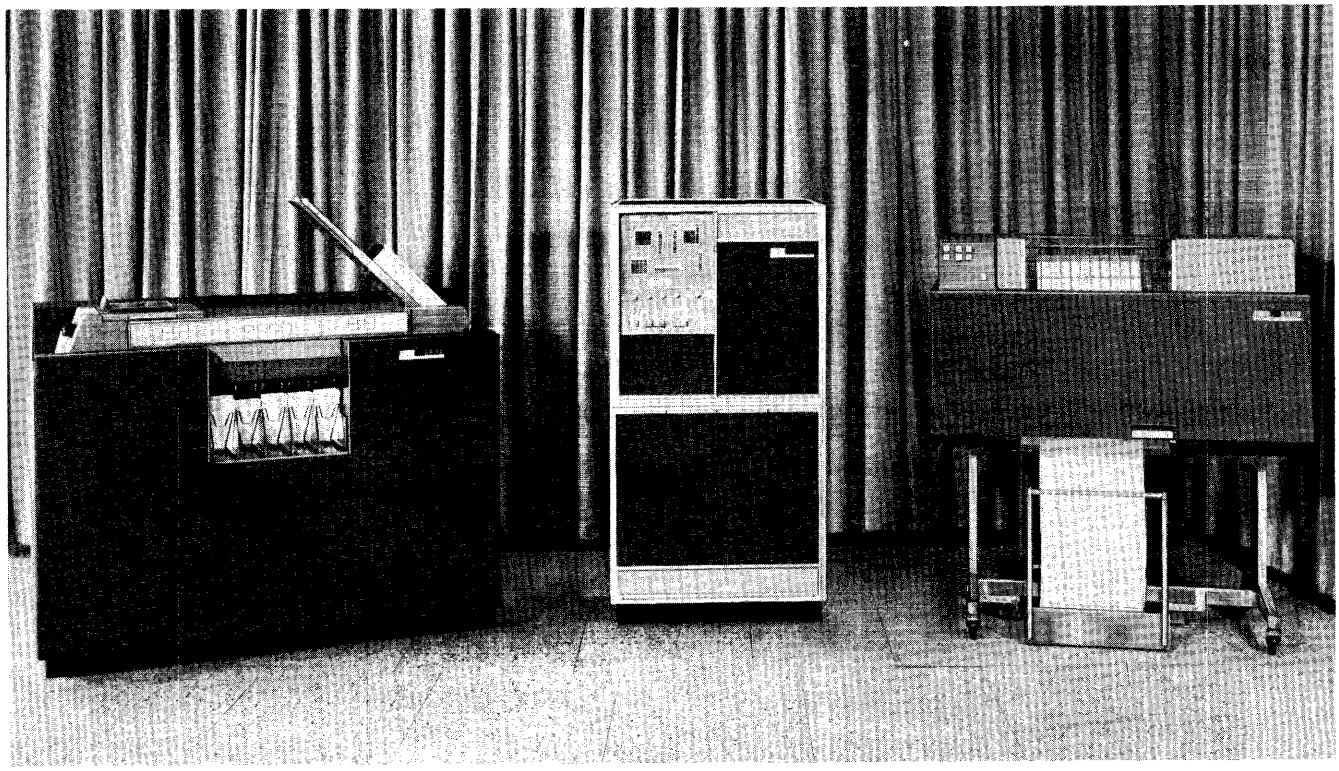


FIGURE 1. IBM 1401 DATA PROCESSING SYSTEM

IBM 1401 Data Processing System

Significant characteristics in the growth of data processing are the transitions from manual to mechanical, and from mechanical to electronic methods.

Tremendous advantages accrue to business when manual or semi-manual procedures are converted to mechanical procedures. The curve of efficiency rises sharply while the curve of cost declines in such a changeover. Furthermore, important gains are realized in quantity and accessibility of significant data when unit-record equipment and the punched card replace manual and semi-automatic methods.

As the volume of data to be processed increases, as the decision-making process is refined to the point where it requires more and more information, as the time available for decision-making becomes shorter, unit-record equipment continues to offer economies and advantages, but not at the same *rate* of improvement in time-saving and dollar-saving.

The next available step in the mechanization process takes the businessman into intermediate and large-scale data processing. Each time this step from unit-record data processing to data processing systems is taken, it must be preceded by a carefully planned program.

The IBM 1401 Data Processing System was specifically designed and planned in various configurations to make this transition. Punched-card input is at much higher rates of speed than those used in unit-record equipment, and yet the processes in punched-card input are very similar to the unit-record equipment. The same applies to punched-card and printed output. The speeds are greater, the concepts are similar.

The processing unit of the IBM 1401 is built with a degree of accessibility and utility that makes it the equivalent of storage units of greater capacity.

The IBM 1401 is further enhanced by configurations that include tape systems, and the advantages of magnetic-tape data-handling.

The Philosophy of IBM 1401

The IBM 1401 can be considered in two major configurations: the card systems, and the tape systems.

Card systems configurations are planned for procedures involving large volumes of card documents as source data and output, with particular advantage to applications requiring re-entry data.

Tape system configurations provide for the handling of magnetic tape, which has the advantages of compact record handling and storage medium for high-speed data processing systems.

The Stored Program Concept

The philosophy of data processing systems is self-controlled performance of procedures, carried to various degrees. Any such self-controlled performance involves simply a series of actions or movements, each depending on another, and requiring no operator intervention in the completion of the series. The series may be very short, or very long. The series may be completely sequential, or the next action to be taken may be chosen by the last action completed.

An automatic record player is a good example of a series of actions, each one depending on the one immediately preceding it. When records are loaded on the spindle and the record player turned on, the record plays, the arm returns to a neutral position, the next record in sequence drops into place, the playing arm returns to the starting position on the new record, the record plays, and so on until all the records have been played once, without any need for intervention or assistance by anyone. This series of actions is called a *program* (Figure 2).

In data processing systems, the program is more complex. It controls the entire flow of data in and out of various processing units. If, for instance, original data is punched into cards, the program would control the reading of this data, its transport to various proc-

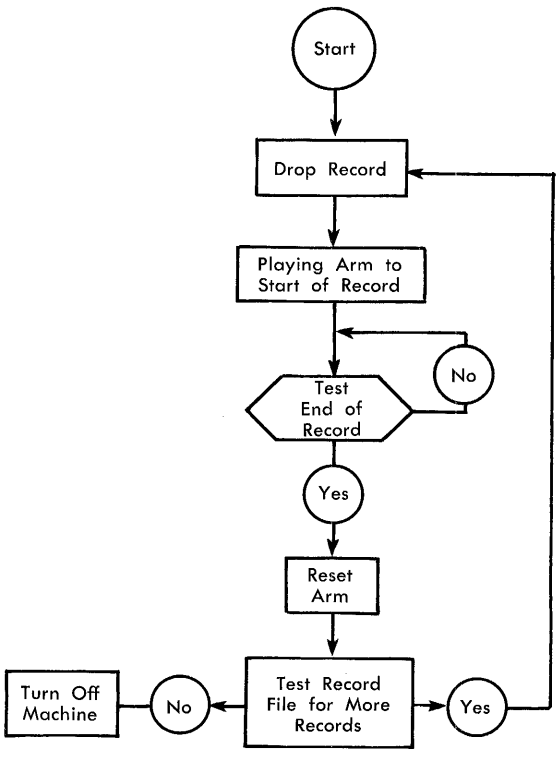


FIGURE 2. SCHEMATIC OF A PROGRAM

essing areas for addition, subtraction, multiplication, division, modification, classification, recording, and any other kind of action to which data can be subjected.

A data processing system is a group of various mechanical and electronic components, interconnected. A system of this kind must be able to handle and complete such a program. The concept of *stored programming* provides this flexibility and efficiency.

In punched-card data processing, the wires in the control panel actually comprise the program of instructions. The requirements of the procedure are studied carefully, and then the proper wires are placed in the control panel. The entire program can be changed by removing one control panel and replacing it with another for a different procedure. The limiting factors in the extent of the program that unit-record equipment can handle is the number of program steps that can be provided within the physical confines of the control panel, and the number of control panels that can be conveniently utilized.

Stored-program data processing systems use a similar, but much more flexible, concept. All the instructions needed to complete a procedure are written in the form of program steps. These program steps are made available to the machine by various methods, the most common of which is punched cards. The data processing system stores these program steps in some kind of storage medium.

Thus, when a procedure is to begin, the stored program is *loaded* into the system (Figure 3), and the entire procedure can be performed from beginning to end. The IBM 1401 Data Processing Systems make use of three kinds of storage: magnetic-core storage, magnetic-tape storage, and the already familiar punched-card storage.

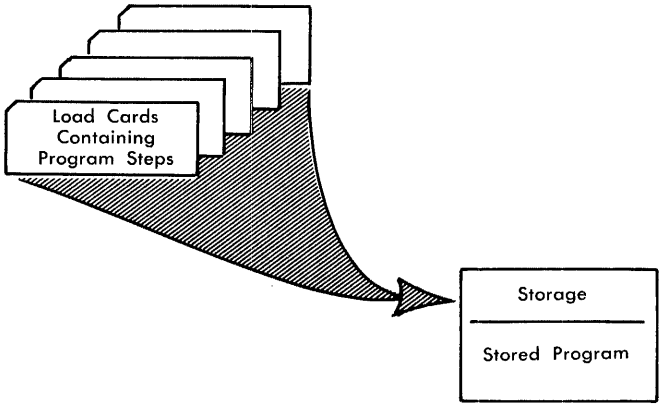


FIGURE 3. STORED PROGRAM

Magnetic-Core Storage

All configurations of the 1401 Data Processing System use magnetic-core storage for storing instructions and data.

The magnetic-core storage unit is composed of a number of tiny rings made of magnetic material. Several electric wires are passed through each of these rings, and each ring is magnetized.

Every magnetic field has *polarity*. This can be demonstrated by the common phenomenon of two horseshoe-shaped magnets, which attract each other firmly when turned one way, and repel each other just as firmly when turned the other way. Similarly magnetic cores possess a magnetic field, and its polarity can be reversed by passing a current through the wires.

These phenomena—magnetism, reverse magnetism, and the change from one to the other (Figure 4)—are used by the magnetic-core storage units to store information.

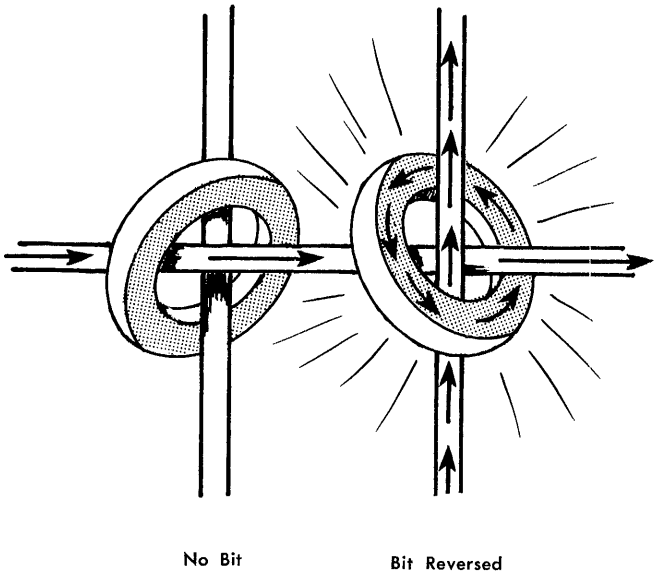


FIGURE 4. MAGNETIC CORE

A core magnetized in one direction contains a *bit* of information which has a value of 1; when the polarity is reversed, the value of the bit is zero. (This condition is referred to as *no-bit.)* Furthermore, all data in core storage is instantly available, and in the IBM 1401, the core-storage units have been specifically designed for high utility by making each location of core storage *addressable.* This means that a program step can designate the exact cores needed for that step.

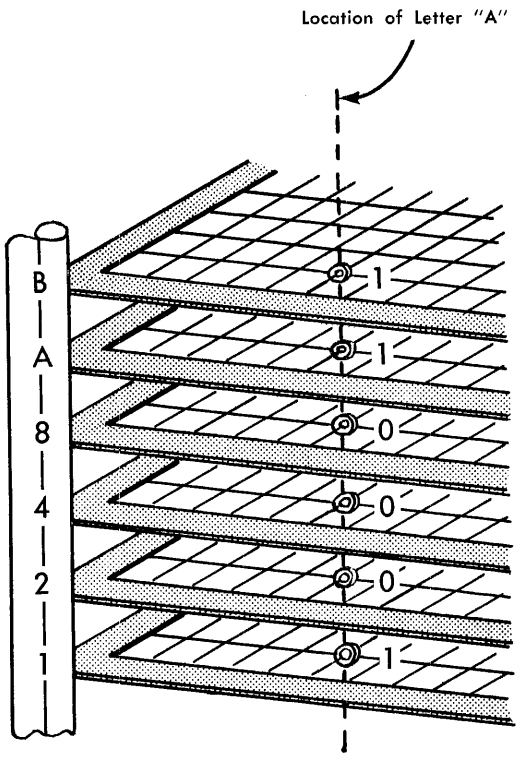


FIGURE 5. REPRESENTATION OF LETTER "A" IN CORE STORAGE

Each location of core storage consists of a number of *planes* or levels of magnetic cores. Various combinations of bits designate digits, letters, and special characters (Figure 5).

Notice that the planes are stacked, and the cores representing a single character (in this case the letter "A") are all at the intersection of the same two wires in each plane.

The physical makeup of each core storage location and its associated circuitry makes it possible for the IBM 1401 to modify instructions and process data directly in the storage area. (This is called *add-to-storage* logic.)

The design, construction, and circuitry of the core-storage unit in the IBM 1401 make it possible for this compact but extremely powerful storage unit to do as much or more than storage units of greater size.

Magnetic-Tape Storage

Magnetic tapes are made of plastic material, coated with a metallic oxide. It has the property of being easily magnetized in tiny spots, so that patterns of these magnetized spots are codes for digits, alphabetic characters, and special characters.

Data can be read from a variety of sources, and put on the tape. The magnetic spots representing the information that has now been stored on the tape remain until they are changed by positive action.

This means that, in addition to being used as data storage, this data itself can be part of input and output.

This makes magnetic tape an ideal storage medium for a large volume of data, because there is no limit to the amount of information that can be kept permanently. The reels of tape are removable from the system, and can be filed (Figure 6). They can also be transported from place to place, and used in other systems.

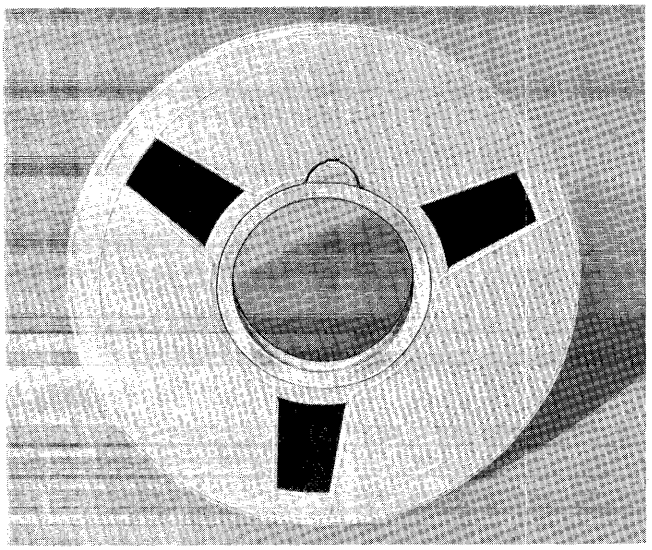


FIGURE 6. REEL OF MAGNETIC TAPE

Data stored on the magnetic tape is read sequentially. The data processing system can search the tape to find the data to be used. Program steps can be stored on magnetic tapes, and this method of storing is a common one for collecting a *library* file of procedures.

Another great advantage of magnetic-tape storage is that a reel of tape that has been produced as an output of a procedure can be removed from the data processing system, and reports written with an independent unit, while the data processing system proceeds with the next program to be performed.

Language

In the punched-card area of data processing, the language of the machine is the holes in the card. As data processing needs increase, the basic card language remains the hole in the card. But in the transition from

unit record systems to the 1401 Data Processing Systems, and from there into computer systems, another faster, more flexible machine language emerges.

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

Obviously, many different code patterns can be set up. The internal code used in the IBM 1401 Data Processing System is called *binary-coded-decimal.* 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), the binary-coded-decimal code is used in all data flow and processing from that point on, until it is translated into printed output as reports and documents are written, or converted to punched-card code for punched-card output. Converting input data to the 1401 internal code, and subsequently reconverting, is completely automatic.

Processing

The manipulation that data undergo in order to achieve desired results is called *processing,* and the part of the 1401 systems that houses these operations is called *processing unit.*

Processing can be divided into three general categories: logic, arithmetic, and editing.

Logic

The logic function of any kind of data processing system is comprised of its ability to execute program steps; but even more, its 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 or No X, 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 the 1401 system 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 all variations.

Arithmetic

The 1401 processing unit has the capacity to perform add, subtract, multiply, and divide operations. Multiplication and division can be accomplished in any 1401 system, by programmed sub-routines. When the extent of the calculations might otherwise limit the operation, a direct multiply-divide feature is available.

Editing

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

Checking

Advanced circuit design with extremely reliable components is built into the 1401 system to provide assurance of accurate results. Self-checking within the machine is separated into three categories: *parity, validity,* and *hole count.*

PARITY CHECKING. Achieved by testing for the proper number of 1-bits for any given character, known as *parity* for that character.

VALIDITY CHECKING. Checking for the correct configuration of bits to represent each character in storage.

HOLE COUNT. Counting the number of holes punched in a card, to establish that it is equal to the number of holes called for in the same card at a previous station.

Solid State Circuitry

Transistorization of 1401 components is another significant design characteristic. In addition to providing a lower cost system, use of transistors increases reliability, while decreasing maintenance requirements. Other advantages are carefully controlled:

space requirements
heat dissipation
power requirements

The physical arrangement of the system components offers a less tangible, but equally important benefit, in greater operating efficiency, in that the components requiring operator attention can be situated for accessibility and convenience. The controls and arithmetic components are consolidated into a single set of modular cabinets.

Thus far, only the most obvious advantages offered by the 1401 have been given. As the system components and features are described in greater detail, further advantages become evident. The power and econ-

omy of the 1401 is not derived from any single characteristic or component, but from the many considerations that led to the design of a balanced system in which every component can operate at its optimum rate.

Advanced Design

Advanced systems design of the IBM 1401 permits use of the machine as a complete, independent, accounting system. It can also perform low-cost, direct input and output, and auxiliary tape-operations for large scale data processing systems.

The entire system is operated by the stored program. Time saving features, such as the powerful editing function, and the elimination of control panels, provide increased flexibility for application development. The capacity to use magnetic-tape data means economy in recording, transporting, and storing large volumes of information in compact form.

The IBM 1401 Card Systems are completely transistorized, and utilize the modern technique of stored-program control.

This system can perform all basic functions (such as: read-a-card, print-a-line, compare, add, subtract, edit), and variations of these functions.

The IBM 1401 incorporates an advanced design of many outstanding features of existing equipment, for improved programming and operating efficiency.

CORE STORAGE. Instant access to information, and application of stored programming. Every position is alphanumerical, and individually addressable.

VARIABLE WORD-LENGTH. Permits a maximum utilization of the storage facility.

HIGH SPEED PRINTING. A medium of output efficiency.

HIGH SPEED READING AND PUNCHING. Simplified input-output facilities and easy integration of the 1401 into existing accounting machine procedures.

EDITING. Completeness in preparing output information for printing, or with magnetic tape operations.

Component	*Card System*		*Magnetic Tape System*	
IBM 1401 Processing Unit	Model A1-A2-A3	Model B1-B2-B3	Model C1-C2-C3	Model D1-D2-D3
IBM 1402 Card Read Punch	Model 1	Model 1	Model 1	Not Available
IBM 1403 Printer	Model 1 or 2	Model 1 or 2	Model 2	Model 2
IBM 729 Magnetic Tape Unit	Not Available	Not Available	Model II or IV	Model II or IV
IBM 1401 PROCESSING UNIT SPECIAL FEATURES				
*Expanded Print Edit	Optional	Optional	Standard	Standard
*Read Punch Release	Optional	Optional	Standard	Not Available
*Sense Switches	Optional	Optional	Standard	Standard
*Additional Print Control	Optional	Optional	Standard	Standard
Multiply-Divide	Not Available	Optional	Optional	Not Available
Print Storage	Not Available	Optional	Optional	Optional
Column Binary	Not Available	Optional	Optional	Not Available
High-Low-Equal Compare	Not Available	Optional	Optional	Optional
*Can be field installed				

NOTES:

1403 Model 1 has 100 print positions.
1403 Model 2 has 132 print positions, also requires the Additional Print Control optional feature.
1401 Processing Unit Models A1-B1-C1-D1 have 1400 storage positions.
1401 Processing Unit Models A2-B2-C2-D2 have 2000 storage positions.
1401 Processing Unit Models A3-B3-C3-D3 have 4000 storage positions.
Dual Speed Carriage is a standard feature for the 1401 DPS, Models B, C, and D. It is not available on Model A.

All Model A's provide a low cost card system—only certain special features (as indicated above) may be installed.
All Model B's provide the expanded card version—all special features except magnetic tape may be included.
All Model C's provide the full magnetic tape system with certain features standard and others optional.
All Model D's provide an edit system without card I/O—it should be noted that some options available on Model C are not available on Model D.

FIGURE 7. IBM 1401 DATA PROCESSING SYSTEM COMPONENTS

Physical Features

The physical features of the units that make up the card system are compact and of modern design. All units are mobile for convenient and efficient arrangement for operating.

The processing unit is the only unit that is changed in physical size when the different systems configurations (Figure 7) are required.

The 1401 Data Processing System in its card configurations is composed of three interrelated units:

1. IBM 1401 Processing unit, containing 1400 characters of alphanumerical core storage (expandable to 2000 or 4000 positions)
2. IBM 1402 Card Read-Punch, equipped with an 800-card-per-minute read feed and a 250-card-per-minute punch feed
3. IBM 1403 Printer, capable of printing up to 600 lines per minute, with a print span of 100 positions of alphanumerical data per line (expandable to 132 print positions).

IBM 1401 Processing Unit

The processing unit (Figure 8) contains the magnetic-core storage unit to perform all the machine logic.

The storage capacity is 1400, 2000, or 4000 alphanumerical characters of 8-bit core storage. The eight bits consist of six bits for alphanumerical binary code, a redundant bit for checking, and an eighth bit for field definition.

Three areas of storage are reserved for input and output data. In the first, 80 storage positions receive 80 columns of card information from the card reader. Another 80 positions are reserved for assembly of data to be punched. The third area is reserved for the assembly of 100 (or 132) characters of printer information. However, when these areas are not being used as specified, they can be used for other purposes. (NOTE: If 132-character printing is ordered, the *Additional Print Control* feature is required in the 1401.)

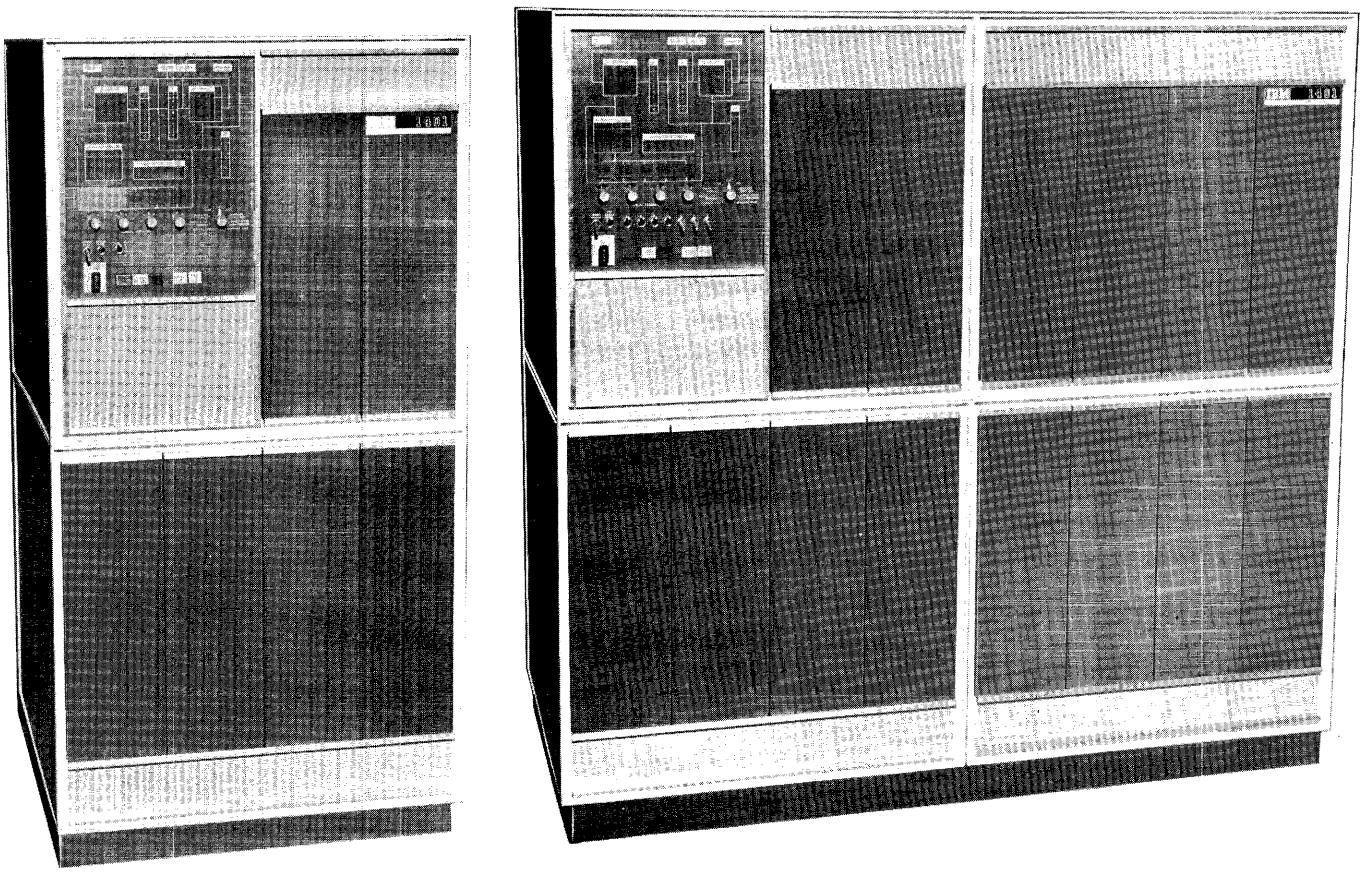


FIGURE 8. IBM 1401 PROCESSING UNIT (2 CUBE & 4 CUBE)

CODED ADDRESSES IN STORAGE

Actual Addresses		*3-Character Addresses*
000 to 999	No zone bits	000 to 999
1000 to 1099	A-bit, using 0-zone	+00 to +99
1100 to 1199		/00 to /99
1200 to 1299		S00 to S99
1300 to 1399		T00 to T99
1400 to 1499		U00 to U99
1500 to 1599		V00 to V99
1600 to 1699		W00 to W99
1700 to 1799		X00 to X99
1800 to 1899		Y00 to Y99
1900 to 1999		Z00 to Z99
2000 to 2099	B-bit, using 11-zone	$\bar{0}$00 to $\bar{0}$99
2100 to 2199		J00 to J99
2200 to 2299		K00 to K99
2300 to 2399		L00 to L99
2400 to 2499		M00 to M99
2500 to 2599		N00 to N99
2600 to 2699		*O00 to O99
2700 to 2799		P00 to P99
2800 to 2899		Q00 to Q99
2900 to 2999		R00 to R99
3000 to 3099	A-B-bit, using 12-zone	$\overset{+}{0}$00 to $\overset{+}{0}$99
3100 to 3199		A00 to A99
3200 to 3299		B00 to B99
3300 to 3399		C00 to C99
3400 to 3499		D00 to D99
3500 to 3599		E00 to E99
3600 to 3699		F00 to F99
3700 to 3799		G00 to G99
3800 to 3899		H00 to H99
3900 to 3999		I00 to I99

* Letter O followed by Zero Zero

FIGURE 9. STORAGE ADDRESS CODES

Each of the storage positions is identified by a 3-character address. The first 1000 positions of storage have the addresses 000-999. The remaining 3000 storage positions require the use of an alphabetic or special character in the hundreds position of the address, as in Figure 9.

The 1401 Processing Unit stores the program instructions and the data. It employs a *variable word-length concept,* and each position is addressable.

Stored programming involves the concept of *words.* A word is a single character, or group of characters, that represents a complete unit of information. One of the most important characteristics of the IBM 1401 Data Processing System is this variable word-length principle, in which words are not limited to any predetermined number of character positions in the storage unit.

Each word occupies only that number of character positions actually needed for each specific instruction, or for the specific data involved. This facility contributes to the high efficiency of the 1401 core-storage unit.

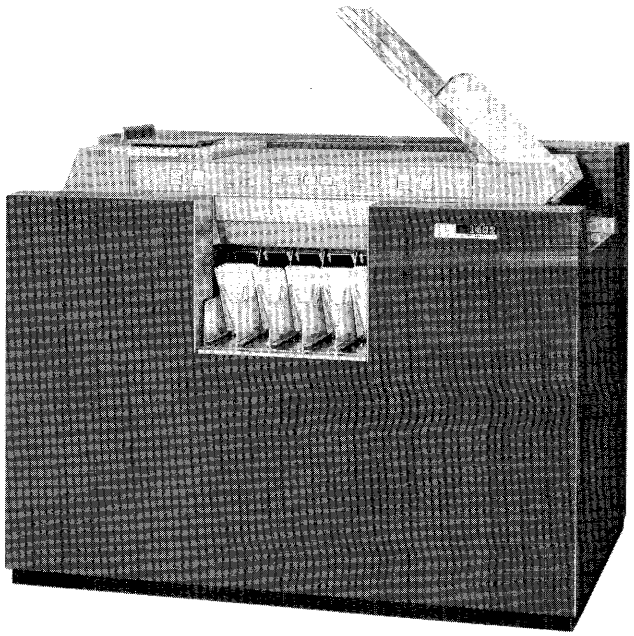


FIGURE 10. IBM 1402 CARD READ-PUNCH

IBM 1402 Card-Read Punch

The IBM 1402 Card Read-Punch (Figure 10) provides the card system with simultaneous punched-card input and output. This unit has two card feeds. The read section has a rated reading speed of 800 cards per minute. Actual card speed realized is governed by the program routine for each particular run. The read feed is equipped with a device for large capacity loading, called a *file feed.* With the file feed device, the read feed can be loaded with as many as 3000 cards, which reduces operator-attendance requirements.

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 *first reading* station reads 80 columns of the card to establish a hole-count for checking purposes. The *second reading* station also reads the 80 columns, proves the hole count,

and directs the data into 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 read hopper and is used unless the cards are program-directed to stackers 1 or 2.

The punch section has a rated speed of 250 cards per minute. The card hopper capacity is 1200 cards. The cards feed 12-edge first, face down. The feed path is 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 station counts all the holes in all 80 columns of the card, for punch-checking. 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 stacker 4 or 8 (Figure 11).

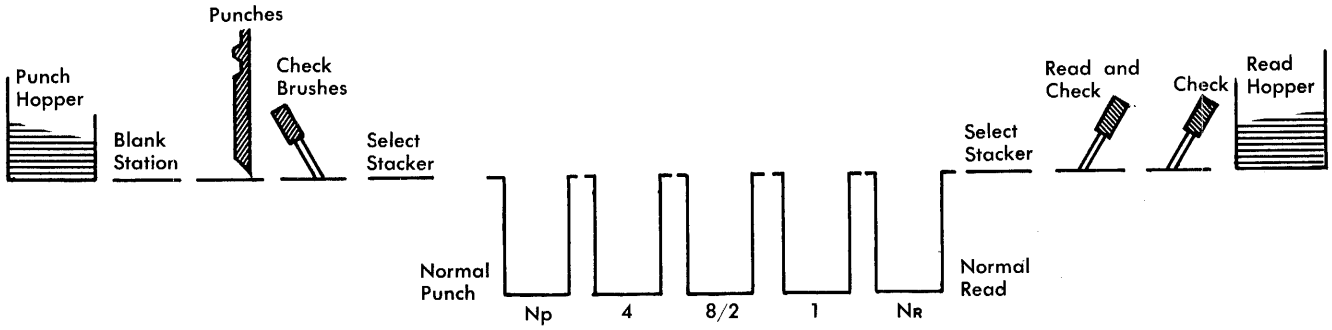


FIGURE 11. IBM 1402 CARD TRANSPORT SCHEMATIC

All these stackers are radial-type stackers (Figure 12) with a capacity of 1000 cards each. Cards can be removed from the stackers without stopping the machine. Two stackers are assigned exclusively to the reader and two are assigned exclusively to the punch. The center or *common* stacker (8/2 stacker) can be used by either unit, but it must be assigned by the program to one or the other, in any one run. All stackers other than the normal stackers are selected only under program control.

Both feeds are equipped with jam detection devices and with misfeeding detection. A card jam or a misfeed in either the read or punch feed causes the 1401 DPS to stop, and a console light glows, indicating which feed caused the stop.

There is no electrical or mechanical coupling between the read and punch units. Therefore, any information from the read side must be entered into storage and read out of storage to the punch unit, for operations equivalent to reproducing or gang punching.

IBM 1403 Printer

The printer (Figure 13) is another output medium for the 1401 DPS Card System. This unit has a rated printing speed of 600 lines per minute. The standard printing capacity is 100 positions, with an additional 32 positions optional.

Horizontal spacing is 10 characters to the inch. Vertical spacing of six or eight lines to the inch can be manually selected by the operator. In the 1401 Card System, Model A, vertical line spacing is performed by single-speed, tape-controlled carriage directed from the 1401 stored program. In Models B, C, and D a dual-speed tape-controlled carriage is standard. This carriage skips at the rate of 75 inches per second after the first eight lines of any skip. The single-speed carriage has a constant speed of 33 inches per second while skipping.

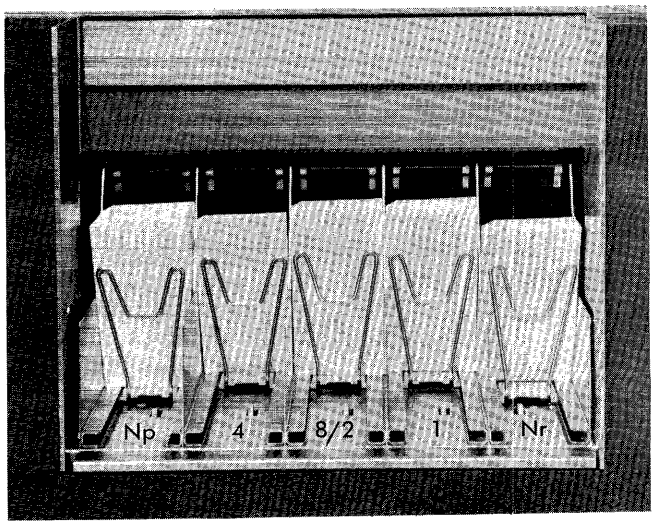


FIGURE 12. RADIAL STACKERS

Each position can print 48 different characters: 26 alphabetic; 10 numerical; and 12 special characters, (& , . ¤ − $ * + / % # @). In tape systems, the "+" character is replaced by a record-mark character (‡). The printing format is controlled by the 1401 DPS stored program. The information to be printed is checked when it is read out to the printer.

Print Storage (Optional)

This optional feature provides 100 or 132 non-addressable extra positions of core storage. They are used in conjunction with printer output. These extra positions of core storage increase processing speed in applications where printing volume is high.

The data to be printed is moved by the PRINT instruction from the area in core storage assigned to the

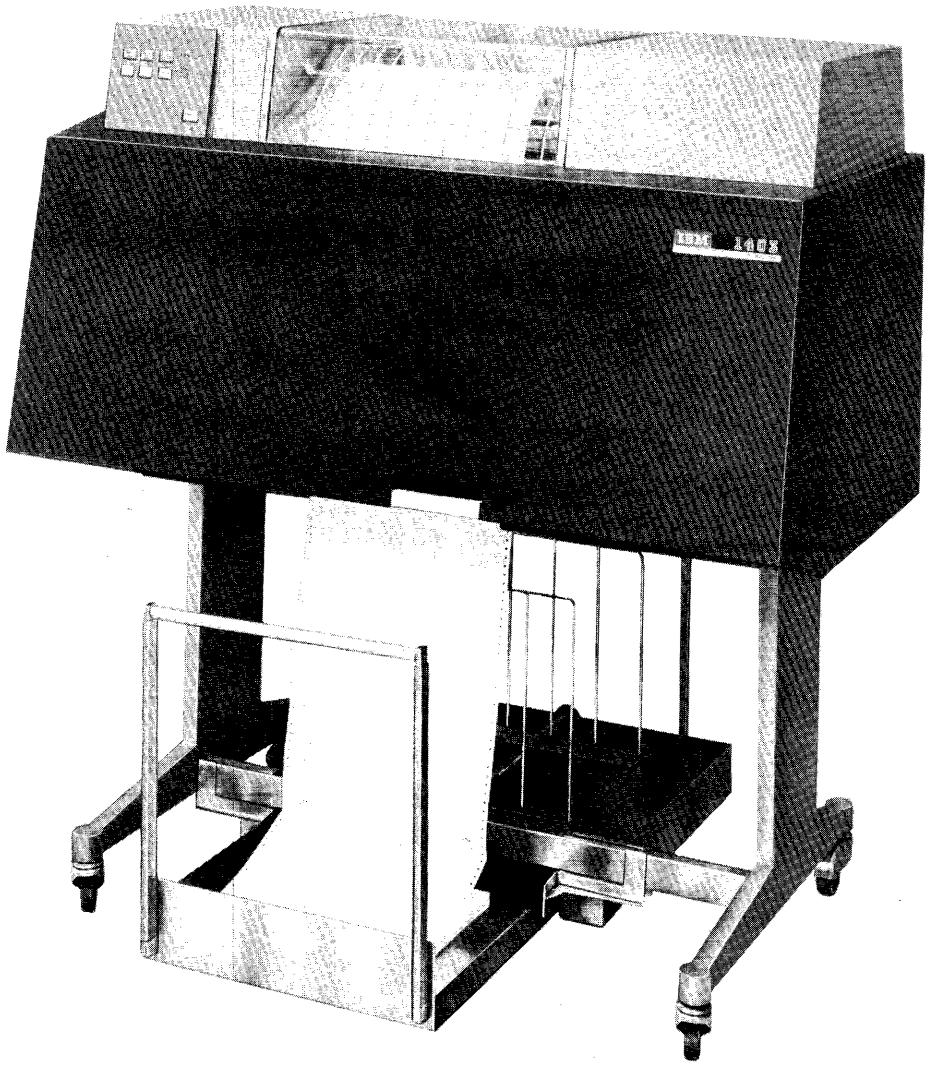


FIGURE 13. IBM 1403 PRINTER

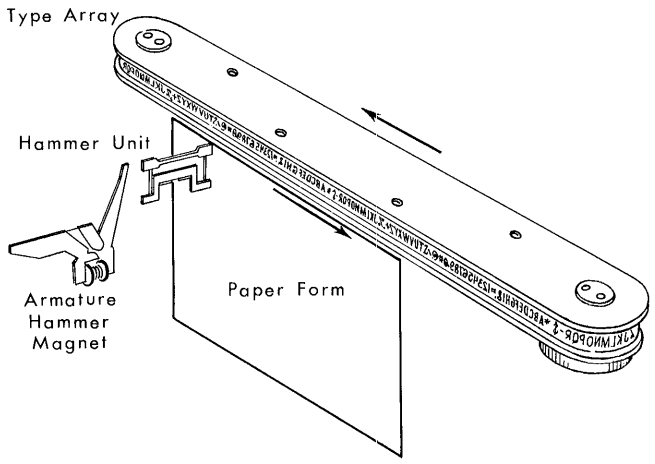


FIGURE 14. SCHEMATIC OF PRINTING MECHANISM

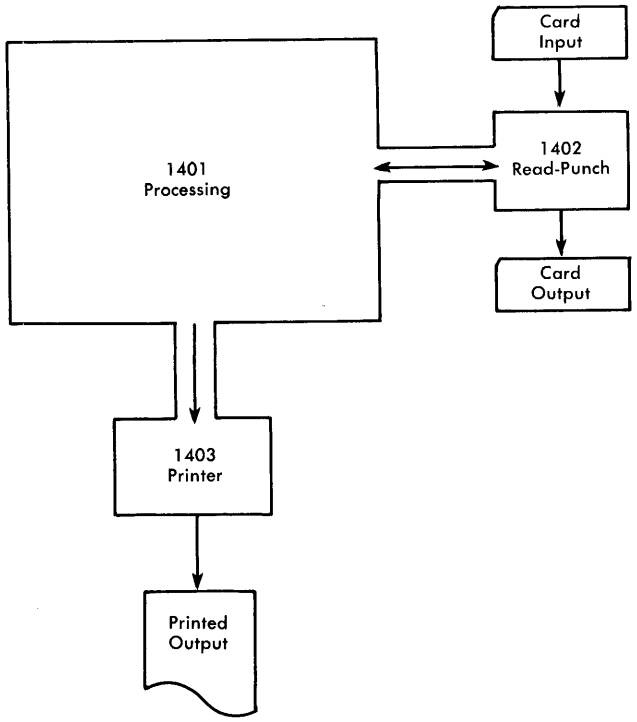


FIGURE 15. GENERAL DATA FLOW SCHEMATIC

printer to the optional Print Storage area. The time required for the transfer of data is 1.15 milliseconds for a 100-character print span; 1.52 milliseconds for a 132-character print span. On completion of this transfer of data, normal program execution is resumed while the print storage area sets up the print mechanism. During this setup, other operations can be performed in the normal manner. Only one instruction that involves the printer can be executed at any one time.

Method of Printing

The alphabetic, numerical, and special characters are assembled in a chain (Figure 14). As the chain travels in a horizontal plane, each character is printed as it is positioned opposite a magnet-driven hammer which presses the form against the chain (Figure 14).

Before a character is printed, it is checked against the corresponding position in the print area of core storage to insure the accuracy of printed output.

Data Flow

The IBM 1402 Card Read Punch and the IBM 1403 Printer are input and output units for the IBM 1401 Card System. All data passes through the 1401 Processing Unit, where arithmetic and logical functions are performed (Figure 15).

Each operation code is analyzed in the *operation register*. The *A* and *B Registers* contain the data characters at the storage location shown by the A and B Address Registers. The *I Address Register* contains the instruction address (Figure 16).

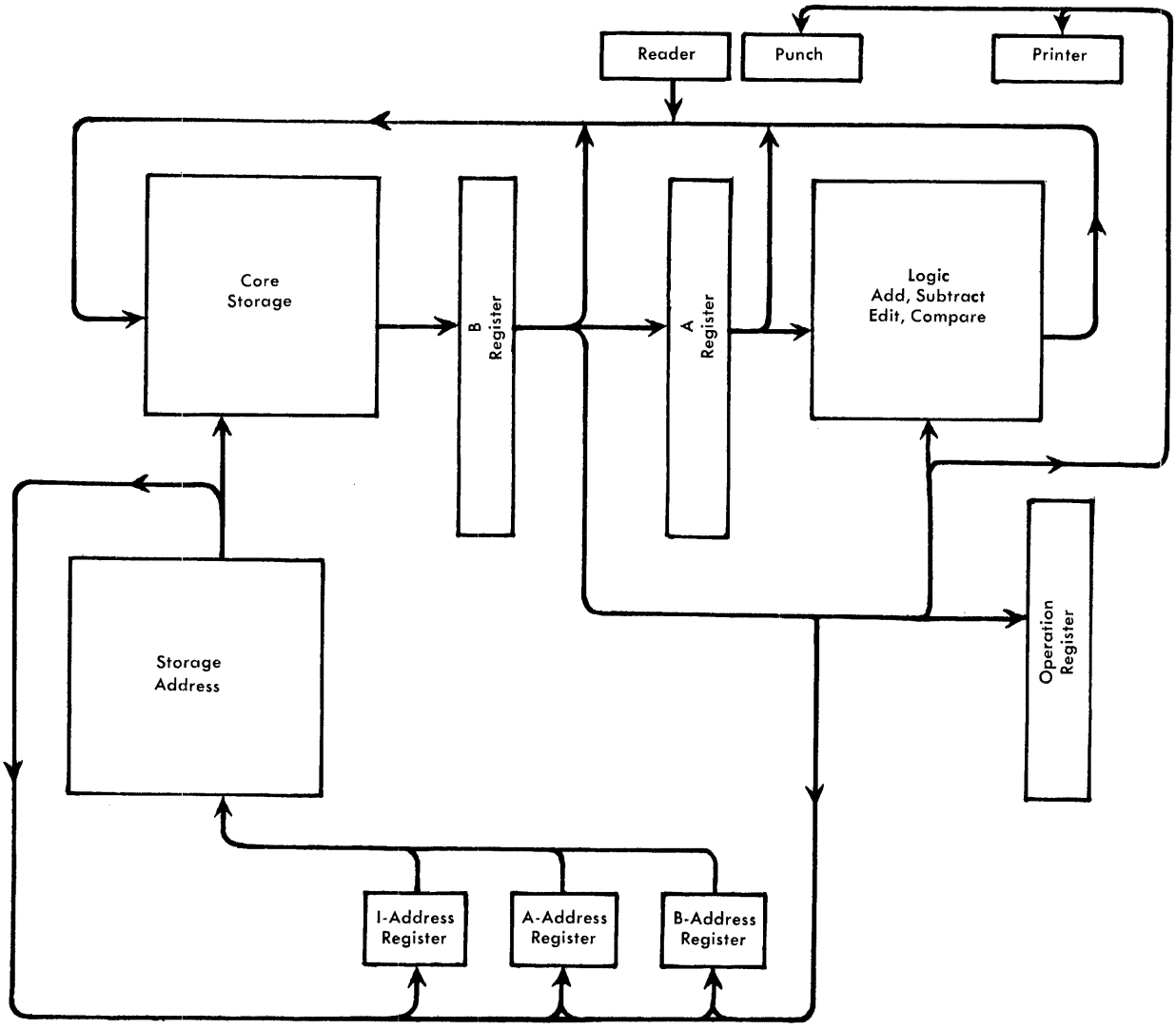


FIGURE 16. IBM 1401 CARD SYSTEM DETAIL DATA FLOW SCHEMATIC

Checking

The IBM 1401 Data Processing System contains many important design factors to insure maximum efficiency and reliability.

The self-checking features built into the 1401 are designed to insure a high degree of error detection. Each data character is represented by an alphanumerical binary code consisting of 6 bits, plus 1 bit for an odd-parity check, and 1 bit for field definition.

Parity Check

The odd-number bit configuration is used for the parity check. The proper number of bits for any given character is known as parity for that character. Word marks are included in the odd-number bit configuration on a parity check when they appear with a character.

When information is moved within the system, a parity check is performed to test the presence of an odd-number of bits for each character being moved.

Validity Check

A validity check is performed on all information when it is read into storage from the card reader, to insure that all characters are valid. A validity check is also made on data in the op code register and address registers. If any invalid characters are detected, the machine stops and the associated check light comes on.

Hole Count Check

The Hole Count feature compares the total number of punches read in a card column at the first reading station, with the total number of punches in the same card column at the second reading station. The Hole Count feature is also effective with the punch side to compare the total number of holes set up for punching in a column, with the number of holes punched in the card column. If in either case, the result of the Hole Count comparison is unequal, the system stops, and check lights indicate the unit involved. If storage is *scanned,* the scanning process stops at the position corresponding to the card column in error.

Word Mark

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.*

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 location 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.

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

All arithmetic and logical functions are performed by the instructions retained in storage. One form of an instruction consists of an operation code followed by two 3-character addresses. The 2-address instruction is required to move data from one location to another, to perform the arithmetic operations of addition or subtraction, to compare two fields, or to edit.

Because the 1401 system uses a variable word-length concept, the length of an instruction can vary from one to eight characters.

Instruction Format

OP (A/I) (B) d
x xxx xxx x

OP is a 1-character operation code, which defines the basic instruction. A word mark is associated with the Op Code position. (This word mark is set under program control or by loading routines.)

(A/I) is a 3-character storage address. A is the location of a data word. I is the address of the next instruction to be executed.

(B) is a 3-character storage address of a data word.

d is a 1-character modifier to the operation code. It can be an alphabetic, numerical or special character. It is positioned as the last character of the instruction and can be used in any instruction length.

NOTE: Underlining any position of an instruction or data word indicates that a word mark is associated with that position.

Instruction Example

OP (A) (B)
A 072 423

This is an *add* instruction. The operation code A, causes the field whose units position is in storage location 072 to be added to the field whose units position is in location 423. This operation continues until a word mark for the high-order position of field B, (which must have a defining word mark) is sensed. The word mark stops the operation being performed and causes the program to advance to the next instruction. If field A is shorter than field B, it must also have a defining word mark.

As stated before, not all instructions have the 2-address form. Others consist of only one address, or no address. This concept results in what is known as *variable-length instructions.*

Examples of the six combinations possible in variable-length instructions are:

Number of Positions	*Operation*	*Instruction Format*
1	READ	OP 1
2	STACKER SELECT	OP d K 2
4	BRANCH UNCONDITIONAL	OP (I) B 400
5	BRANCH UNEQUAL	OP (I) d B 625 /
7	ADD	OP (A) (B) A 072 423
8	TEST CHARACTER AND BRANCH	OP (I) (B) d B 650 080 4

Addressing

The 1401 processes data by following a series of stored instructions. The storage unit stores both the instructions and the data. Each position in storage can be addressed. The high-order position of a field in storage is identified by an associated *word mark.*

An instruction in core storage is addressed by the location of its high-order position. The machine reads the instruction from left to right until it senses the word mark associated with the next instruction. The final instruction in the program must have a word mark set at the right of the low-order position.

The high-order character is the operation code, with an associated word mark which is set by the program when the instruction cards are loaded. In contrast to this, a data word is read from right to left until a word mark is sensed with its own high-order position. In addressing a data word, we specify its units position.

Input-Output Storage Assignments

Certain areas of storage are reserved for the use of the input-output devices. The assignments are such that a correlation is achieved between input-output columns and/or print positions. The storage location assignments are:

card input 001 through 080
card output 101 through 180
print output 201 through 300 (or 332, as needed).

Storage locations 081 through 099 and 181 through 200 can store other data used for normal processing. Storage locations 000 and 100 are also available for normal use except during a read or punch operation (Figure 17).

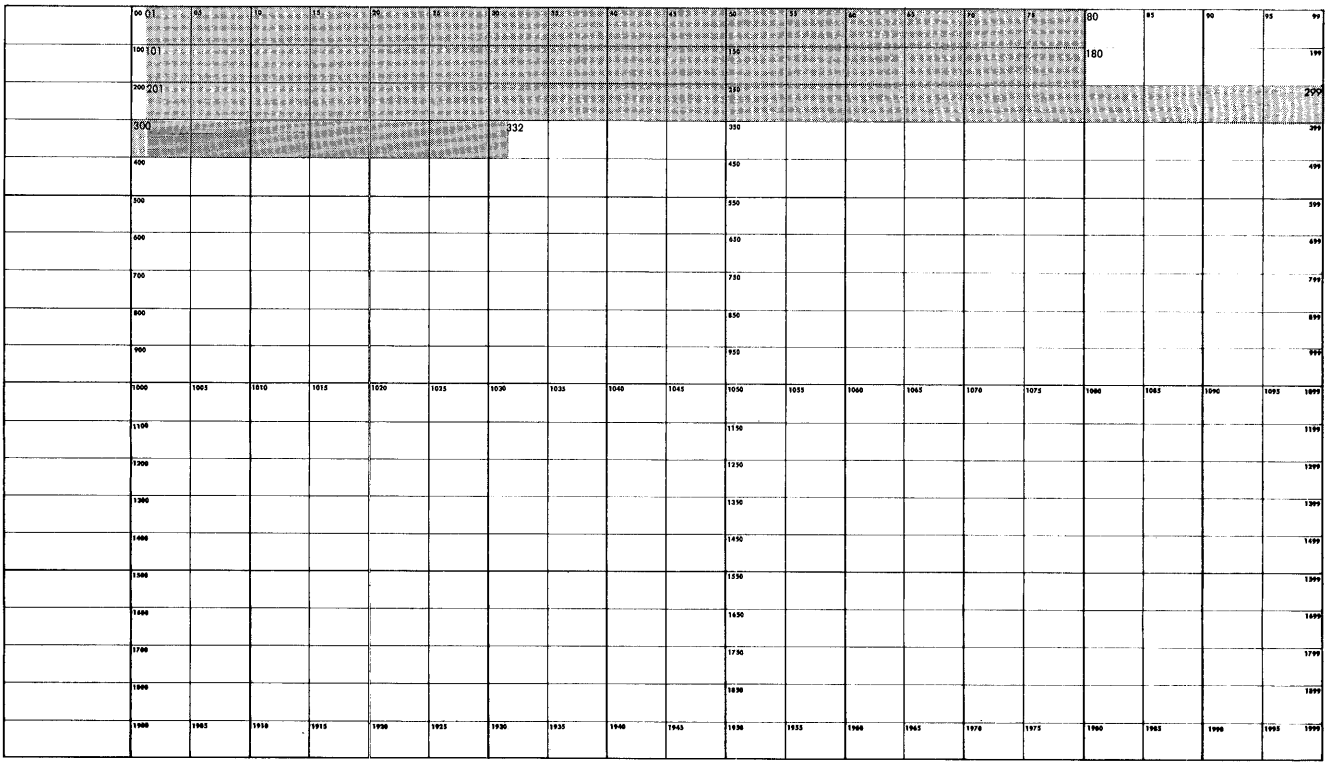


FIGURE 17. STORAGE LAYOUT

Address Registers

Three address registers are incorporated in the IBM 1401 Processing Unit. Two address registers control the transfer of data from one storage location to another; the other, controls the program location sequence.

1. The A address register contains the storage location of the data in the (A) portion of an instruction. The number in this register is decreased by 1 after the execution of the *storage cycle* that involves the (A) address.
2. The B address register contains the storage location of the data in the (B) portion of an instruction. The number in this register is decreased by 1 after the execution of the storage cycle that involves the (B) address.
3. The location of the next instruction character to be used by the stored program is contained in the I (Instruction) Address Register.

Figure 18 is a detailed schematic for the loading of a 7-character instruction in the operation code register, in the A and B registers, and in the A and B address registers. Eight storage cycles are required to load this complete instruction in the registers. Each storage cycle takes .0115 milliseconds.

Cycle	*Operation*
1	Enter OP Register
2	Enter thousands and hundreds position of A address register; also A register.
3	Enter tens position of A address register; also A register.
4	Enter units position of A address register; also A register.
5	Enter thousands and hundreds position of B address register, as well as the A register.
6	Tens position of B address register, and the A register.
7	Units position of B address register, and the A register.
8	Next op code enters B register only.

Chaining Instructions

In some programs, it becomes possible to perform a series of operations on several fields that are in sequence in storage. 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 op-

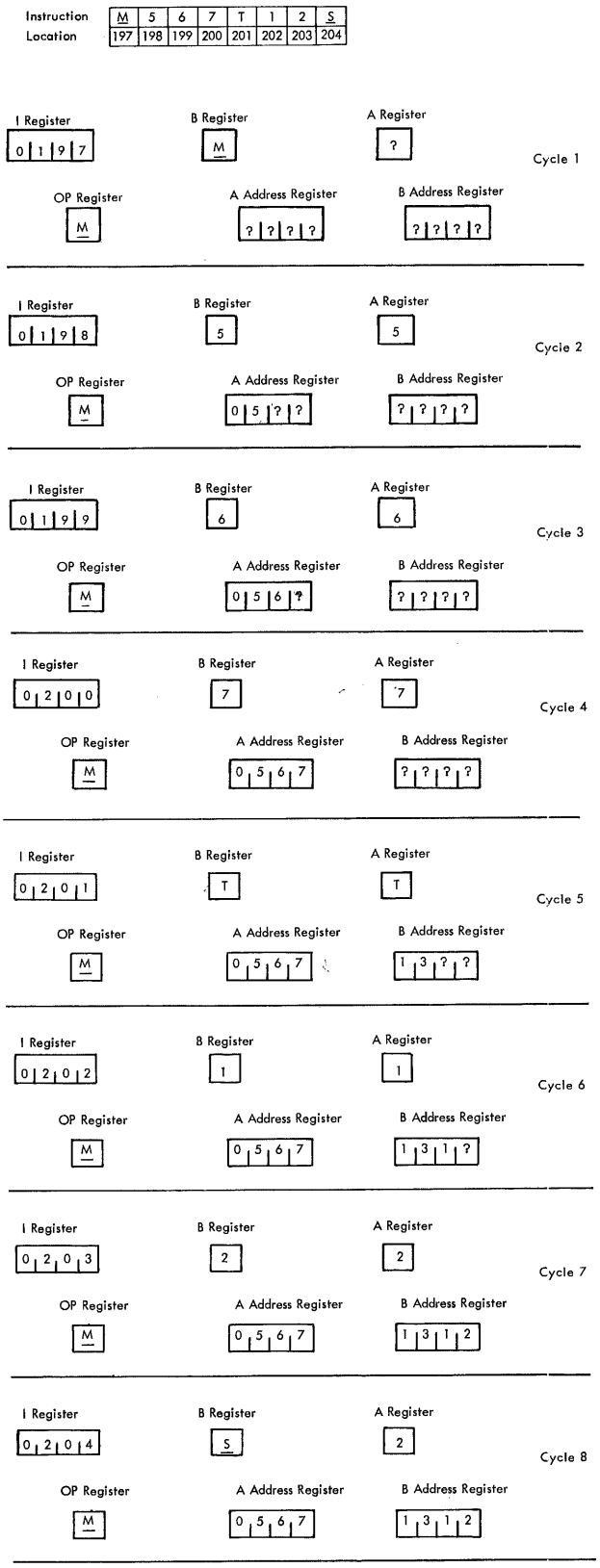


FIGURE 18. SCHEMATIC OF LOADING A 7-CHARACTER INSTRUCTION

erations, 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 ms. 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 the (B) addresses. The following three instructions contain only the operation code for those instructions. The (A) and the (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 milliseconds to execute. Only ten storage positions are required to store these chained instructions.

The ability to chain a series of instructions is not dependent on the use of the same operation code. Chained instructions may have various Op codes. The requirement is that the (A) fields to be operated on 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.

As operation codes are individually explained, instructions that can be chained are so indicated.

Loading Instructions

Before the 1401 can start processing, program instructions must be put into the system. This is accomplished by means of a loading routine, one of which is included in another section of this manual.

Instructions are placed in the machine by the use of load cards. Several different types of load cards condition the 1401 to accept information for processing. They cause word marks to be set at specific storage locations, and load a series of instructions which allow the cards containing the actual program instructions to be stored in their correct locations.

Input-output operation codes control reading and punching data cards, and printing reports. Branching instructions are provided to transfer the program automatically at the completion of a function. More than one function can be initiated by a single instruction.

Input-Output Codes

1 READ

This instruction activates the card feed, and causes all 80 columns of information to be read from the card into the IBM 1401 storage unit, addresses 001 through 080. The word marks for these 80 positions are not disturbed.

1 (I) READ AND BRANCH

Same as the READ instruction, except that when the (I) address follows the READ operation code, the next instruction is taken from the (I) address instead of from the next instruction location in sequence. This produces a branch in the program after the card has been read from the 1402 card-read unit. The position immediately following this instruction must contain a character with a word mark or a blank character with or without a word mark.

2 PRINT

This instruction causes the program to stop, and the data in the print area of storage to be transmitted to the printer. The program continues sequentially, immediately after printing is complete. The print area of storage is designated as addresses 201 through 300 for the basic 1403, and addresses 201-332 for the 1403 equipped with 32 additional print positions. The printer automatically spaces one line after printing unless instructed to do otherwise.

When the system is equipped with the Print Storage optional feature, the program can continue as soon as the data is received in print storage. Thus the interlock time is greatly reduced.

2 (I) PRINT AND BRANCH

Same as the PRINT instruction, except that the location of the next instruction is at location (I). The position immediately following this instruction must contain a character with a word mark or a blank character with or without a word mark.

2 □ PRINT WORD MARKS

This instruction causes each word mark associated with storage addresses 201 through 300 (201-332 for additional print control) to print as the digit "1". The printer automatically spaces one line after printing unless instructed to do otherwise.

2 (I) □ PRINT WORD MARKS AND BRANCH

Same as PRINT WORD MARKS (2 □) instruction, except that the location of the next instruction is at address (I).

3 PRINT AND READ

This instruction combines the operations of READ (1) and PRINT (2). The printer is given priority and operates first. However, a signal to start the card read unit can be given before the end of the print operation. Thus actual reading from the card may start shortly after completion of the print operation.

When the system is equipped with the Print Storage optional feature, the program can continue as soon as the data is received in print storage.

3 (I) PRINT, READ, AND BRANCH

Same as the previous instruction, except that the location of the next instruction is at address (I). The position immediately following this instruction must contain a character with a word mark or a blank character with or without a word mark.

4 PUNCH

This instruction causes the data located in addresses 101 through 180 to be punched into an IBM card.

4 (I) PUNCH AND BRANCH

Same as the PUNCH (4) instruction, except that the location of the next instruction is at address (I). The position immediately following this instruction must contain a character with a word mark or a blank character with or without a word mark

5 READ AND PUNCH

This instruction combines the READ (1) and PUNCH (4) operations as described. The machine can, in effect *simultaneously* read and punch when this instruction is used, because these two operations can overlap.

5 (I) READ, PUNCH, AND BRANCH

Same as the previous instruction except that the location of the next instruction is at address (I). The position immediately following this instruction must contain a character with a word mark or a blank character with or without a word mark.

6 PRINT AND PUNCH

This instruction combines the functions of PRINT (2) and PUNCH (4). The printer has priority. However, the punch is signalled to start before the end of the print operation, so that actual punching into the card may start shortly after the print operation is completed.

When the system is equipped with the Print Storage optional feature, the program can continue as soon as the data is received in print storage.

6 (I) PRINT, PUNCH, AND BRANCH

Same as the previous instruction except that the location of the next instruction is at address (I). The position immediately following this instruction must contain a character with a word mark or a blank character with or without a word mark.

7 PRINT, READ, AND PUNCH

This instruction combines the functions of READ (1), PRINT (2), and PUNCH (4). The printer has priority and operates first, with the read and punch process overlapped as previously explained.

7 (I) PRINT, READ, PUNCH, BRANCH

Same as the previous instruction, except that the location of the next instruction is at address (I). The position immediately following this instruction must contain a character with a word mark or a blank character with or without a word mark

8 READ RELEASE (OPTIONAL)

This instruction causes the card reader to start the next cycle, and allows processing to continue. A READ instruction must then be given prior to the time the reader is ready to read the 9-row of the card. If the instruction is not given early enough, the card passes the brushes without being read, and the machine stops. This instruction allows a gain of 20 milliseconds of processing time between successive card read cycles.

9 PUNCH RELEASE (OPTIONAL)

This instruction causes the card punch to start the next cycle, and allows processing to continue. A PUNCH instruction must then be given, prior to the time the 1401 begins emitting the information to the punch for punching the 12-row of a card. If the instruction is not given early enough, the card passes the punch station without being punched, and the machine stops. This instruction allows a gain of 35 milliseconds of processing time between successive punch cycles.

Add and subtract, and the optional multiply and divide operation codes, perform the arithmetic operations. Because the operations are performed within core storage, no accumulators or counters are necessary. Thus, the capacity for arithmetic functions is not limited by a predetermined number of counter positions.

Arithmetic Operation Codes

A (A) (B) ADD

This instruction causes the numerical data at the (A) address to be added algebraically to the numerical data at the (B) address. If the two numerical fields contain different signs, a complement addition takes place. A word mark associated with the B field stops the ADD operation. If the A field is shorter than the B field, a word mark should be inserted with the A field to stop transmission of data from the A field.

A negative field is indicated by a "B" bit and the absence of an "A" bit in the units position of that field (11-punch in a card). Any other "A" or "B" bit combination in the units position is considered a positive sign. For compatibility with other machines all positive numbers should always be indicated by either "A" and "B" bits (12-punch in a card) or by the absence of both "A" and "B" bits.

In a true ADD operation (both fields have the same sign), the zone bits in the units position of the B field are unaffected. In a complement ADD operation (unlike signs), the zone bits in the units position of the B field are always set to "A" and "B" bits (12-punch in a card) for a positive result, but remain as a "B" bit only for a negative result (11-punch in a card).

NOTE: Zone bits located in any character position within the B field other than the *sign* (units) position or the *overflow* (high-order) position are lost (removed) in any arithmetic operation. If the sign of the B field is required to change as a result of the arithmetic operation, the machine takes an automatic recomplementing cycle to restore the data to true form.

Upon completion of the ADD operation, the address registers contain the addresses of the fields to the left of the original A and B fields; therefore, the ADD instruction can be chained when sequential fields are being used provided the A field is equal to or less than the B field.

The ADD instruction can also be executed using only the (A) address as follows:

A (A).

In this form the (A) portion and the implied (B) portion are the same, and are added to each other. This form of the instruction can be used to double the A field. The result is located in the A field.

Overflow

An overflow condition can occur as a result of a true arithmetic operation if the B field is not large enough to accommodate the answer. An overflow condition sets an indicator, which can be tested by a BRANCH instruction. (This indicator is not reset until the next ADD or SUBTRACT instruction is given). The overflow condition also causes an "A" bit to be stored in the high-order position of the B field. If other data is added to this same B field without adjusting the length of the field, additional overflows can occur and change the "AB" bit configuration as follows:

	Zone Bits in High-Order Position of B Field
1st Overflow	"A" bit, No-"B" bit
2nd Overflow	No-"A" bit, "B" bit
3rd Overflow	"A" bit, "B" bit
4th Overflow	No-"A" bit, No-"B" bit
5th Overflow	Same as for 1st overflow, etc.

Because the machine signals an overflow condition by placing zone bits in the high-order position of the field, overflow can be used for *address modification*. Because the address of storage positions 1000 and up are represented by an alphabetic or special character, the high-order position of an address can be modified.

Example: Data word A is 550
Data word B is 840

If the instruction ADD (A) (B) is executed the result in location (B) is 390 with an "A" bit over the high-order position. The 1401 translates this BCD coding as the alphabetic character "T". Thus the data word T90 is in (B). If this is used as an address it represents storage location 1390.

> NOTE: If the A and B fields contain alphabetic instead of numerical digits in the positions corresponding to the high-order position of the B field, the result of a true ADD operation contains the sum of the zone bits of the positions corresponding to the high-order position of the B field in the high-order position of the result. The digit portions of both fields add correctly. However, the high-order position of the B field could be modified by the overflow conditions.

An overflow condition cannot occur as a result of a complement-add arithmetic operation.

S (A) (B) SUBTRACT

This instruction is the same as the ADD (A) instruction, except that the A field is algebraically subtracted from the B field. If both fields contain the same sign, a complement-add arithmetic operation results.

This instruction can also be executed using only the (A) address as follows:

S (A)

In this form the A field and the implied B field are the same, and the A field is subtracted from itself. This form of instruction can be used to clear the A field and put zeros in it.

The SUBTRACT instruction can be chained for sequential data fields.

$\overset{+}{0}$ (A) (B) RESET ADD

This instruction is similar to the ADD (A) instruction, except that the B field is, in effect, set to zero before the A field is added to the (B) location.

This instruction is not the same as the MOVE instruction. If the A field is shorter than the B field in a reset add instruction, high-order positions of the B field are filled with zeros. With a MOVE instruction, these positions are unaffected. The A field must have an associated word mark only if it is shorter than the B field. This instruction can be chained.

$\overline{0}$ (A) (B) RESET SUBTRACT

This instruction is similar to the RESET ADD instruction, except that the A field is subtracted (algebraically) from the B field, which is in effect set to zero before the A field data is subtracted from it. This instruction can be chained. It can also be used without a (B) address, which in effect causes a sign change in the A field.

@ (A) (B) MULTIPLY (OPTIONAL)

This code causes the multiplicand (data located in the A field) to be repetitively added to the data in the B field. The B field consists of the multiplier in the high-order positions, and enough additional positions for the development of the product. At the completion of the operation, the units position of the product is at the location given by the (B) address. The multiplier can be retained in another area of storage if it is required for further use in the program.

The rules for multiplication are:

1. The product is developed in the B field. The length of the B field is determined by adding "1" to the sum of the number of digits in the multiplicand and multiplier fields.

   Example:

   1246 4-digit multiplicand
   x543 3-digit multiplier
   + 1
   8 positions must be allowed in the B field.

2. The multiplicand and multiplier data words must have a sign in the units position, and a word mark associated with the high-order positions.

3. In all cases "A" and "B" bits for plus signs, and "B" bits for minus signs, must appear in the units position of the fields. The multiply operation uses algebraic sign control.

   As a result of a multiply operation, the multiplier in the B field is destroyed. However, it is still available at its original location. This instruction should not be chained. An example of a multiply routine is shown in Figure 19.

IBM 1401 PROGRAM CHART — FORM X24-6437 PRINTED IN U.S.A.

Program: MULTIPLICATION

Programmer: Date:

Step No.	Inst Addr	Instruction OP	A/I	B	d	Remarks	Effective No. of Characters Inst.	Data	Total
		L	065	605		Load multiplier in the high order positions of the (B) field.			
		@	502	610		Multiply 1246 x 543 and develop product beginning in position 610 of storage (8 positions maximum).			
		L	610	178		Load product to output area			

The size of the product field is 1 + 4 + 3 = 8. Therefore, the multiplier is placed in the 3 high order positions of the (B) field, which are storage locations 603-4-5.

Location of Data Word	Data Word	Description of Data
502	1246	Multiplicand
065	543	Multiplier
610		Product

FIGURE 19. MULTIPLY ROUTINE

% (A) (B) DIVIDE (OPTIONAL)

This instruction causes the dividend in the low-order positions of the B field, to be divided by the divisor located in the A field, and the resultant quotient to be stored in the high-order positions of the B field. Any remainder is located in the low-order positions of the B field.

Prior to a DIVIDE operation the dividend is loaded in the B field under these conditions:

1. The field reserved for the divide operation must be twice the size of the dividend field. For example: if there are 4 digits in the dividend, then 8 positions must be reserved in B field.
2. A word mark must be set to indicate the high-order position of the B field, to signify the high-order digit of the quotient.
3. A word mark must be associated with the high-order digit of the dividend. This word mark signals the end of the DIVIDE operation.
4. Plus and minus signs must be indicated by "AB" bits, and "B" bits, respectively, in the units positions of the dividend and of the divisor.
5. The dividend is loaded in the low-order positions of the B field (Figure 20). The instruction % (A) (B) starts the divide operation.

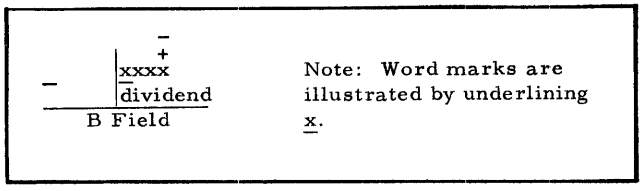


FIGURE 20. DIVIDEND LOAD LOCATION IN B FIELD

At the completion of division:

a. The quotient is in the high-order half of the B field.

b. The remainder is in the low-order half of the B field.

c. The sign of the quotient is over the units position of the quotient field.

Figure 21 shows the Result of the Divide Operation.

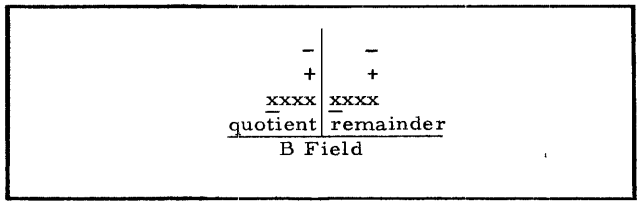


FIGURE 21. LOCATION OF THE RESULTS OF A DIVIDE OPERATION

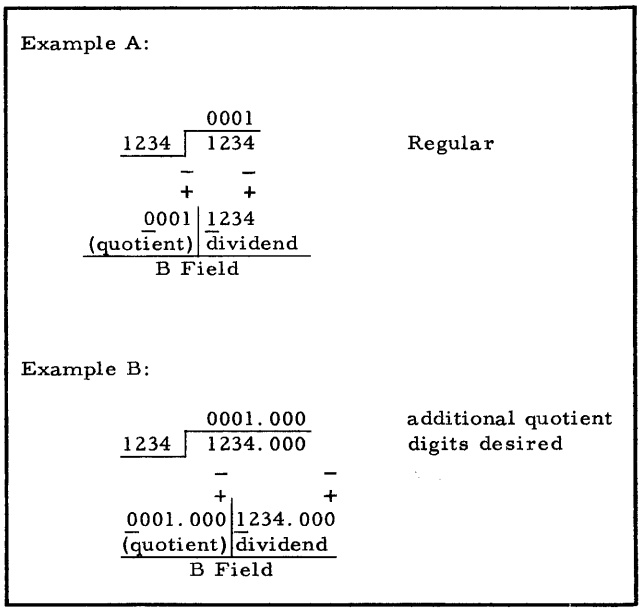


FIGURE 22. ADDITIONAL QUOTIENT DIGITS

Extra zeros can be added to the dividend prior to a DIVIDE operation when a larger quotient is required. For each additional quotient digit desired, place one zero to the right of the dividend, and allow extra positions in the quotient fields, as shown in Figure 22. Figure 23 is an example of a divide operation. Also see *Multiplication and Division Subroutines* for machines not equipped with this optional feature.

IBM 1401 PROGRAM CHART

FORM X24-6437-0 PRINTED IN U.S.A.

Program: DIVISION

Programmer: Date:

Step No.	Inst Addr	Instruction OP	A/I	B	d	Remarks	Effective No. of Characters Inst.	Data	Total
		,	603			Set word mark for high order position of quotient.			
		L	502	610		Load dividend into divide area (B).			
		%	065	610		Divide 543 into 1246 and develop quotient beginning in			
						storage position 606. Remainder if any is in storage			
						position 610.			
		L	606	178		Load quotient to output area.			

Location of Data Word	Data Word		Description of Data
502	1246	(B)	Dividend
065	543	(A)	Divisor
606			Quotient (units position)
610			Remainder (units position)

FIGURE 23. DIVIDE ROUTINE

Logic Operations

The decision-making ability of the IBM 1401 Data Processing System is based on the logic operation codes. As a result of testing for conditions that can arise during processing, the program can be directed to predetermined sets of instructions, or subroutines.

Logic Operation Codes

B (I) UNCONDITIONAL BRANCH

The next instruction is taken from the (I) address. This instruction must be followed by either a character with a word mark or a blank character (with or without a word mark).

B (I) d TEST AND BRANCH

The d-character specifies the condition tested. If the condition is satisfied, the next instruction is taken from the (I) address. If the test condition is not met, the next sequential instruction is taken. A chart of characters that are valid for the d-position and the type of test they indicate, is shown in the table included (Figure 24). This table also includes testing for high, low, or equal, when the High-Low-Equal Compare feature is installed.

The conditions tested are not reset by this instruction except as noted by ‡. When carriage tape-channels 9 or 12 are sensed, corresponding indicators are set. These carriage channel-indicators are reset when any other carriage tape-channel is sensed. The comparisons are reset by the next COMPARE instruction, and the overflow is reset by the next arithmetic instruction.

B (I) (B) d TEST CHARACTER AND BRANCH

This instruction causes the character at the (B) address to be tested for the same bit configuration as the d-character. If it is the same, the program branches to the (I) address. Otherwise, the next sequential instruction is executed. The d-character can be any combination of the six binary-coded-decimal code bits (BA 8421). This instruction can be chained.

EXAMPLE: B (601) (350) b. The d-portion is blank. If location 350 is blank, the program branches to location 601 for the next instruction.

d	*Branch on:*	*d*	*Branch on:*	*d*	*Branch on:*	*d*	*Branch on:*
bl	Unconditional	A	Sense Switch A "Last Card" Switch	K	End of Reel * ‡	/	Unequal Compare B ≠ A
9	Carr. Chan. #9	B	Sense Switch B *	L	Tape Channel Transmission Error * ‡	S	Equal Compare B = A *
@	Carr. Chan. #12	C	Sense Switch C *			T	Low Compare B < A *
		D	Sense Switch D *	+o	Reader Error if I/O Check Stop Switch OFF ‡	U	High Compare B > A *
		E	Sense Switch E *	−o	Punch Error if I/O Check Stop Switch OFF ‡	W	Divide Overflow *
		F	Sense Switch F *	+	Printer Error if I/O Check Stop Switch OFF ‡	Z	Overflow ‡
		G	Sense Switch G *			%	Procesing Check with Process Check Switch OFF ‡

* optional
‡ Condition reset by a Test and Branch instruction

FIGURE 24. d-CHARACTER FOR BRANCH INSTRUCTION

If this instruction is chained, B (601) (350) b BBB, the program also tests locations 349, 348 and 347 for blanks. Thus, the entire field is tested for blanks, and sends the program to location 601 if any blank positions are found in the field.

Word marks do not affect this instruction.

V (I) (B) d TEST FOR ZONE OR WORD MARK AND BRANCH

This is a single-character-test instruction, which tests the character located at address (B) for a specific condition as specified by the d-character, and branches if the condition is met as follows:

d-Character	Branch to (I) if address (B) contains:
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.

Move and Load Codes

M (A) (B) MOVE

This instruction causes the data in the A field to be stored in the B field. If both fields are the same length, only one of the fields need have the defining word mark. If the fields are different lengths, the first word mark encountered defines the length of both fields, and stops the operation. Sensing an A word mark first allows the completion of one more B cycle before stopping the operation. The word marks themselves are not affected by the move operation, nor is the data in the A field. At the end of this operation, the A address register and B address register contain the addresses of the units positions of the fields to the immediate left of the high-order positions of the preceding locations. Thus, the next instruction can be chained, if it is to use the locations that are provided in the A and B address registers.

M (A)

The address in the B address register is used as the address of the B field. This instruction can be used to assemble fields in sequential order.

Figure 25 is a detailed schematic, showing movement of a 3-character data word from the A field to the B field.

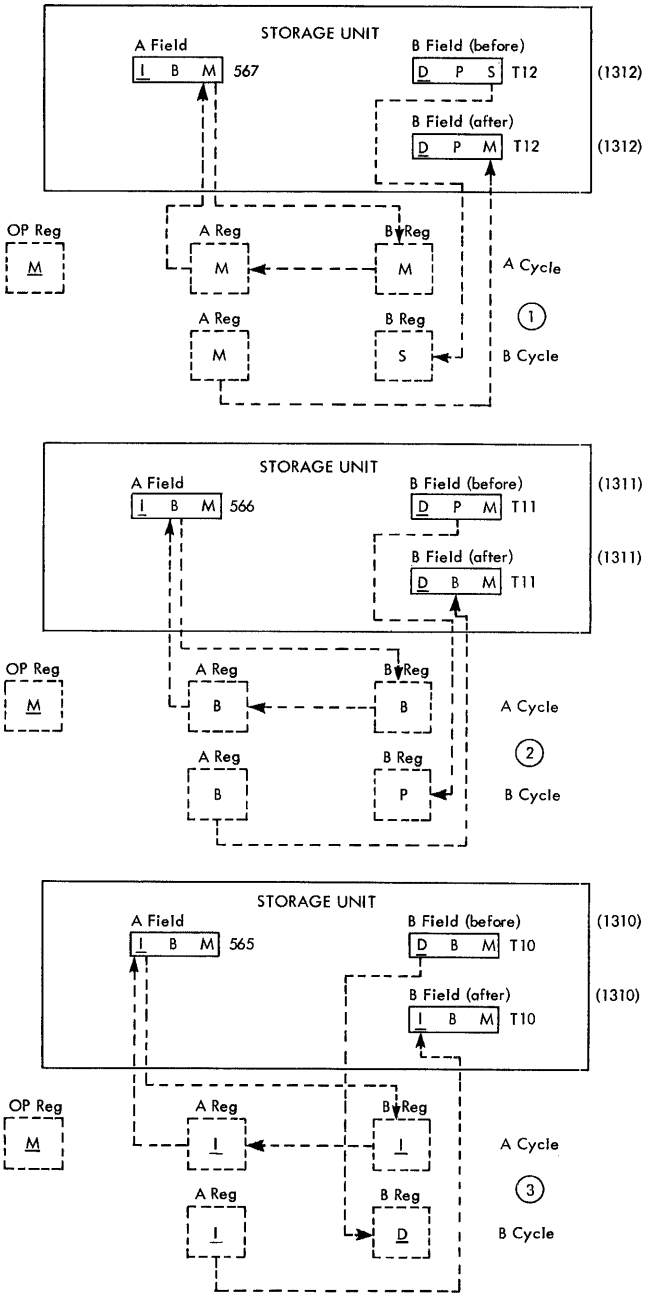


FIGURE 25. SCHEMATIC OF A 3-CHARACTER MOVE INSTRUCTION

	Cycle	Type of Cycle	B Register	A Register	Address Registers at End of Cycle			Op Register	Remarks
					I	A	B		
	1	I_{OP}	Word Mark M	?	002	???	???	M	Read Instruction
A Address Developed	2	I_1	5	5	003	5??	???	M	
	3	I_2	6	6	004	56?	???	M	Load A Address Register
	4	I_3	7	7	005	567	???	M	
B Address Developed	5	I_4	T	T	006	567	13??	M	
	6	I_5	1	1	007	567	131?	M	Load B Address Register
	7	I_6	2	2	008	567	1312	M	
	8	I_7	Word Mark OP	2	008	567	1312	M	OP code of next instruction.
	9	A	M	M	008	566	1312	M	Execute move inst.
	10	B	S	M	008	566	1311	M	
	11	A	B	B	008	565	1311	M	
	12	B	P	B	008	565	1310	M	
	13	A	Word Mark I	Word Mark I	008	564	1310	M	
	14	B	Word Mark D	Word Mark I	008	564	1309	M	Completion of MOVE Instr.
	15	I_{OP}	Word Mark OP	Word Mark I	009	564	1309	Next OP Code	Read next instr.

FIGURE 26. SINGLE CYCLE OPERATION OF MOVE INSTRUCTION (M 567-T12)

Figure 26 is the single-cycle operation chart, that explains the schematic.

Z (A) (B) MOVE AND ZERO SUPPRESS

This instruction causes the data in the A field to be stored in the B field. The B field contains blanks instead of zeros to the left of the first significant digit. At the completion of the MOVE, the A address register contains the address of the field immediately to the left of the A field and the B address register contains the address of the units position of the B field plus one (1). Therefore, this instruction should not be chained. Only a word mark with the A field stops the transmission of data. This code removes the sign from the units position of the resultant field.

D (A) (B) MOVE DIGIT

This instruction causes the numerical portion (8-4-2-1 bits) of the *single* character in the A address to be written in the B address. The zone portion ("AB" bits) at both addresses is not affected. Because this is a *single-character* operation, no word marks are required with the A address or the B address. This instruction can be chained.

Y (A) (B) MOVE ZONE

This instruction is similar to MOVE DIGIT (D), except that only the zone ("AB" bits) are moved.

This instruction can be chained.

L (A) (B) LOAD

This instruction is similar to MOVE (M), except that the length of the word in the A field must be defined by a word mark. The word mark for the A field is transferred to the B field, and all B field word marks up to this newly moved word mark are cleared. This instruction is commonly used to load data into the printer, or punch areas of storage, or to bring data or instructions from the reader area of storage to another location. The word mark for the A field stops the transfer.

L (A)

The address in the B address register is used as the address of the B field.

This instruction can be chained.

, (A) (B) SET WORD MARK

This instruction can include one or two addresses, and causes a word mark to be set for each

of the specified addresses without disturbing the data at these addresses. This instruction can be chained.

⌑ (A) (B) CLEAR WORD MARK

Same as the SET WORD MARK (,) instruction, except that the word marks are cleared at the specified addresses. This instruction can be chained.

Miscellaneous Operation Codes

F d FORMS CONTROL

This instruction causes the carriage to move as specified by the d-character. A numerical 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 27) shows the effect of the d-character. In order to maintain the highest possible machine speed, the immediate skip or space instruction should be given as early in the program as possible. If the carriage is in motion when a FORMS CONTROL 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.

F (I) d FORMS CONTROL AND BRANCH

Same as the previous instruction except that the next instruction is taken from the (I) address.

K d STACKER SELECT

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

d-character	*Feed*	*Stacker Pocket*
1	Read	1
2	Read	8/2
4	Punch	4
8	Punch	8/2

This instruction must be given in the first 10 milliseconds of process time after a read operation has been completed, in order to select the card that has just been read. Giving this instruction at any other time is ineffective for selecting a card in the card read unit, and the card enters the NR (non-selected reader) pocket (Figures 11 and 12).

To select a card in the punch unit, this instruction can be given any time after the PUNCH instruction has been completed, as long as it is before the *next* PUNCH instruction. Even when this instruction is given immediately after a card is punched, the card is not selected into the proper pocket until after the next PUNCH instruction is executed. It is not selected as specified by the STACKER SELECT instruction if during this second punch cycle a hole-count check occurs.

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	ǂ ǂ	Channel 10
#	Channel 11	•	Channel 11
@	Channel 12	⌑	Channel 12

d	*Immediate Space*
J	1 space
K	2 spaces
L	3 spaces

d	*After Print-space*
/	1 space
S	2 spaces
T	3 spaces

FIGURE 27. d-CHARACTER FOR FORMS CONTROL

An error check in the card read unit stops the entire system at the completion of the READ instruction, and the card in error is automatically selected into the NR pocket.

A punch error condition overrides the stacker-select operation, and the card enters the NP (non-selected punch) pocket.

K (I) d STACKER SELECT AND BRANCH

Same as the previous instruction except that the location of the next instruction is at address (I).

C (A) (B) COMPARE

This instruction causes the data in the B field to be compared to an equal number of characters in the A field. It compares the bit configuration of each character in the two fields. When the B field is longer than the A field, an unequal-compare results. When the A field is longer than the B field, the comparison is stopped by the B word mark. The result of the comparison is stored in the machine for later use by a BRANCH CONDITIONAL instruction. The COMPARE instruction should not be chained.

/ (A) CLEAR

This instruction is used to clear an area of storage (up to 100 characters) of data and word marks.

It causes clearing in all positions from address (A) down to the nearest hundreds position. The cleared area is set to blanks.

EXAMPLE: if the (A) address is 563, all positions 563 through 500 are cleared. This instruction can be chained.

/ (I) (B) CLEAR AND BRANCH

Same as the previous instruction (the B address is the start location for CLEAR) except that the location of the next instruction is indicated by address (I).

N NO OPERATION

This operation code can be substituted for the operation code of any instruction to make that instruction ineffective.

. STOP

This instruction causes the machine to stop, and the stop-key light turns on. Pressing the start key causes the program to resume from the next instruction in sequence.

. (I) STOP AND BRANCH

Same as the previous instruction, except that when the start key is pressed, the location of the next instruction is at the (I) address.

A blank character immediately following this instruction acts as a word mark to terminate this instruction.

Editing

Editing in the IBM 1401 Data Processing System is automatic control of zero suppression, insertion of identifying symbols, and punctuation of an output field. This function can be performed with two simple instructions. One single edit instruction can cause all desired commas, decimals, dollar signs, asterisks, credit symbols, and minus signs, to be automatically inserted in a numerical field. In addition, unwanted zeros to the left of significant digits are suppressed (Figure 28).

Example:

		(A)	(B)
Edit Instruction		E (789)	(300)
	A Field (data)		B Field (control word)
Storage	00257426		$\underline{\$}$bbb, bb0. bb&CR&**
Result of Edit			B Field
Storage	00257426		$ 2,574.26 **

FIGURE 28. EDITING

The process is analagous to the arithmetic operation. In addition, a character from the A field is read from storage; then a character from the B field is read from storage; an operation (addition) is performed on the two characters, and the result is written back into storage.

Likewise, in editing, two fields are needed: the data field, and a control field. The control field indicates how the data is to be edited. It specifies the location of commas, decimals, conditional CR and minus symbols, and indicates where zero suppression is to occur.

The two fields are read from storage alternately, character-by-character, as they are in the addition process, but are under control of the editing rules.

The control word is divided into two parts: the body (used for punctuating the A field) and the status portion (which contains the sign symbols and *'s). Printing sign symbols is in part controlled by the sign of the A field.

To edit a field, a LOAD instruction loads the control word in the output area; the EDIT instruction moves the data to the output area, and performs the editing function.

E (A) (B)

This instruction takes the data in the A field, modifies it by the contents of the edit-control word in the B field, and stores the results in the B field. The type of modification is controlled by the following set of rules.

RULE 1. All numerical, alphabetic and special characters can be used in the control word. However, some of these have special meanings as listed below.

Control Character	*Function*
b (blank)	Replaced with the character from the corresponding position of the A field.
0 (zero)	Used for zero suppression. 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.
. (period)	Undisturbed in the punctuated data field, in the position where written. Functions as a significant character unless *Expanded Print Edit* feature installed (See Expanded Print Edit).
, (comma)	Undisturbed in the punctuated data field, in the position where written, unless zero suppression takes place, and no significant numerical characters are found to the left of the comma.
CR (credit)	Undisturbed if the data sign is negative. It is blanked out if the data sign is positive. Can be used in body of control word without being subject to sign control.
– (minus)	Same as CR.
& (ampersand)	Causes a space in the edited field. It can be used in multiples.
* (asterisk)	Can be used in singular or in multiple, usually to indicate class of total. When used with the Expanded Print Edit optional feature, this takes on a special meaning (see Expanded Print Edit).
$ (dollar sign)	Undisturbed in the position where written. When used with the Expanded Print Edit optional feature, this takes on a special meaning (see Expanded Print Edit).

RULE 2. A word mark with the high-order position of the B field controls the EDIT operation.

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

RULE 4. The body of the control word is defined as that portion beginning with the right-most blank or zero, and continuing to the left until the A field word mark is sensed. The remaining portion of the control field is referred to as the *status* portion.

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

RULE 6. *Zero Suppression.* This is the deletion of unwanted zeros at the left of significant digits in an output field (Figure 29).

EXAMPLE:

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 29. ZERO SUPPRESSION

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

Forward Scan:

1. The positions in the output field at the right of this special zero are replaced by the corresponding digits from the A field.
2. When the special zero is detected in the control field, it is replaced by the corresponding digit from the A 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. All zeros and punctuation at the left of the first significant character (up to and including the zero suppression code position) are replaced by blanks in the output field.
2. When the automatically set zero suppression word mark is sensed, it is erased and the operation ends.

RULE 7. Any A field data that has not been moved before the word mark for the control field is sensed, does not appear in the edited output data. The data field can contain fewer, but should not contain more, positions than the number of the blanks and zeros in the body of the control word. Dollar signs and asterisks are included in the control word with the Expanded Print Edit feature.

An illustration of the application of these rules is shown in Figure 30.

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	07??	07??	7	7	7	same	Load A Address Register
3	I_2	004	078?	078?	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	0389	3	3	3	same	Load B Address Register
6	I_5	007	0789	0309	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-Rule 1 and 6
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	0291	4	0	4	$bb2,574.26bbbb**	Rule 6

FIGURE 30. STEP BY STEP EDITING OPERATION

Expanded Print Edit

The basic operations of the EDIT instruction can be expanded by an optional Expanded Print Edit feature. This provides the functions of Asterisk Protection, Floating Dollar Sign, Decimal Control, and Sign Control Left.

Asterisk Protection

When it is necessary to have asterisks appear at the left of significant digits, the asterisk protection feature is used (Figure 31).

EXAMPLE:	
A field	00257426
Control word (B field)	bbb,b*0.bb&CR
Forward scan	002,574.26 CR
Reverse scan	**2,574.26 CR
Results of edit	**2,574.26 CR

FIGURE 31. ASTERISK PROTECTION

The control word is written with the asterisk at the left of the zero suppression code.

Forward Scan:

1. The normal editing process proceeds until the asterisk is sensed.
2. The asterisk is replaced (in the output field) by the corresponding digit from the A field.
3. The editing process continues normally until the B field word mark is sensed and removed.

Reverse Scan:

1. Zeros, blanks, and punctuation to the left of the first significant digit are replaced by asterisks.
2. The word mark (set during the forward scan) signals the end of editing. It is erased, and the operation is stopped.

Floating Dollar Sign

This feature causes the insertion of a dollar sign in the position at the left of the first significant digit in an amount field (Figure 32).

EXAMPLE:	
A field	00257426
Control word (B field)	bbb,b$0.bb
First forward scan	002,574.26
Reverse scan	bb2,574.26
Second forward scan	$2,574.26
Results of edit	$2,574.26

FIGURE 32. FLOATING DOLLAR SIGN

The control word is written with the "$" at the left of the zero suppression code.

Three scans are necessary to complete this editing operation.

First Forward Scan:

1. The editing proceeds until the "$" is sensed.
2. The "$" is replaced (in the output field) by the corresponding digit from the A field.
3. Editing continues until the B field word mark is sensed and removed.

Reverse Scan:

1. Zeros, and punctuation to the left of the first significant digit are replaced by blanks.
2. The reverse scan continues until the word mark (set during the first forward scan) signals the start of the second forward scan.

Second Forward Scan:

1. The word mark is erased and the scan continues until the first blank position is sensed. This blank position is replaced by "$," and the operation stops.

Sign Control Left

CR or − symbols can be placed at the left of a negative field, if the sign control left feature is used (Figure 33).

EXAMPLE:	
A field	00378940
Control word (B field)	CR&bbb,bb0.bb
Forward scan	CRb003,789.40
Reverse scan	CRbbb3,789.40
Results of edit	CR 3,789.40

FIGURE 33. SIGN CONTROL LEFT

The control word is written with the CR or – symbols in the high-order position.

Forward Scan:

1. The scan proceeds until the zero suppression character in the control field is sensed.
2. The corresponding character from the A field is placed in this position of the output field.
3. A word mark is automatically inserted in this position in the output field.
4. Editing continues and the CR or – symbols are undisturbed in their corresponding positions in the output field, only if the sign of the A field is minus. If the sign is plus, they are blanked.

Reverse Scan:

1. Zeros and punctuation are replaced by blanks in the output field. The scan continues until the automatically-set word mark is sensed.
2. This word mark is erased and the operation ends.

Decimal Control

This feature insures that decimal points print only when there are significant digits in the A field (Figure 34).

Two scans are sufficient to complete this editing operation *unless* the field contains no significant digits. Then three scans are required.

First Forward Scan:

1. When the zero suppression code (0) is sensed during editing, this position is replaced by the corresponding digit from the A field.
2. A word mark is set automatically in this position in the B (output) field.
3. Editing continues normally until the B field word mark is sensed and removed.

EXAMPLES:

1.	A field	00000
	Control word (B field)	bbb.b0
	First forward scan	000.00
	Reverse scan	bbb.00
	Second forward scan	bbb
	Results of edit	(Blank Field)
2.	A field	29437
	Control word (B field)	bbb.b0
	First forward scan	294.37
	Reverse scan	294.37
	Result of edit	294.37
3.	A field	00001
	Control word (B field)	bbb.b0
	First forward scan	000.01
	Reverse scan	bbb.01
	Results of edit	.01

FIGURE 34. DECIMAL CONTROL

Reverse Scan:

1. Zeros and punctuation are replaced by blanks in the output field until the decimal point is sensed.
2. The decimal point and the digits at its right are unaltered. The automatically-set word mark is erased. If there are no significant digits in the field, the second forward scan is initiated. Otherwise, the edit operation stops.

Second Forward Scan:

1. The zeros at the right of the decimal point and the decimal point itself are replaced by blanks.
2. The operation stops at the decimal column.

Operating Features

The IBM 1401 Data Processing System is equipped with operating features that give complete operator control for setting up and checking machine operation.

Console Keys, Lights and Switches

POWER ON. Controls the main power supply for the entire system. Pressing it causes POWER ON key to light.

POWER OFF. Turns off the main power supply.

START. This key (Figure 35) is used to initate or resume machine operation after a stop: manual, programmed or automatic. Similar keys are found on each of the other units in the system. Operation of this key is conditioned by the setting of the mode switch.

a. During a normal *run* mode, the system can be started by pressing the start key on any of the units.

b. During a *single cycle* process mode, any of the start keys can cause the system to advance through the program, except on an input-output

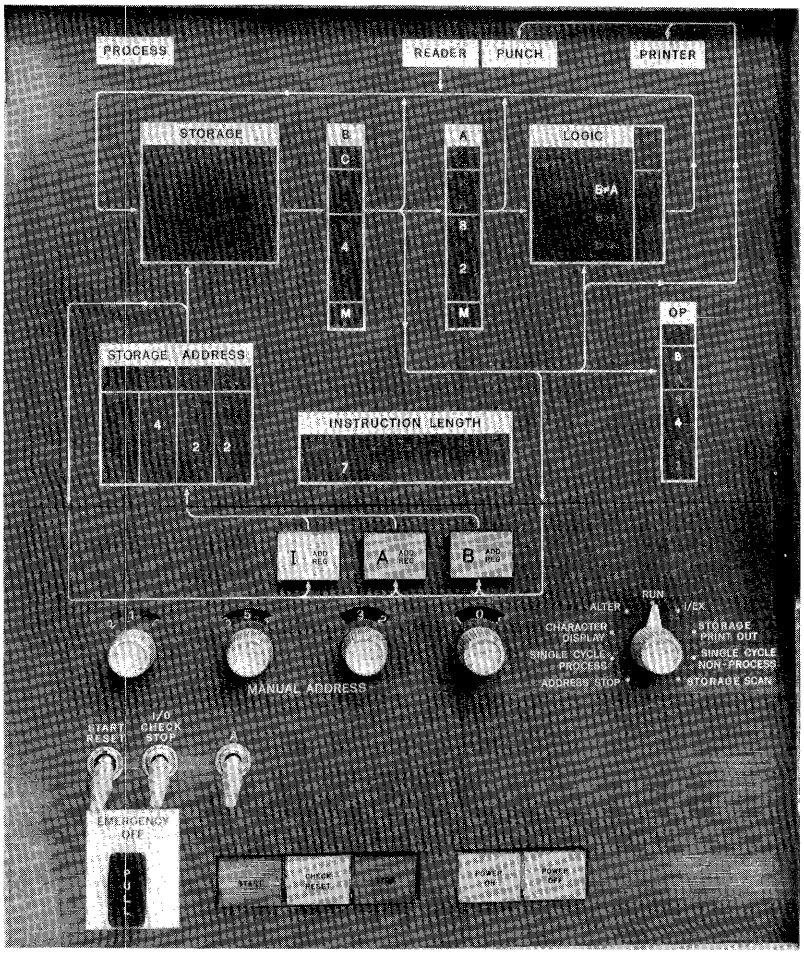


FIGURE 35. CONSOLE

execution cycle. The start key at the input-output unit must be pressed for this operation.

c. To restart following an error indication, the *check reset* key must be pressed prior to the operation of the start key.

d. Following a card jam or misfeed in either the reader or the punch, the cards in the associated feed must be run out by means of the *non-process-runout* key for that feed, and its hopper must be reloaded before the start key is pressed.

START RESET. This switch is used to reset the system (except for the data in storage) so that the operator can restart the operation.

STOP. This is a lighted key, and is used to stop processing in the system. It is not effective until the instruction being executed is completed. Similar stop keys (without lights) are provided on each of the other units within the system.

EMERGENCY OFF. This is a *pull* switch, located on the console. In an emergency, pulling this switch disconnects all the power to the entire system. This switch should be manually reset by a customer engineer before power is restored to the system.

CHECK RESET. An error detected by the checking circuits causes this key to light. It must be pressed following a 1401 Processing Unit error, and the system is restarted by pressing the start key.

Checking Lights

Four lights are provided at the top of the console panel, representing the Processing Unit, Reader, Punch, and Printer. When the machine is operating normally, these lights appear as white areas with black lettering. When the machine stops, requiring operator attendance at one of the four units, the appropriate light glows red, indicating an error. The light is extinguished when proper action is performed by the operator.

STORAGE. The storage light is red when an error at the input to storage is detected by a parity check.

B-LIGHT. The B-light comes on when a B register parity check error occurs. The lights underneath display the BCD coding check-bit status, and the word mark status of the character in the B register.

A-LIGHT. The A-light comes on when an A register parity check error occurs. The lights below indicate the coded character, check-bit status, and word mark status of the character in the A register.

Logic Block Lights

O-FLO. Lights when an overflow condition exists.

B $\neq$ A. Is on when an unequal-compare condition exists after a compare instruction. Additional lights are provided for high-low-equal compare when this optional feature is included in the system.

BIT DISPLAY. Shows the bit configuration of the sum of the characters being processed in an arithmetic operation.

Register Lights

OP REGISTER. The *Op* light is red when an incorrect operation code exists in the OP register, or if the code is incorrectly interpreted. The lights below indicate the coded character and the check-bit status of the character in the OP register.

INSTRUCTION LENGTH LIGHTS. Indicate the number of characters in the instruction.

STORAGE-ADDRESS LIGHT. Red when an address register parity check occurs. The lights below, displaying the address, can be checked for the error condition.

STORAGE ADDRESS DISPLAY. A group of storage address lights display the storage address (in binary-coded-decimal form) contained in the address register indicated by one of three key-lights:

I ADDRESS REGISTER. Glows when the I address is in the storage address display.

A ADDRESS REGISTER. Glows when the A address is in the display.

B ADDRESS REGISTER. Glows when the B address is displayed.

Stopping the machine and holding down one of these keys causes the contents of the associated register to be displayed in the storage-address lights.

I/O Check Stop Switch

When in the ON position (up), the machine stops at completion of an I/O operation if an error occurs during that operation. In the OFF position (down), the machine does not stop if it detects a hole count check in the Card Reader or Card Punch, a validity for the Card Reader, or a Print Check. With the switch in the OFF position, error detection must be accomplished by programming.

Manual Address Switches

The four dial switches labeled *Manual Address* are used to select the address to be entered in the storage-address register. These work in conjunction with the address register key-lights and the storage-address display lights.

For example, set the contents of the A address register to 1200.

1. Set the mode switch to ALTER.
2. Set the manual address switches to 1200.
3. Press the A address register key.
4. Press the start key.

The storage-display lights then show the bit configurations for this address (1200).

The manual address switches are also used to select a storage location for a display or alteration, without disturbing the contents of the address registers.

Sense Switches

Seven sense switches can be included in the 1401 Processing Unit. The manual toggle switches that control them are located on the console. Switch A is used to control last card operations by making the TEST AND BRANCH SENSE SWITCH ON instruction effective only when the last card in the reader has passed the second reading brushes. Switch A is standard in all systems except Model D. Six additional sense switches (B, C, D, E, F, and G) are optional features.

The B (I) d TEST AND BRANCH SENSE SWITCH ON instruction can be used to interrogate the setting of the switch specified by the d-character, at any time during processing, and causes a branch to the (I) address if the switch is ON.

Mode Switch

The nine *modes* of machine operation are selected by the *Mode Switch*:

1. RUN. When the mode switch is set to RUN, the system is under control of the stored program.
2. I/EX (INSTRUCTION/EXECUTION). When the mode switch is set to I/EX, the first time the start key is pressed, the machine reads one complete instruction from storage and stops. This is called the *instruction phase.*

   The next time the start key is pressed, the machine executes that instruction. This is called the *execution phase.*

   Subsequent pressing of the start key results in alternate instruction and execution phases.
3. SINGLE-CYCLE PROCESS. Each time the start key is pressed, one .012 millisecond storage cycle is taken when the machine is in the *single-cycle process mode.* Console indicating lights display the contents of the OP, I Address, A Address, B Address, A and B registers, and the logic unit. (See Figure 25 and Figure 26.)
4. SINGLE CYCLE-NON PROCESS. This is similar to the single-cycle-process mode, except that no data enters storage from the A register or the logic unit. Data always enters storage from the B register only. This mode permits observing the results of arithmetic operations, one character at a time, in the logic display, without destroying the original B field data.
5. CHARACTER DISPLAY. When the machine is operating in this mode, the start key is pressed to cause the character at the address selected by the manual-address switches to be displayed in the B register.
6. STORAGE PRINT OUT. This mode of operation permits any 100-character block of storage to be printed. The hundreds and thousands manual address switches are used to select the desired block of storage.

   Example: 12xx is set in the manual address switches and the start key is pressed. The 100 characters in the selected block 1201-1300 are printed automatically in print positions 1 through 100. Another automatic print cycle causes the word marks for that block to be indicated by printing 1's in their corresponding print positions on the second line. This feature is used to great advantage in program testing, because the contents of a block in core storage is printed and can be easily examined by the programmer. Thus, this feature serves to increase both processing and programming efficiency.
7. ALTER. The operator can manually change the contents of any address register or storage location if the mode switch is set to ALTER. For example, to change the contents of address registers:

   set the manual address switches at the desired location;
   press the appropriate address register key-light;
   press the START key;
   the selected address register is set with the new address.

   To change the contents of a storage location:

   set the manual address switches to the desired location;

select the bit-structure of the character to be entered, by setting the eight BIT-switches located on the auxiliary console;
press the ENTER key (also on the auxiliary console).

8. STORAGE SCAN. When the mode switch is set to STORAGE SCAN, pressing the start key causes the 1401 to start reading out of storage beginning at the address set in the manual-address switches. If an error condition is detected that had been previously set by an input-output device the machine stops, and the check light with the corresponding unit is turned on; and the location of the card column or print position in error is shown in the storage address display unit. The B-register contains the storage position in which the error was detected, the actual location in storage can be corrected by using the BIT-switches and ENTER key as described under the ALTER mode.

   After the error condition is corrected, the MODE switch is again set to STORAGE SCAN and the START key is pressed to cause a read out of storage starting from the address set in the manual address switches. This mode is used as a service aid to insure that all positions of storage are correct.

9. ADDRESS STOP. When the mode switch is set to ADDRESS STOP, pressing the start key starts the program and the machine stops at the address selected by the manual address switches.

Auxiliary Console

The auxiliary console panel (Figure 36) is located below the main console of the 1401 processing unit. Its purpose is to provide additional operator control of the system.

Keys Lights and Switches on the Auxiliary Console

BIT SWITCHES. Eight bit switches are used to alter characters in storage. These switches are used in conjunction with the alter mode as explained in the mode switch description.

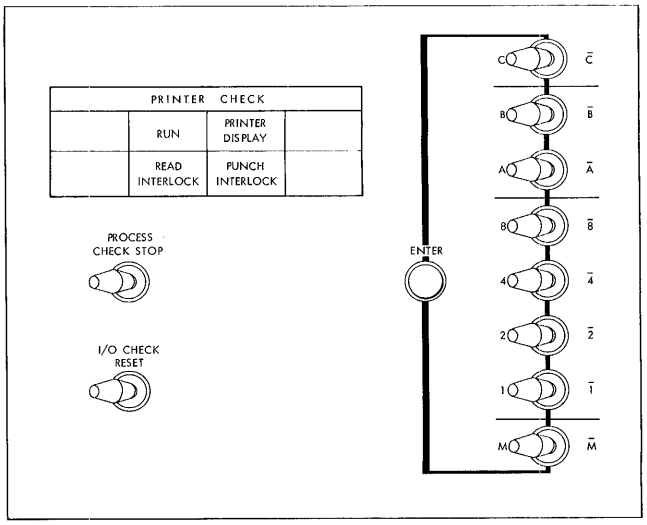


FIGURE 36. AUXILIARY CONSOLE

ENTER. This key is used to enter the characters selected by the BIT-SWITCHES into storage, when the mode switch is set to ALTER.

PROCESS CHECK STOP SWITCH. This is normally ON to cause the machine to stop automatically when a process check occurs. If the switch is in the OFF position, the machine does not stop on error conditions, except for op register and address register checks, and input-output checks.

I/O CHECK RESET SWITCH. This switch resets error conditions sensed on the read punch unit and permits the start key to be effective. It is primarily used by Customer Engineering.

READ INTERLOCK. When this light is on the reader is in a ready condition, when it is off the reader is interlocked until the print operation is completed.

PUNCH INTERLOCK. When this light is on the punch is in a ready condition, when it is off the punch is interlocked until the print operation is completed.

PRINTER DISPLAY. This light is on when a print operation is being executed.

RUN. When this light is on the printer is in a ready condition, when it is off the printer is interlocked until the print operation is completed.

IBM 1402 Card Read Punch Operating Keys, Lights, and Switches

START. Causes the machine to start, and feed two cards into the read feed. If the punch switch is ON, two cards are also fed into the punch unit (Figure 37).

STOP. Used to stop the system. If a program step is in process, it is completed before the stop occurs.

NON-PROCESS RUNOUT READ. Pressed to clear the read feed. The last two cards in the normal stacker have not been processed.

NON-PROCESS RUNOUT PUNCH. Causes the punch feed to be cleared of cards. The last two cards in the normal stacker have not been processed.

LOAD. Used to start loading instruction cards. Pressing the load key causes the read feed to operate until a card has passed the second read station. The I Address Register is reset to 001, and a word mark is set in address 001. All other word marks in addresses 002 through 080 are removed.

When the card is read at second read, the program starts and executes the instruction that is punched in the first columns of the card.

Continued operation is completely under control of any program in that card or succeeding cards, as conditioned by the first instruction in the first card. When the PUNCH SWITCH is ON, pressing the load key also causes the punch feed to operate until a card reaches the punch station.

CHECK RESET. Must be pressed to reset any error indication by a punch, read, or validity check, before the start key can become effective.

PUNCH SWITCH. Controls the *punch* section of the machine. When this switch is OFF the punch is inoperative. When it is ON, the machine runs automatically if all the interlock circuits in the punch side are satisfied.

POWER ON LIGHT. When power is supplied to the read-punch unit, the *power on* light is ON.

READER STOP. A condition (empty hopper, feed failure, or a card jam) causes the machine to stop and the reader stop light to come on.

PUNCH STOP. A condition (empty hopper, feed failure, or a card jam) causes the machine to stop and the punch stop light to come on.

VALIDITY. This light is ON if an invalid character is detected during a read operation.

READ CHECK. This light comes on if a hole count error is detected during card reading. If the count from the first and second reading brushes for a given card do not agree, an error is indicated by the *read check* light.

PUNCH CHECK. This light is ON if a hole count error is detected in the punch unit. If the hole counts are unequal, an error is indicated by the *punch check* light.

STACKER. If any of the five stackers becomes full, the machine stops, and this light signals the operator.

FUSE. When a fuse in the Card Read Punch burns out, this light signals the condition.

IBM 1403 Printer Operating Keys, Lights and Switches

Printer Controls (Figure 38)

START. Starts the machine.

STOP. Stops the machine at completion of the instruction in process.

END OF FORM. This light shows an end-of-form indication and the machine stops.

FORMS CHECK. This light indicates paper feed trouble in the forms tractor or the carriage stop has been used. This light must be cleared by the check reset key before the print start is effective.

CARRIAGE STOP. Pressing this button permits immediate stopping of the carriage operation and turns on the Forms Check light.

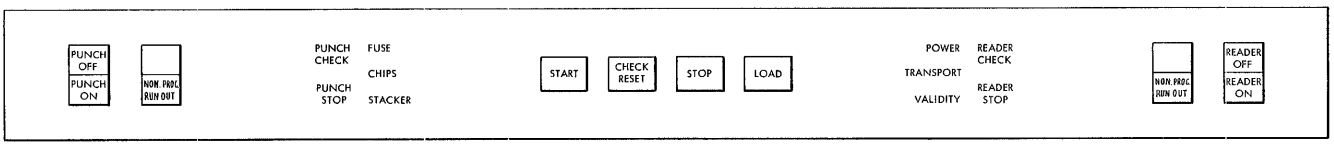


FIGURE 37. 1402 CARD READ PUNCH KEYS, LIGHTS & SWITCHES

READY. This light comes on when the printer is in condition to print, and all error detecting devices are reset.

PRINT CHECK. This light indicates a print error.

SYNC CHECK. This light comes on to show that the chain was not in synchronism, at all times, with the compare counter for the printer. The timing is automatically corrected. The light is extinguished by pressing *check reset* key.

Carriage Controls (Figure 38)

RESTORE KEY. Causes the carriage to position at Channel 1 *(home position)*. If the carriage feed clutch is disengaged, the form does not move. If it is engaged, the form moves in synchronization with the control tape.

SPACE KEY. Causes the form to advance one space each time it is pressed.

SINGLE CYCLE KEY. Initiates the operation of the printer for one print cycle on each pressing of the key.

Manual Control (Figure 39)

FEED CLUTCH. Controls the carriage-tape drive and form-feeding mechanism. If it is set to neutral, automatic form-feeding can not take place. It is also used to select six- or eight-lines-to-the-inch spacing.

FEED KNOB. Used to position the form vertically, and can be used only when the *feed clutch* is disengaged.

VERNIER KNOB (vertical). Used for fine spacing adjustment of forms at the print line. Carriage tape is not affected by this knob.

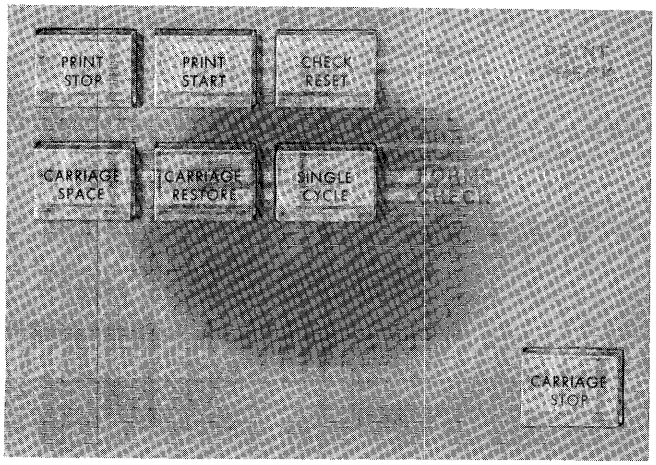


FIGURE 38. 1403 PRINTER KEYS, LIGHTS & SWITCHES

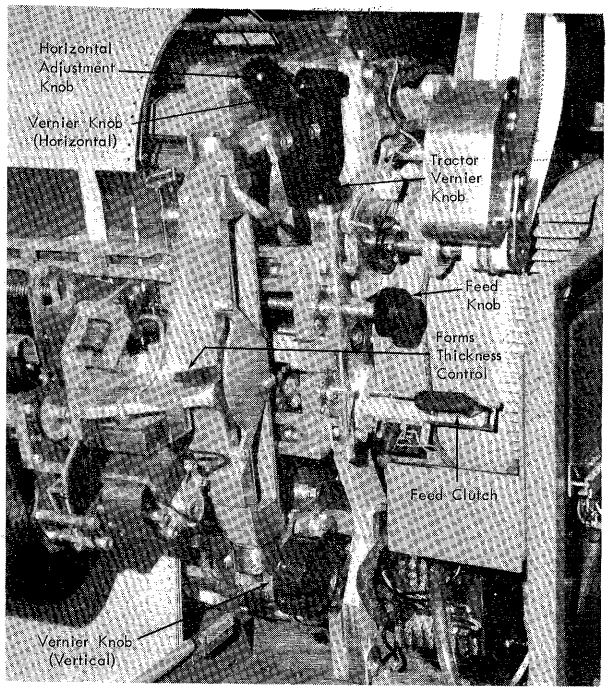


FIGURE 39. PRINTER MANUAL CONTROLS

HORIZONTAL ADJUSTMENT. The printing mechanism is positioned, horizontally, by using this device.

VERNIER KNOB (horizontal). Turn this knob to obtain fine horizontal positioning.

FORM THICKNESS CONTROL. Sets the proper clearance between the hammer and the chain to obtain optimum printing quality on multipart forms.

PRINT UNIT RELEASE LEVER. Permits access to form transport area (Figure 40).

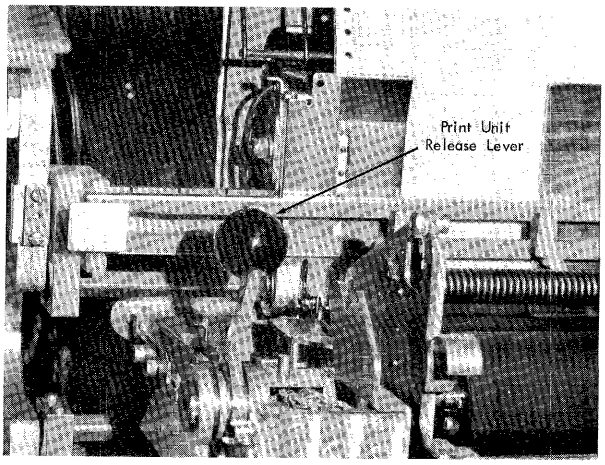


FIGURE 40. PRINT UNIT RELEASE LEVER

IBM 1401 Magnetic Tape System

The IBM 1401 Data Processing System can be expanded to meet individual requirements in many areas of data processing. A 1401 System can be tailored to individual needs because it can consist of varying numbers and types of units (Figures 41, 42, and 43).

The Tape System can be used alone as a complete data processing system, or as auxiliary equipment for intermediate and large-scale systems. It provides economical off-line tape editing and printing.

As many as six magnetic tape units can be connected to the IBM 1401 Data Processing System, providing low-cost, full-scale, punched card, and magnetic tape input and output operations.

The logic, arithmetic, editing, and tape instructions of the Tape System can be used to perform the primary functions:

Magnetic tape to printer
Punched cards to magnetic tape
Magnetic tape to punched cards
Punched cards to printer
Tape Sorting-Merging

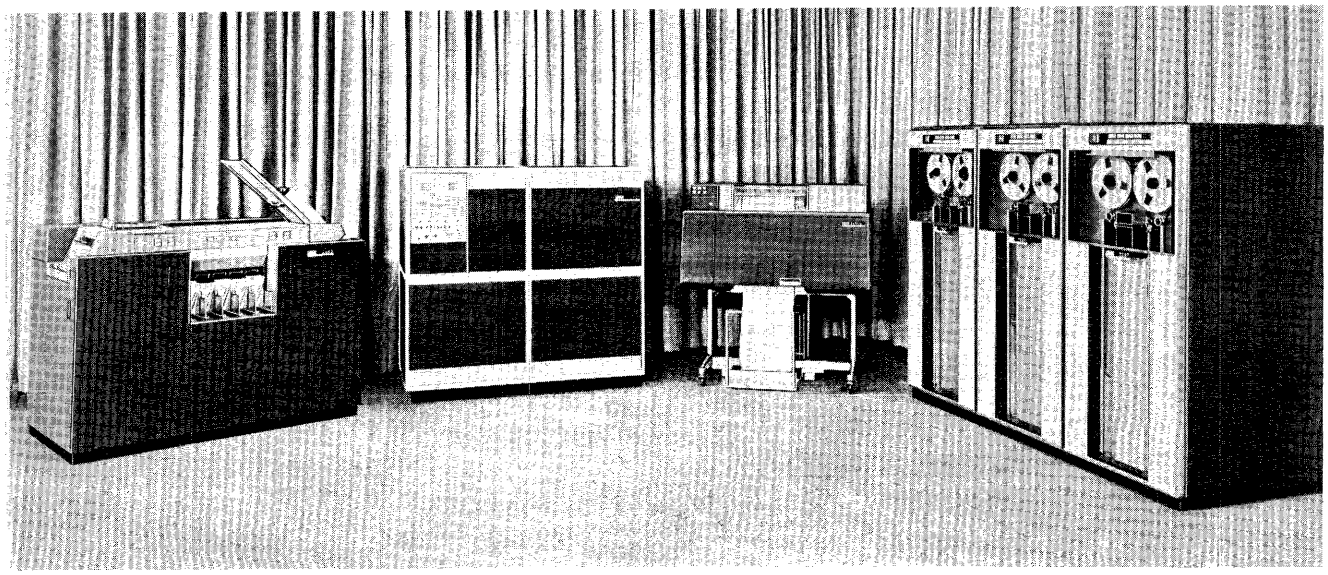


FIGURE 41. 1401 TAPE SYSTEM, MODEL C

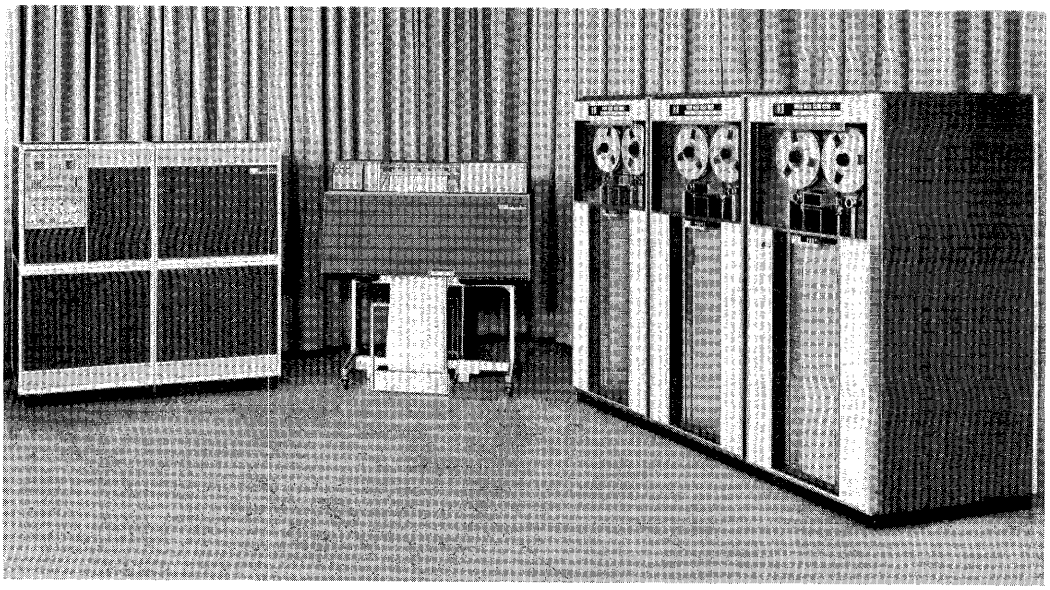


FIGURE 42. 1401 TAPE SYSTEM, MODEL D

	Model C—1401 Processing Unit with 1400 Character Core Storage	Model D—1401 Processing Unit with 1400 Character Core Storage
COMPONENTS AVAILABLE:	Model 1 Card Read Punch Model 2 Printer 729 Model II or IV Tape Units	Model 2 Printer 729 Model II or IV Tape Units
STANDARD FEATURES:	Expanded Print Edit Read Punch Release Sense Switches Additional Print Control Dual Speed Carriage Control	Expanded Print Edit Additional Print Control Sense Switches Dual Speed Carriage Control
OPTIONAL FEATURES:	2,000 or 4,000 Character Core Storage Multiply Divide High Low Equal Compare Column Binary Print Storage	2,000 or 4,000 Character Core Storage High Low Equal Compare Print Storage

FIGURE 43. TAPE SYSTEM COMPONENTS

Data Flow

Data can be entered into the 1401 systems by means of punched cards, magnetic tape, or both. Information is read into the system, rearranged, calculated and edited by the stored program. Output can be in the form of punched cards, magnetic tape, or printed reports. Model D is not equipped with a card read-punch (Figure 44).

All data passes through 1401 core storage, where a series of validity checks insure accuracy and reliability.

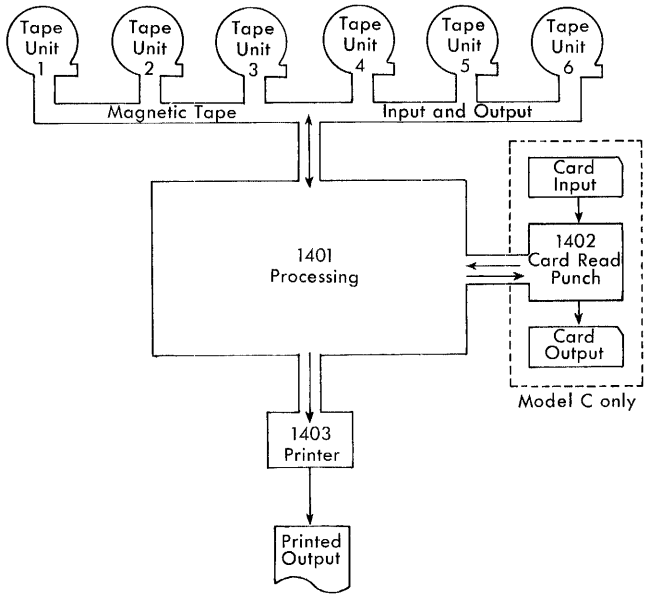


FIGURE 44. DATA FLOW SCHEMATIC 1401 TAPE SYSTEM

Magnetic Tape

An important feature in economical processing of business data is compact storage. A magnetic tape reel (10½ inches in diameter) contains 2400 feet — sufficient tape to record as many as 14,000,000 characters. Tape reels can be easily stored or transported from one installation to another. In addition, magnetic tape records have gained wide acceptance as legal documents.

Magnetic Tape Characteristics

The magnetic tape recording code used with the IBM 1401, is the same *binary-coded-decimal* code used with other IBM Data Processing Systems. This compatability permits interchanging tapes between installations that employ different IBM systems. The tape itself is a ribbon, ½-inch wide, coated with a magnetic oxide material.

Data is recorded in a seven-bit code, in seven parallel channels along the tape. Tape characters and their corresponding codes are shown in Figure 45.

Records are separated from each other by approximately ¾ inch of blank (unrecorded) tape, called an *inter-record gap*.

Each tape character is composed of an even number of magnetic bits. A check bit (labeled C in Figure 46) is written if the number of bits in the other six positions is odd. An even-parity check on each character insures accuracy for tape-read and tape-write operations.

In addition to this vertical parity check, a horizontal check (HC in Figure 46) is made on each record. The bits in each horizontal row are automatically counted when the record is written, and a bit (similar in function to the vertical check bit) is written at the end of each odd-count row. The vertical combination of these horizontal-check bits makes up the horizontal-check

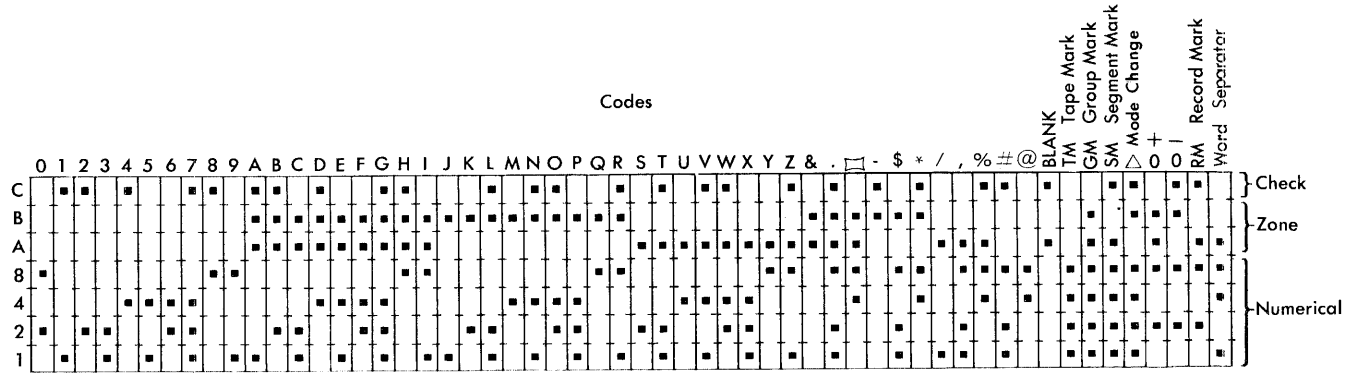


FIGURE 45. MAGNETIC TAPE 7-BIT CODING

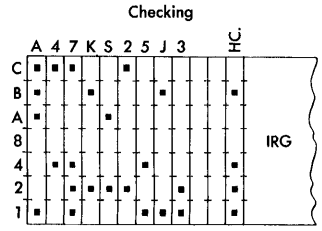


FIGURE 46. VERTICAL AND HORIZONTAL CHECK

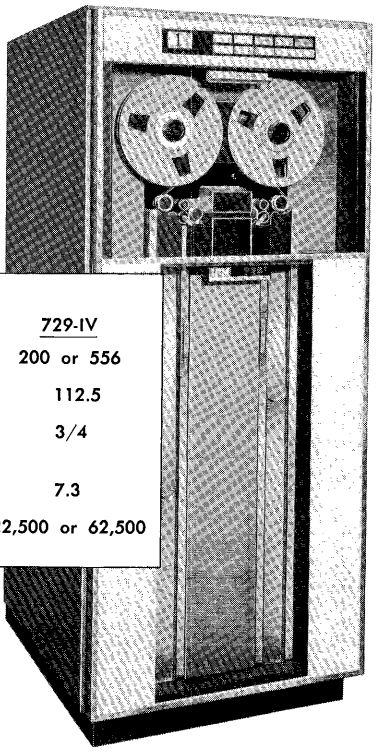
character. Thus, the coding of this character can change from record to record. When the tape is read, the same automatic count is made, but now each row in the complete record should have an even number of bits, or an error exists. The horizontal-check character is used for checking only, and is never read into 1401 core storage.

Odd bit redundancy tapes can be processed by the 1401.

Tape Units

Two models of the IBM 729 Magnetic Tape Unit (Model II and Model IV) are available for use with the IBM 1401. The Tape System can accommodate as many as six 729 tape units, which are attached to the Tape Adapter. All the units used with one 1401 system must be of the same model, and must use the same tape density.

The significant operating characteristics of the 729 II and IV tape units are outlined in Figure 47. Higher density tapes provide significant storage advantage in that fewer reels are required for a given volume of data.



	729-II	729-IV
Density, characters per inch	200 or 556	200 or 556
Tape Speed, inches per second	75	112.5
Inter-record gap size, inches	3/4	3/4
Start/Stop Time, Read/Write operation; milliseconds	10.8	7.3
Character Rate, characters per second	15,000 or 41,667	22,500 or 62,500

FIGURE 47. OPERATING CHARACTERISTICS OF IBM 729 TAPE UNITS

Tape Checking

The 729 tape units achieve increased reliability through two new features: the *two-gap head,* and *dual-level sensing.* The first of these, the two-gap head, makes it possible to verify automatically the validity of recorded information at the time it is written. The relative position of the read and write gaps (Figure 48) is such that a character recorded by the write gap passes the corresponding read gap approximately four milliseconds later. Thus, as each character of a record is written, it is read and a parity check applied.

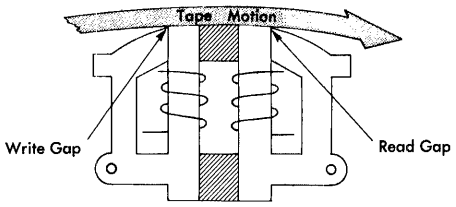


FIGURE 48. READ & WRITE GAPS ON A TWO-GAP HEAD

If an error is detected, the stored program receives a signal, and corrective action can be taken. With the two-gap head, a parity check is detected when the character is written.

The ability of the two-gap head to read tape in both reading and writing operations makes it possible to check these operations by dual-level sensing. The read head reads the tape at two levels of pulse strength, high and low.

There is a 7-position high register and a 7-position low register (one position for each channel). These registers accept the pulses from the tape at a high sensitivity level and low sensitivity level, respectively, as each character is read. The registers function in a slightly different manner for reading than for writing.

In a tape-read operation, the high sensitivity level register is checked for even parity. If there is an odd number of bits, the contents of the low sensitivity level register is sent to the read-write register.

The contents of the read-write register is sent to core storage. If a validity check is detected at the read-write register a validity check indicator is set. This indicator can be interrogated by use of the "L" modifier of the TEST AND BRANCH instruction.

Thus, a bit that results in a weak signal, but is valid, can be read from tape. If the character is still invalid, a validity check signal is given.

In checking tape-write operations, the unit becomes harder to satisfy by automatically making the high register less sensitive than for tape-read. (It still has a higher sensitivity level than the low register, however.) Each tape character written is read back, and must be

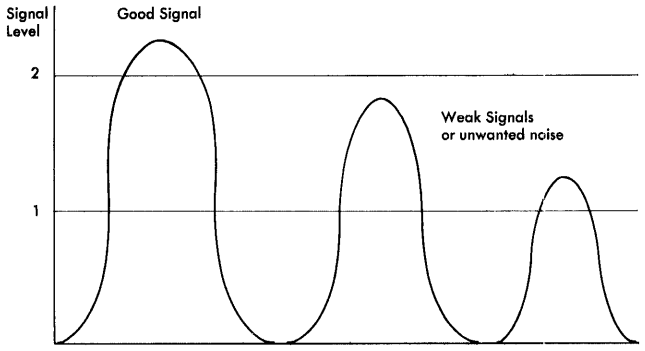


FIGURE 49. RELATIVE SENSITIVITY FOR DUAL LEVEL SENSING

valid in both registers. The contents of the low register are validity checked, and then are matched, bit for bit, with the contents of the high register.

If the validity check in the low register detects an odd number of bits, or if the bit-by-bit match between registers is unequal, a validity check signal is given.

Figure 49 shows the sensitivity levels, and the relative strength of pulses that are acceptable or not acceptable in read and in write conditions. Note that high sensitivity means that an impulse of comparatively low strength (voltage difference) is acceptable. Low sensitivity means that signals below a certain level or strength are not detected.

If a tape error is suspected, the tape unit (through programming) can be back-spaced and the record reread. If the error persists, the operator can intervene, or the program can branch to an error routine.

Dust or damage to the magnetic tape is the most frequent cause of errors detected during write operations. Since such imperfections are usually isolated, the 1401 has been provided with a command that causes the tape to *space forward* approximately 8 inches when the next write operation is initiated, in order to skip the defective section. As the tape is passed, this short length is erased so that extraneous data are not sensed when the tape is read. After the skip is completed, the tape-write operation continues.

Another feature for file protection is a plastic ring (Figure 50) that fits into a groove in the tape reel. The tape can be read with or without this file protection ring in place, but no writing can be done without it.

The file protection ring should be removed from a tape reel when writing is completed, thus protecting tape records from any accidental writing.

Tape Instructions

OP — The operation code

(A) — %xx always appears in the (A) portion of a 1401 regular tape instruction.

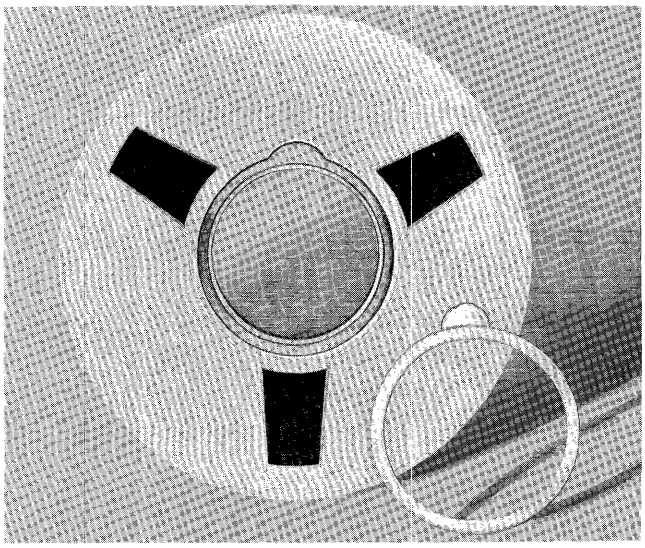


FIGURE 50. FILE PROTECTION DEVICE ON TAPE REEL

The "%" sign signals that a tape unit is to be selected. The second character can be varied to specify a particular type of operation. The third digit specifies the particular tape unit involved.

EXAMPLE: %U1 selects tape unit 1

(B) — is the location in 1401 core storage of the high-order position of a tape record.

d—The actual operation is represented in this position.

M (A) (B) d MOVE MAGNETIC TAPE

This instruction starts the tape unit specified by the (A) address. Word marks are not affected by a move instruction. Then the d-portion of the instruction is interpreted.

TAPE READ (R). A tape-read operation is terminated when an inter-record-gap is sensed. A *group mark* (code CBA 84231) is inserted in 1401 core storage to indicate the end of a tape record.

EXAMPLE: M(%U2) (419)R. Read the record from tape unit 2 to 1401 core storage in a TAPE READ operation. The high-order tape-record character is moved to location 419, the next character is moved to location 420, etc., until transmission is stopped by an inter-record-gap in the tape record, or a group mark in 1401 core storage.

TAPE WRITE (W). Transmission of data from 1401 core storage to a tape is stopped when a group mark is sensed. The (B) address is the high-order position of the record (in core storage) that is to be written on tape.

EXAMPLE: M(%U3) (525)W. In a tape-write operation, transfers the contents of core storage to tape unit 3, starting at location 525 and ending at the first group mark sensed.

L (A) (B) d LOAD MAGNETIC TAPE

The load instructions are basically the same as move instructions. Where the (A) address specifies the tape unit, (B) is the 1401 core storage address of the high-order position of the tape record, and the d-character is R for tape-read, and W for tape-write.

However, the L operation code affects word mark identification in core storage:

TAPE WRITE. A word mark associated with any position in storage causes a *word-separator character* (A 841) to be written automatically on tape, one character ahead of that which contained the word mark. Thus, word marks are translated to word-separator characters for tape storage.

EXAMPLE:

1401 Core Storage Locations	A	B	C	
1401 Core Storage Code	C82	41W	4	
1401 Meaning	0	5	4	
Tape Positions	A	B	C	D
Tape Code	82	A841	41	C4

TAPE READ. Word-separator characters are translated to word marks in tape-read operations. A word-separator character read from tape causes a word mark to be associated with the next tape character, when it is transferred to 1401 core storage.

EXAMPLE:

Tape Positions	A	B	C	D
Tape Code	82	A841	41	C4
1401 Core Storage Locations	A	B	C	
1401 Meaning	0	5	4	
1401 Core Storage Code	C82	41W	4	

Load instructions must be used when word marks are needed for identification in tape storage. If tape is written by a LOAD instruction, it must be read back by a LOAD instruction for proper translation between the tape and 1401 core storage.

U (A) *d* UNIT CONTROL

This instruction is used to control other tape operations as specified by the d-character function:

d character	*operation*
B	Backspace Tape
E	Erase Forward
M	Write Tape Mark
R	Rewind Tape

The (A) address specifies the tape unit selected.

U (A) B BACKSPACE TAPE

This instruction causes the specified tape unit to backspace over one tape record. The backspace operastops when an inter-record-gap is sensed.

EXAMPLE: U(%U4)B—Tape unit 4 backspaces until an IRG is sensed.

U (A) E ERASE FORWARD

This operation causes the specified tape unit to space forward, and erase approximately 8 inches of tape, to bypass the defective tape areas. The skip does not actually occur until the next tape-write operation is given.

EXAMPLE: U(%U2)E—Tape unit 2 erases approximately 8 inches of tape when the next M (%U2) (B) W; or L (%U2) (B) W (tape-write) instruction is ordered.

U (A) M WRITE TAPE MARK

A special tape character (8421) is recorded following the last record on a tape, to indicate an *end-of-reel* condition. When the tape mark character is read back from a tape, the end-of-reel indicator is turned on. This signals the 1401 program that the end of the utilized tape has been reached.

EXAMPLE: U(%U1)M. A tape mark is inserted after the last tape record that was written on tape 1.

U (A) R REWIND TAPE

This instruction is usually given subsequent to an end-of-reel condition, and causes the selected tape unit to rewind its tape. When the operation is initiated, the unit specified is effectively disconnected from the system. Rewind time is approximately 1.2 minutes per 2,400-foot reel for the 729 II, and .9 minutes for the 729 IV. The next instruction following a REWIND TAPE instruction is normally a STOP, so that the operator can replace the reel, and restore the tape unit to a ready status at the completion of the rewind operation.

EXAMPLE: U(%U3)R. The tape in tape unit 3 begins to rewind.

Tape Test Instructions

B (I) D TEST AND BRANCH

An instruction for testing tape conditions is provided. The d-character specifies the type of test, and the (I) portion is the location of the next stored-program instruction if the test is successful. If the tested condition is not present the program continues in normal sequence. The "K" and "L" modifiers reset the condition tested.

d-character	*condition*
K	End-of-reel
L	Tape error

B (I) K END-OF-REEL INDICATOR TEST

When a tape mark is read by the 1401, or the reflective spot sensed during a write operation, the end-of-reel indicator is turned on. This instruction tests the indicator and branches to location (I) if it is ON. If it is OFF, the program continues normally.

EXAMPLE: B(496)K. If there is an EOR condition, the program branches to core-storage location 496. If no EOR condition exists, the program continues in sequence.

B (I) L TAPE TRANSMISSION ERROR TEST

Whenever an error occurs in transmission between a tape unit and the 1401 during a tape-read or write operation, an error indicator is turned on in the 1401, and a tape-error light on the console glows red. The B (I) L instruction tests the error indicator, and branches to the location specified in the (I) address if it is ON. If no tape error occurred, the program continues in sequence.

EXAMPLE: B(521)L. The 1401 program branches to location 521 if a tape error has occurred. If there was no error in transmission, the normal program sequence is uninterrupted.

TESTING CONDITIONS

The end-of-reel and tape transmission error tests must be made immediately following a tape-read or write operation to insure correct operation. A tape operation on any tape unit resets the error indicator. The EOR indicator cannot be tested if another tape unit is selected.

Tape Sorting

The high-low-equal compare feature provides speed and flexibility in tape-sorting operations. A control number in storage can be used to determine the sequence of records that have been read from tape.

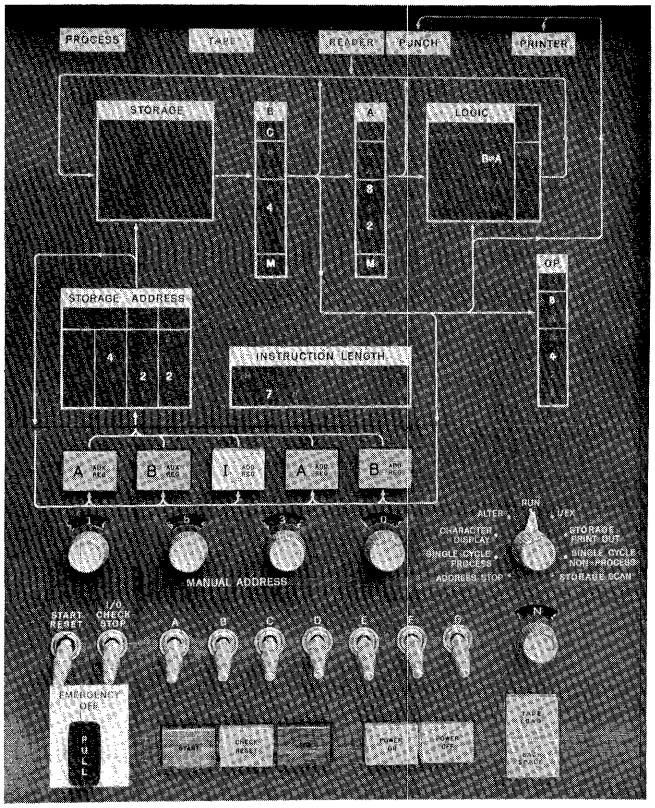


FIGURE 51. CONSOLE KEYS, LIGHTS AND SWITCHES FOR TAPE SYSTEM

B (I) d TEST HIGH LOW OR EQUAL COMPARE

The B (I) d instruction tests the result of the previous compare operation and branches to the location of the next instruction if the condition is satisfied:

d-character	*condition*
/	Unequal (B is not equal to A)
S	Equal (B is equal to A)
T	B is lower than A
U	B is higher than A

EXAMPLE:

C(495)(596) Compare the data at storage location 596 to the control number at 495.

B(797)U If the data at 596 is higher than the control number at 495, branch to location 797 for the next instruction.

Console Keys, Lights and Switches

TAPE SELECT. This rotary switch is set to the normal position (N) during automatic operation. The switch can be set to the number (1-6) that corresponds to any of the attached tape units, when manual operation is desired (Figure 51).

BACKSPACE. This key works in conjunction with the tape-select switch. When the switch is set to a specific tape unit, pressing this key causes the tape in the selected unit to back space over one group of records, until an inter-record gap is sensed.

TAPE. This light glows red if a tape error occurs during a read or write operation. It is turned off automatically when the error indicator is reset by a subsequent tape operation.

TAPE LOAD. When this key is pressed, tape unit 1 is automatically selected and tape data starts loading at address 001 and continues until an inter-record-gap is sensed.

A AND B AUXILIARY REGISTERS. Pressing either the A or the B Auxiliary Register key displays the contents of the particular register. The auxiliary registers are part of the multiply-divide optional feature.

Column Binary Device (Optional)

This feature provides compatibility of input-output information between the IBM 1401 Data Processing System and IBM Scientific Data Processing Systems such as the IBM 704, IBM 709, and IBM 7090.

With this feature, cards, and magnetic tapes which are binary coded, can be processed by the 1401, making use of its operational functions such as reading, writing and logic operations.

The reading of column binary cards is controlled by modified feed instructions. The IBM 1401 operation codes for this feature are:

1 C	READ COLUMN BINARY
1 (I) C	READ COLUMN BINARY AND BRANCH

The READ COLUMN BINARY operation code 1 C cannot be combined with any other code.

The card reader takes a feed cycle and reads the information into storage as a normal feed cycle. However, instead of taking only 80 storage cycles during each read scan, the reading requires 160 storage cycles for column binary coding.

During the reading of column binary information two different areas of storage are used. The read cycle is executed in two parts to permit using the two areas of storage. The card cycle time 9 through 4 uses the normal feed addresses 001 through 080 and new area addresses 501 through 580. The other portion of the card cycle time 3 through 12 uses addresses 001 through 080 and 401 through 480.

A validity check on this operation is not performed because all the characters are considered valid. At the completion of this instruction, a BCD coded image of the card is stored in addresses 001 through 080. The portion of the card that contains column binary information appears as *hash* in the corresponding addresses 001 through 080, and the portion of the card that contained alphameric characters is stored in BCD code in storage addresses 001 through 080. Storage address 401-480 and 501-580 contains the true coded card image. In these areas all alphameric characters appear as *hash* and all column binary information appears as illustrated in the following example:

	BCD	Punches in Card Column
	C	
	B	12
Storage	A	11
Address 401	8	0
	4	1
	2	2
	1	3
	C	
	B	4
Storage	A	5
Address 501	8	6
	4	7
	2	8
	1	9

The punching of column binary cards is controlled by modified punch instructions. The IBM 1401 operation codes for this feature are:

4 C	PUNCH COLUMN BINARY
4 (I) C	PUNCH COLUMN BINARY AND BRANCH

The punch instructions require controls similar to the read instructions. The storage addresses 401 through 480 are punched in the 12 through 3 positions of the card in columns 1 through 80. The storage addresses 501 through 580 are punched in the positions 4 through 9 in the card in columns 1 through 80. Because storage positions 001-080 are scanned for a columnar count check, 160 storage cycles are required for each scan.

To write the column binary information on tapes in the correct order, the data located in addresses 401-480 and 501-580 must be arranged in the following sequence:

Addresses 401, 501, 402, 502, 403, 503 — 472, 572.

To arrange this information in the proper order a modified MOVE instruction is used with the stipulation that a word mark must be located in the high-order position of storage, in the 400 area. This area contains the column binary information.

M (A) (B) A MOVE AND UNSCRAMBLE COLUMN BINARY

This code provides for moving and arranging data into the correct order and locations for tape output. The address at (B) can be any valid address. The (A) address is normally 572 or 580, depending on whether the card has 72 or 80 columns of binary information.

At the completion of this instruction, the information is arranged in the proper binary order for writing on tape.

When binary information is read into the 1401 from tape, it must be put into a coded card image. The modified MOVE instruction to do this is:

M (A) (B) B MOVE AND SCRAMBLE COLUMN BINARY

This code moves and arranges information read in from a tape unit, into coded-card image (binary form). The (A) address can be any valid address, the (B) address is normally 572 or 580 depending on the number of columns being processed.

A word mark must be located in the high-order position of the B field, normally 401, to end the operation. A word mark in the A field can also end the operation after the following B cycle. At the completion of this instruction, the information is in the coded card image form in addresses 401-480 and 501-580.

The column binary information must be put on magnetic tape with an odd redundancy mode. This operation is accomplished by using a modified tape instruction:

M (%B1) (B) W WRITE TAPE BINARY

Causes magnetic tapes to be written in binary form.

M (%B1) (B) R READ TAPE BINARY

Causes magnetic tapes, written in binary form, to be read into the 1401 system.

The only difference between this type of instruction and the basic read and write magnetic tape instructions is the B character located in the third position of the instruction. This B character causes the 1401 to read or write tape in an odd redundancy mode.

W (I) (B) d BIT TEST

The digit (d) can be any character or combination of bits that can exist in a character position in the 1401. If the single character at address (B) contains a bit that matches a bit in the (d) character the program branches to address (I). Otherwise the next instruction follows in sequence. Example: Address (B) contains G (A, B, 1, 2, and 4 bits) and digit (d) contains 9 (8 and 1 bits), a branch occurs because the 1 bits match.

This instruction can be chained so that a program branch takes place if any one of a number of characters contains the bits tested for:

W (I) (B) d W W W W

Program Loading Routine

This is a procedure for loading information into the IBM 1401 Data Processing System. It is not the only method that can be used, but it is typical of methods used by programmers.

This loading procedure pre-supposes use of an instruction card format as shown on the chart in Figure 52.

The rules to be followed in preparing each of the 6 types of instruction cards used for loading are:

RULE 1.

Card formats must follow those shown in the storage-layout chart. The first three cards are used to set word marks necessary for succeeding operations.

CARD 1. Does not need a word mark for location 001. The load key automatically sets the program to this location. No word mark is necessary for location 008 for the first card. The 1401 recognizes the end of a SET WORD MARK instruction when it has placed seven characters into the program registers. The card sets word marks in locations 008 and 012, initiates the reading of card 2, and branches to location 001.

CARD 2. Sets two word marks for locations 060 and 067, initiates the reading of card 3 and branches to location 001.

CARD 3. Sets two word marks, for locations 074 and 078. It causes card 4 to be read, and branches to location 060 for the next instruction.

INSTRUCTION CARD. A standard load card. The information contained in card columns 1-4, 8-11 and 60-80 is constant, and should be pre-punched. The data punched in card columns 5-7 identifies the location of the instruction (high-order position). The instruction to be loaded is punched starting in card-column 12, and may continue through card-column 19.

CONSTANT LOAD CARD. The length of the constant can vary from 1 character to 48 characters. The constant load card is a standard card and may be prepunched. The location (XXX) of variable data (units position) is in columns 5-7. The constant to be loaded is in card columns 12-59, and the number of characters (YYY) to be loaded is in columns 2-4.

NOTE: The information to be prepunched differs from that prepunched in the instruction load cards.

TRAILER CARD. Used to clear input area and to branch to the first program step.

RULE 2.

Pressing the LOAD key on the reader causes an instruction card to feed, places the contents of the card into locations 001 through 080, and automatically starts execution of the load program at location 001. This eliminates the need for manual setting of console dials in preparation for loading.

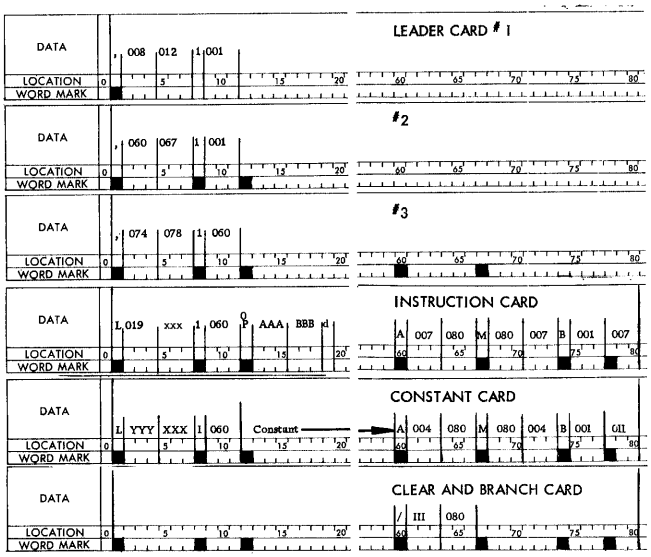


FIGURE 52. INSTRUCTION CARD STORAGE FORMAT

IBM 1401 PROGRAM CHART

FORM X24-6437-0 PRINTED IN U.S.A.

Program: Clear Storage

Programmer: Card 1 of 2 Date:

Step No.	Inst. Address	Instruction OP	A/I	B	d	Remarks	Effective No. of Characters Inst	Data	Total
	001	,	008	013		Set Word Marks in Read Area			
	008	N	000	0		"			
	013	,	020	027		"			
	020	,	031	035		"			
	027	,	039			"			
	031	,	043			"			
	035	,	050			"			
	039	,	054			"			
	043	,	058	074		"			
	050	,	077			"			
	054	1	001			Read and Branch			

IBM 1401 PROGRAM CHART

FORM X24-6437-0 PRINTED IN U.S.A.

Program: Clear Storage

Programmer: Card 2 of 2 Date:

Step No.	Inst. Address	Instruction OP	A/I	B	d	Remarks	Effective No. of Characters Inst	Data	Total
	001	C	038	079		Check for Clear Address			
	008	B	035		/	of 099			
	013	L	073	116		Load Instructions into Pch. Area			
	020	,	110	106		Define Instructions in Punch			
	027	,	105			Area			
	031	B	101			Branch to Punch Area			
	035	/	T99			Clear Block of Storage			
	039	,	036			Modify Clear Instruction			
	043	A	076	038		To Next Lowest Century			
	050	K	036			Block			
	054	B	001			Branch to Compare			
	058	/	099			Clear from 099 to 000			
	062	1				Read Card			
	063	,	001			Set Word Mark in 001			
	067	/	001	116		Clear Punch Area			
	074		100			Constants			
	077		099						
						Column 36 contains a T for a 1.4K			
						machine, a Z for the 2K and an I for			
						the 4K model.			

FIGURE 53. CLEAR ROUTINE

Clear Storage Routine

This is a procedure that can be used to prepare core storage to accept program and data information. This is not the only clearing routine that can be used; others are left to individual creativity of the programmer.

The following two-card program can be used to clear storage of all characters and word marks. The character in column 36 of card 2 is variable according to the number of storage positions available:

T for 1400 positions,
Z for 2000 positions, and an
I for 4000 positions.

Multiplication and Division Subroutines

These are subroutines for multiplication and division operations, discussed here to illustrate programming methods and to aid programming for machines not equipped with the multiply-divide optional feature. These are not the only methods of performing these operations—they are typical methods.

Multiplication

This multiply sub-routine occupies the 900 block of storage, and provides for a maximum of a 9-digit multiplier, 11-digit multiplicand, and a 20-digit product.

A multiplier area is provided in storage positions 901-909, and the product area is assigned in storage positions 910-929. The multiplicand can be located anywhere.

Any program that uses this sub-routine must include a step that moves the multiplier address to location 937 (XXX) and the multiplicand address to location 952 (YYY).

At the completion of the multiply sub-routine the program instruction step 12 can use a branch to the main program or stop.

The routine starts in storage position 930. The product is found in 929 for a 9-digit multiplier, 928 for 8-digit, 927 for 7-digit, 926 for 6-digit, etc.

IBM 1401 PROGRAM CHART

FORM X24-6437-0
PRINTED IN U.S.A.

Program: Multiply Subroutine

Programmer: Date:

Step No.	Inst. Address	Instruction OP	A/I	B	d	Remarks	Effective No. of Characters Inst.	Data	Total
1	930	/	929			Clear product area	4		
2	934	L	XXX	909		Load multiplier into multiplier Area	7		
3	941	B	967	909	0	Test 909 for 0	8		
4	949	A	YYY	921		No 0, add multiplicand to product	7		
5	956	S	986	909		Reduce multiplier by 1	7		
6	963	B	941			Branch to zero test	4		
7	967	V	987	909	1	Test for word mark in 909	8		
8	975	L	928	929		No word mark--right shift	7		
9	982	B	941			Branch to zero test	4		
10	986	1				Constant 1	1		
11	987	M	930	230		Move product to output area	7		
12	994	3				Read and Print	1		
			XXX			Location of multiplier			
			YYY			Location of multiplicand			

FIGURE 54. MULTIPLY SUBROUTINE

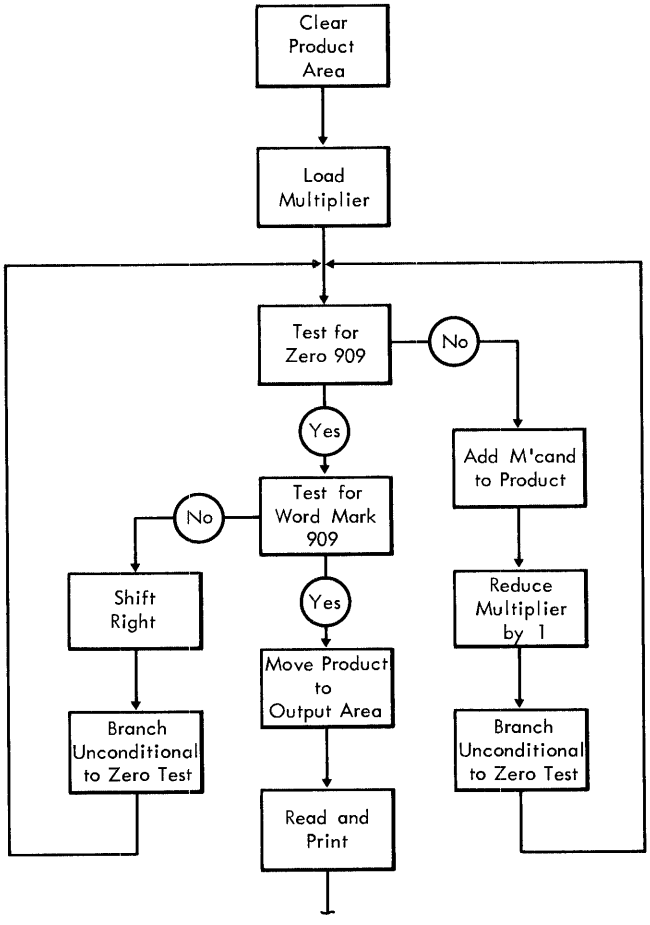


FIGURE 55. MULTIPLY FLOW CHART

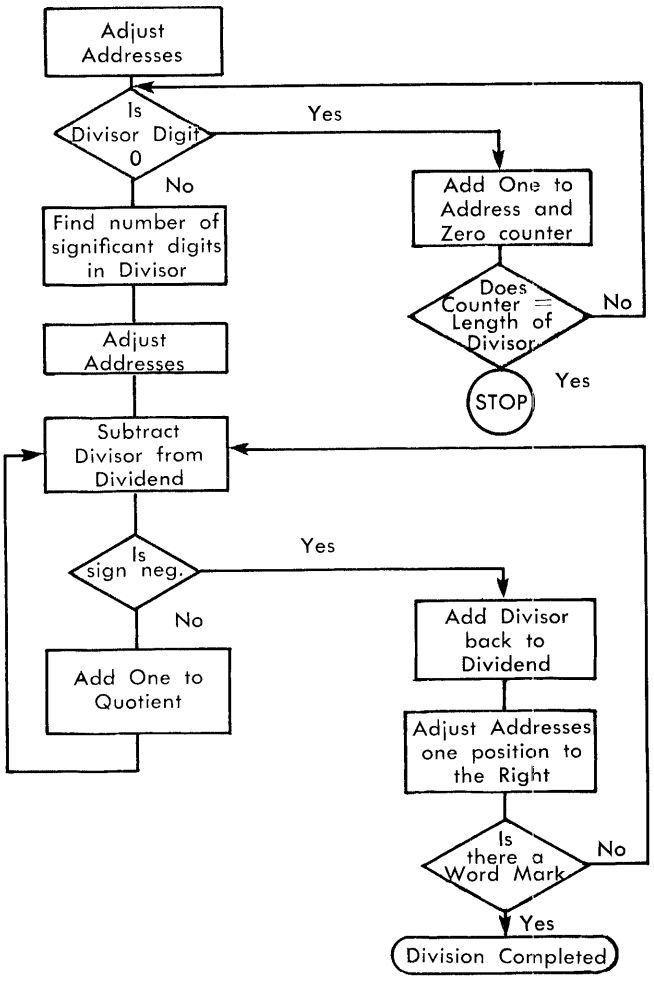


FIGURE 56. DIVIDE FLOW CHART

Division

The restrictions placed on this subroutine are:

1. The dividend and quotient fields must be of equal length.
2. All fields must be positive.
3. The divisor cannot contain more than nine non-significant zeros.
4. All fields must be located completely below address 999.
5. The remainder is left in the dividend field.

IBM 1401 PROGRAM CHART

FORM X24-6437-0
PRINTED IN U.S.A.

Program: DIVIDE SUB-ROUTINE

Programmer: _____ Date: _____

Step No.	Inst Addr	Instruction OP	A/I	B	d	Remarks	Effective No. of Characters Inst.	Data	Total
	516	M	508	529		Store address of word mark of divisor			
	523	B	642	YYY	0	Branch if divisor digit equals "0"			
	531	S	512	515		Find significant digits in divisor			
	538	A	515	502					
	545	A	515	505		Modify address			
	552	S	513	502		WWW and XXX			
	559	S	513	505					
	566	M	502	628					
	573	M	502	635					
	580	M	502	684		Set modified addresses into divide routine			
	587	M	502	691					
	594	M	502	740					
	601	M	505	643					
	608	M	511	625					
	615	M	511	681		Set divisor address			
	622	S	ZZZ	WWW		Subtract divisor from dividend			
	629	V	678	WWW	K	Branch if negative			
	637	A	513	XXX		Add one to Quotient			
	644	B	622			Repeat			
	648	A	513	529		Add one to YYY address			
	655	A	513	512		Increase counter by one			
	662	C	512	515		Test for divisor equals 0			
	669	B	523		/	Branch if unequal			
	674	.	746			Store--cannot divide by zero			
	678	A	ZZZ	WWW		Add divisor back to dividend			
	685	Y	499	WWW		Remove zone bits			
	692	A	513	628		Modify address by one			
	699	A	513	635					
	706	A	513	643					
	713	A	513	684		Modify address by one			
	720	A	513	691					
	727	A	513	740					
	734	V	746	WWW	1	Divide complete if word mark exists			
	742	T	622			Continue dividing			
	746					Divide complete			

Data for Division Subroutine

Location of Data Word	*Data Word*	*Description of Data*
499	blank	Constant
502	WWW	Address of word mark position of dividend
505	XXX	Address of word mark position of quotient
508	YYY	Address of word mark position of divisor
511	ZZZ	Divisor address
512	blank	Counter for # of zeros in divisor
513	1	Constant
515	Lq	Length of quotient

FIGURE 57. DIVIDE SUB-ROUTINE

CHARACTER	CARD CODE	BCD CODE						
		C	B	A	8	4	2	1
BLANK	No Punches	X						
.	12-3-8		X	X	X		X	X
¤	12-4-8	X	X	X	X	X		
(Undefined Special Character)	12-5-8 *		X	X	X	X		X
(Undefined Special Character)	12-6-8 *		X	X	X	X	X	
(Group Mark-Special Character) (Note 1)	12-7-8	X	X	X	X	X	X	X
&	12	X	X	X				
$	11-3-8	X	X		X		X	X
*	11-4-8		X		X	X		
(Undefined Special Character)	11-5-8 *	X	X		X	X		X
(Undefined Special Character)	11-6-8 *	X	X		X	X	X	
Δ (Mode Change (Delta) Special)	11-7-8		X		X	X	X	X
-	11		X					
/	0-1	X		X				X
,	0-3-8	X		X	X		X	X
%	0-4-8			X	X	X		
(Word Separator-Special Char.)	0-5-8 *	X		X	X	X		X
(Undefined Special Character)	0-6-8 *	X		X	X	X	X	
Tape Segment Mark-Special Char.	0-7-8			X	X	X	X	X
1401 Generated Special Character (Note 2)				X				
#	3-8				X		X	X
@	4-8	X			X	X		
(Undefined Special Character)	5-8 *				X	X		X
(Undefined Special Character)	6-8 *				X	X	X	
(Tape Mark-Special Char.)	7-8	X			X	X	X	X
+ 0	12-0	X	X	X	X		X	
A	12-1		X	X				X
B	12-2		X	X			X	
C	12-3	X	X	X			X	X
D	12-4		X	X		X		
E	12-5	X	X	X		X		X
F	12-6	X	X	X		X	X	
G	12-7		X	X		X	X	X
H	12-8		X	X	X			

FIGURE 58. 1401 CHARACTER CODE CHART IN COLLATING SEQUENCE

CHARACTER	CARD CODE	BCD CODE						
I	12-9	X	X	X	X			X
0	11-0		X		X		X	
J	11-1	X	X					X
K	11-2	X	X				X	
L	11-3		X				X	X
M	11-4	X	X			X		
N	11-5		X			X		X
O	11-6		X			X	X	
P	11-7	X	X			X	X	X
Q	11-8	X	X		X			
R	11-9		X		X			X
Record Mark ≠ or +	0-2-8			X	X		X	
S	0-2	X		X			X	
T	0-3			X			X	X
U	0-4	X		X		X		
V	0-5			X		X		X
W	0-6			X		X	X	
X	0-7	X		X		X	X	X
Y	0-8	X		X	X			
Z	0-9			X	X			X
0	0	X			X		X	
1	1							X
2	2						X	
3	3	X					X	X
4	4					X		
5	5	X				X		X
6	6	X				X	X	
7	7					X	X	X
8	8				X			
9	9	X			X			X

*Coding not valid for reading or punching card codes but can be used in tape operations.
The 1401 has the ability to read MLP card codes. The 1401 ignores the 8-9 punches when they appear in the same column. The 1401 does not punch out MLP card codes.
Note 1. In the 705, this is punched as (12-5-8).
Note 2. The A bit coding must be program generated in the 1401 (it cannot be read in from a card); however, it can be punched in a card, and punches as a "0" (zero zone).

(FIGURE 58. CONTINUED)

IBM 1401 Data Processing System Timing

Card Systems

The following definitions are used in specifying the 1401 timings:

LI —Number of characters in an instruction
LA —Number of characters in the A field
LB —Number of characters in the B field
LY —Number of characters back to right-most "0" in control field
LX —Number of characters to be cleared
LW—Number of characters in the A field or the number of characters in the B field, whichever is shorter

This list contains the formulas to be used in calculating the time required to execute an instruction, in milliseconds.

Name	*OP Code*	*Formula*
READ	1	Require .0115 [LI+1] milliseconds plus the timings as shown in (Figure 59).
PRINT	2	
PRINT READ	3	
PUNCH	4	
READ PUNCH	5	
PRINT PUNCH	6	
PRINT READ PUNCH	7	
READ RELEASE	8	
PUNCH RELEASE	9	
NO OPERATION	N	.0115 [LI+1]
STOP	•	"
STACKER SELECT	K	"
UNCONDITIONAL BRANCH	B	"
TEST AND BRANCH	B	"
TEST CHARACTER	B	.0115 [LI+2]
ZONE AND WORD MARK TEST	V	"
BIT TEST	W	"
SET WORD MARK	,	"
CLEAR WORD MARK	⊓	"
MOVE DIGIT	D	.0115 [LI+3]
MOVE ZONE	Y	"
COMPARE	C	.0115 [LI+1+LA+LB]
LOAD	L	.0115 [LI+1+2LA]
MOVE	M	.0115 [LI+1+2LW]
RESET ADD	+ 0	.0115 [LI+1+LA+LB]
RESET SUBTRACT	− 0	"
ADD (no recomplement)	A	.0115 [LI+3+LA+LB]
SUBTRACT (no recomplement)	S	"
ADD AND RECOMPLEMENT	A	.0115 [LI+3+LA+4(LB)]
SUBTRACT AND RECOMPLEMENT	S	"
MOVE AND ZERO SUPPRESS	Z	.0115 [LI+1+3LA]
CLEAR	/	.0115 [LI+1+LX]
EDIT	E	.0115 [LI+1+LA+LB+LY]
FORMS CONTROL	F	.0115 [LI+1] + 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.

Timing Chart

The main purpose of the timing charts (Figure 59) is to illustrate the processing time available during printing, reading, and punching. Knowledge of the execution time for the individual operation codes is necessary before these charts can be used effectively.

Card Read

The Read Start Time is the time devoted to accelerating the card reader and positioning the card for reading. The 44 milliseconds is the time required to read the card. During these 44 milliseconds, no processing takes place. The remaining 10 milliseconds is the time devoted to processing. To maintain 800 cards/minute, another read instruction would be required during this time. If the process time exceeds the 10 milliseconds allotted, the reader will then function at a rate of 400 cards/minute.

Printing

When the operation code 2 is given, the information in the print output area is directed to the printing mechanism. This consumes 84 milliseconds. The remaining 16 milliseconds is devoted to processing time. If the actual processing time exceeds the 16 milliseconds allotted, the basic print cycle is extended by the amount of the excess process involved. For example, if five additional milliseconds of processing were required (21 milliseconds total), the basic print cycle will now be 105 milliseconds. Converting this to lines per minute means that the machine is now printing at a rate of 571 lines per minute.

Card Punching

The start time is the time devoted to accelerating the card punch and positioning the card for punching. The actual time taken for punching a card is 180.5 milliseconds. No processing takes place during this time. The remaining 22.5 milliseconds can be utilized as processing time.

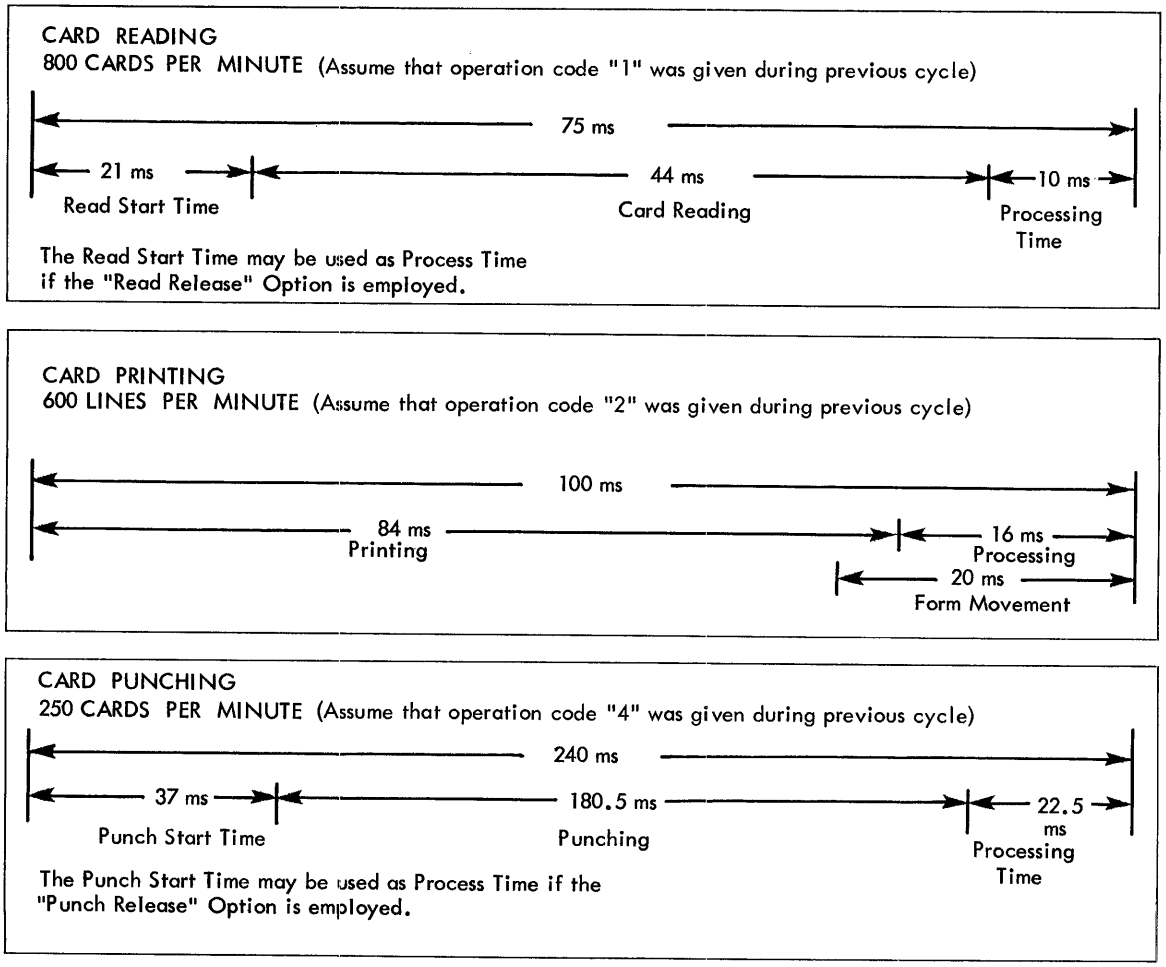


FIGURE 59. 1401 TIMINGS

Magnetic Tape

UNIT CONTROL U .0115 [LI+1] + tape movement time

1. Time for tape instructions is .108 milliseconds.
2. Tape operations (times expressed in milliseconds)

TAPE READ, WRITE
729 Model II: 10.8 + CN ms.
729 Model IV: 7.3 + CN ms.

REWIND
729 Model II: 1.2 minutes/reel
729 Model IV: .9 minutes/reel

ERASE FORWARD (add to subsequent write time)*
729 Model II: 108 ms.
729 Model IV: 72 ms.
*leaves about 8″ between records

BACKSPACE (after read)
729 Model II: 46.4 + CN ms.
729 Model IV: 32.6 + CN ms.

BACKSPACE (after write)
729 Model II: add 7.5 ms. to above
729 Model IV: add 5.0 ms. to above

NOTE:
C = Character rate, time in milliseconds per character.
N = Total number of characters traversed

for 729 II at 200 cpi. = .067 ms.
at 556 cpi. = .024 ms.
for 729 IV at 200 cpi. = .044 ms.
at 556 cpi. = .016 ms.

1401 Operation Codes

Op Codes	*Function*
1	Read
2	Print
3	Print-Read
4	Punch
5	Read-Punch
6	Print-Punch
7	Print-Read-Punch
8	Read-Release (optional)
9	Punch Release (optional)
A	Add
B	Branch
C	Compare
D	Move Digit
E	Edit
F	Forms Control
K	Stacker Select
L	Load

Op Codes	*Function*
M	Move
N	No Operation
S	Subtract
U	Unit Control (optional)
V	Zone and Word Mark Test
W	Bit Test (optional)
Y	Move Zone
Z	Move-Zero Suppress
@	Multiply (optional)
%	Divide (optional)
$\overset{+}{0}$	Reset Add
$\overline{0}$	Reset Subtract
•	Stop
⌑	Clear Word Mark
/	Clear
,	Set Word Mark

Index