Principles of Programming

Section 2: Introduction to Computing Equipment

IBM Personal Study Program

Section 2: Introduction to Computing Equipment

In the previous section we have surveyed some of the methods and concepts used in electronic data processing. In this section we introduce the "hardware" that carries out the operations involved. After these preliminaries, the next section will begin the discussion of how to specify the nature of the desired processing to the data processing system.

2.1 The IBM Punched Card

The starting point for entering data into most electronic data processing systems is the punched card. We must therefore become quite familiar with the way punched cards are used to contain and transmit information.

An IBM card is a piece of light flexible cardboard 7⅜″ wide and 3¼″ high. It is composed of 80 vertical columns, numbered from left to right. Each column may contain one of the digits 0 to 9, the letters A to Z, or any of certain special characters such as dollar sign or percent sign. There are twelve vertical punching positions in each column, of which the punching positions for 0 through 9 are identified by printing on the card. Numerical information is recorded on the card by punching a single hole in a given column in the position that represents that digit. For example, a single hole punched in the 2 position always means the digit 2 to IBM machines.

Alphabetic information is represented by a combination of two punches, a *numerical* punch and a *zone* punch. Positions 1 to 9 are referred to as the numerical positions. There are three zone punching positions:

12 zone—at the top edge of the card.
11 zone—just below the 12 zone position.
0 zone—just below the 11 zone position.

A punch in the 12 zone position is sometimes called a Y punch and a punch in the 11 zone position is sometimes called an X punch. This terminology has nothing to do with the representation of the letters X and Y and, because of the possible confusion, the terminology will not be used here. The 0 zone position is the same as the numerical 0 and is

labeled on the card. The 12 and 11 zone positions are not labeled, as this part of the card is usually set aside for the printing of headings.

The codes (combinations of punches) for the letters of the alphabet are shown in Figure 1. In understanding the basic idea of this coding, it may be helpful to note that the letters A through I are made up of a 12 zone and one of the digits 1 through 9; the letters J through R are made up of an 11 zone punch and one of the digits 1 through 9; the letters S through Z are composed of a zero zone punch and one of the digits 2 through 9. Note that the first letter represented with a zero zone is composed of a zero zone and a numerical punch 2, not 1.

A	12 and 1	J	11 and 1	S	0 and 2
B	12 and 2	K	11 and 2	T	0 and 3
C	12 and 3	L	11 and 3	U	0 and 4
D	12 and 4	M	11 and 4	V	0 and 5
E	12 and 5	N	11 and 5	W	0 and 6
F	12 and 6	O	11 and 6	X	0 and 7
G	12 and 7	P	11 and 7	Y	0 and 8
H	12 and 8	Q	11 and 8	Z	0 and 9
I	12 and 9	R	11 and 9		

Figure 1. Punched card coding of the letters of the alphabet.

The combination of a zero zone and a numerical 1 is used to represent the symbol / (slash). The various other special symbols are made up either of a 12 zone or an 11 zone alone, or the combination 8-3 or 8-4 alone or with the various zone punches. Thus, an ampersand (&) is represented by a 12 zone only; a minus sign (—) is represented by an 11 zone only. The percent sign is represented by the combination of an 8, 4 and zero zone. The combinations used for all of the standard allowable characters on an IBM card are shown in Figure 2. The word *character* is used to describe any digit, letter, or special symbol that can appear in one column of an IBM card.

-48 @
0-48 %
11-48 *
12-48 ¤
-38 #
0-38 ›
11-38 $
12-38 .
11 —
12 + &
01 /

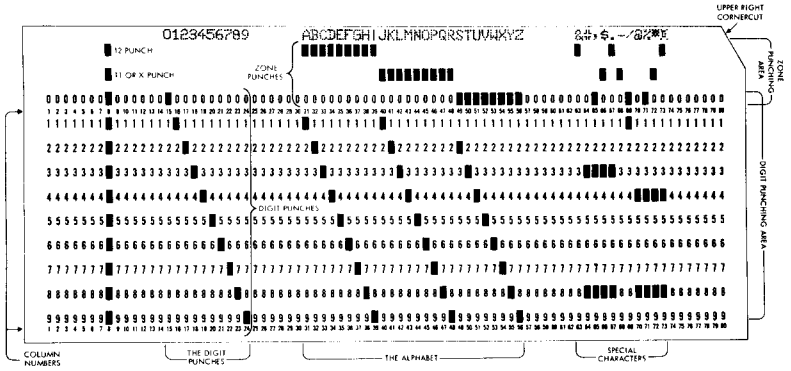

Figure 2. Arrangement of information on an IBM card.

There is no need to memorize these codes, because they are automatically punched by the depression of the keys on the card punch and are read automatically by the various IBM machines that can accept information from cards. Furthermore, a card punch can be equipped with a printing device that prints at the top of the card the character represented by each column. For cards produced by some means other than a card punch, the characters represented by selected columns can be printed at the top of the card, using a machine called an *interpreter*.

Some cards have a distinguishing color stripe, or have one of the corners cut. These features are provided for ease of use and recognition by machine operators and have no meaning to the computer. When it is necessary for any of the various equipment to be able to distinguish between different types of cards, the distinguishing information must be punched in the card. This may be done in many ways. One of the most common is the use of a 12 or 11 zone punch as an identification. For instance, in the example in Section 1.3, the master cards might have been identified by an 11 punch in some column set aside for this purpose.

In normal usage, columns on an IBM card are grouped into *fields*. A field is composed of one or more columns which together represent one piece of information. For example, Figure 3 shows a card layout for the sequential file processing example of Section 1.3. On this card, columns 1-4 make up the product number field. The card punch operator will always punch the digits representing the product number in these columns, and other card machines and the computer will be set up to recognize that field as always representing the product number.

Field assignments change from one job to the next. The group of columns that is regarded as one field in one job might contain parts of several fields in another. There is nothing built into the various card-handling machines that automatically regards any set of columns as being a field. This assignment must be planned by the user and the various machines *instructed* accordingly. On unit record equipment, field assignments are a matter of proper use of wires on the control panel in each machine. In the computer, field assignments are handled by programming techniques that we shall investigate in detail beginning in Section 3.

It often happens that data to be punched into a field contains fewer characters than the number of columns in the field. The most common way to handle this problem is to punch as many zeros in front of the data as are required to fill out the field. For instance, in the card form of Figure 3 four columns are set aside for the quantity. A quantity of 55 would thus be punched as 0055. In other situations it may be possible to leave these unused columns blank, but it is usually necessary to do one or the other consistently.

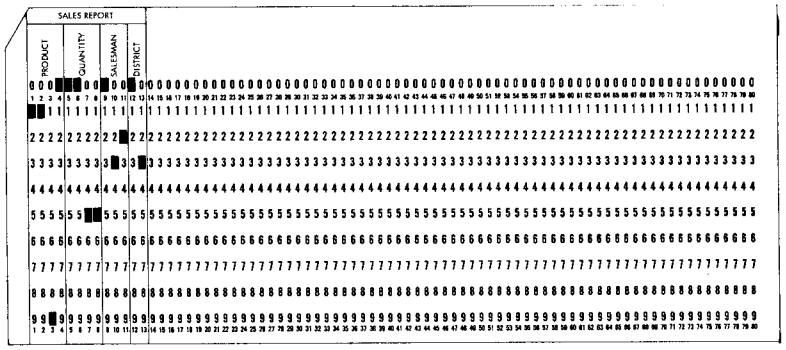

Figure 3. Sample card form for the sales summarization example of Section 1.3.

The handling of fields on a card provides us with the first example of a most important concept in data processing: How the equipment handles data depends on how the user *wants* the equipment to handle it. Suppose, for instance, that the four columns 20-23 contain punches representing the number 4397. If we are simply given a card containing these punches, we do not know whether they represent 4397 bolts, or $43.97, or $4.397 tax per $100 assessed valuation, or almost anything else. The simple and yet fundamental fact is that we don't have any way of knowing what the digits mean without knowing what they are *intended* to mean. It is the responsibility of the person who is planning the data processing operation to assign the desired meaning to these columns; to provide instructions which will tell the card punch operators how to punch data in the field; to set up the card handling equipment so that the digits in the field will be handled in the intended way; and to program the computer so that the intended operations will be properly carried out. For example, the computer operations required to properly process $4,397 may be considerably different from those required to properly process $4.397. The data processing equipment has no way of knowing anything about the difference. It is our responsibility to plan and execute the various operations that will make the equipment do what we want it to. We shall see this principle appearing repeatedly throughout our study, particularly in connection with some of the things we do in the computer.

Another good example of this principle is in the variation of meanings attached to an 11 punch in a column. An 11 punch by itself in a column is sometimes used to distinguish the card from other types of cards used in the same application. In another situation it might be used to cause a printer to print a minus sign (—). In another situation it might be used in combination with a numerical punch to represent one of the letters J through R. Again, it might be used in the units position of a field in combination with one of the digits 0-9 to signal that the entire field is a credit amount, or, mathematically, a negative number.

Review Questions

1. *Using only numerical punches, it would be possible to represent letters by combinations of a single punch in each of two columns. (This is actually done to represent letters within a number of computers.) What would be the disadvantage of this scheme compared with the zone and numerical punch method?*
2. *How is a zero numerical punch distinguished from a zero zone punch?*
3. *Suppose that an 11 zone punch has been punched over the least significant digit of a field to indicate that the field is negative. If that column is interpreted (without any special control panel wiring to separate the punches), what will be printed at the top of the card for that column?*
4. *Why must field assignments on cards be made early in the design of a data processing procedure?*
5. *Give four possible meanings (in different situations) of a punch in the 11 zone position in a column.*

2.2 IBM 1401 Data Processing System Components

Most computing systems are composed of a number of individual pieces of equipment that perform some part of the complete data processing operation. We may begin to get a more complete picture of what is involved in electronic data processing by considering briefly the major components of a 1401 system.

Processing Unit. Every computer system has as its heart an assembly which does the actual arithmetic processing of data. The central processing unit of the IBM 1401 is illustrated in Figure 4. The central processing unit also contains the electronic circuitry for carrying out arithmetic and other operations on data, for interpreting the coded instructions that must be placed in storage to direct the operation of the system, and for controlling all of the other parts of the system as a result of the interpretation of these instructions. It also contains the operator's console, which provides a certain amount of access to the information in the computer and allows for manual control of the system when it is being started.

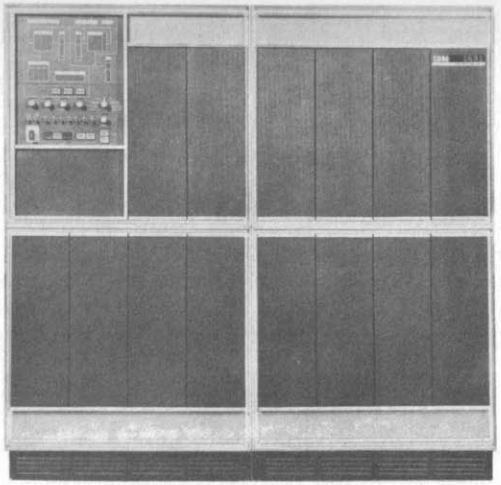

Figure 4. The processing unit of the IBM 1401.

IBM 1402 Card Read Punch. This is the component which reads the information from punched cards, that is, interprets the punched holes and translates the information into the form in which it is stored internally. The 1402 Card Read Punch is able to punch cards with the information which results from internal processing operations. Both reading and punching are controlled by the central processing unit as the result of suitable instructions placed in storage by the programmer. This unit is shown in Figure 5.

The read section, which is on the right in Figure 5, is able to read cards at a maximum rate of 800 per minute. The actual speed is ordinarily less than the maximum because complex processing operations require more time than is available while reading cards at top speed.

Figure 6 is a schematic diagram of the card transport mechanism in the card read punch. Looking at the read side, we see that a card is actually read twice. No information is transferred into storage by the first reading; the only action here is to make a count of the number of holes in each column of the card. When the information is actually read into storage, at the second reading station, a second hole count is made and compared with the first. If the two are not the same, reading stops and a red light comes on to signal the error. Some such checking is performed on all input and output operations in the 1401, although the hole count method obviously can apply only to card reading and punching.

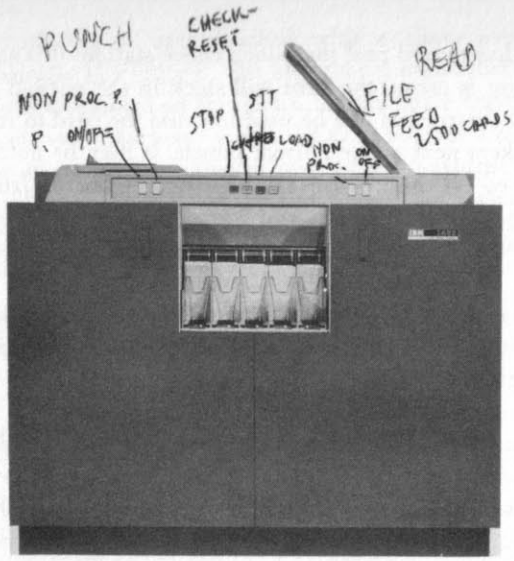

Figure 5. The IBM 1402 Card Read Punch.

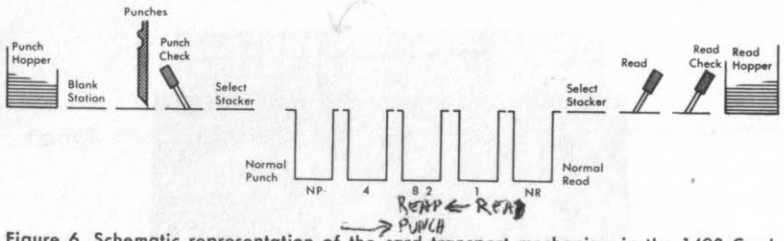

Figure 6. Schematic representation of the card transport mechanism in the 1402 Card Read Punch.

After the information on a card has been read into storage, the card may then be directed to one of three stackers. When no special action is taken, the card will fall into the normal read (NR) pocket. If it is desired to have the card stacked in one of the other two pockets to which a card can be directed from the read side, then a simple instruction can cause this stacker selection.

Figure 6 also shows the sequence of operations for punching, which can be done at a maximum speed of 250 cards per minute. A blank card from the hopper first moves past a blank station and is then punched, as directed by information from the central processing unit storage, at the punch station. As the holes are punched, a hole count is made on each column and this hole count is verified at the punch check station. As with reading, if the hole count is not the same for every column, the machine stops and a light comes on to signal the error.

After the card has moved past the punch check station, it is stacked. If no special action is taken, the card will stack in the normal punch stacker (NP). An instruction can be used to cause the card to stack in either of the stackers next to the normal punch. It may be noted that the stackers labeled NP and 4 are available *only* to cards from the punch side; the stackers labeled NR and 1 are available only for cards that have been read; the stacker labeled 8/2 can receive cards from either side.

It is possible to install another reading station at the position marked "blank" in Figure 6. This is a special feature called *punch feed read*, allowing the programmer to read a card on the punch side and then punch new information back into the same card.

IBM 1403 Printer. In addition to punching cards, it is possible to get information out of the 1401 system with the printer shown in Figure 7. This component is able to print a complete line of 100 characters at one time (it can be increased optionally to 132 positions). The maximum speed is 600 lines per minute. Each of the 100 (or 132) positions in a line can print any one of 48 different characters; these are the 26 letters, the 10 digits, and 12 special characters.

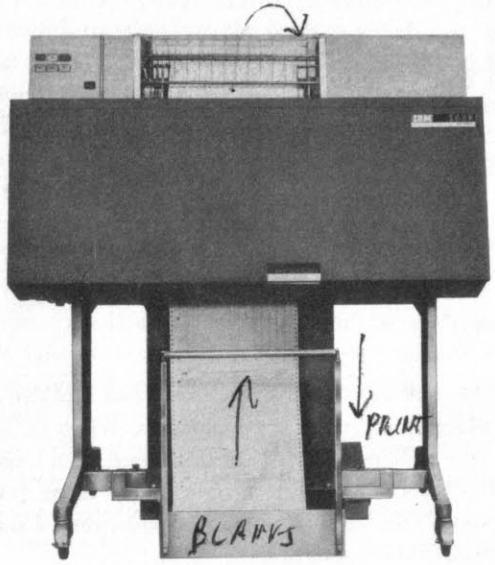

Figure 7. The IBM 1403 Printer.

The printing is accomplished with a chain assembly illustrated schematically in Figure 8. The alphabetic, numerical and special characters are assembled on this chain. As the chain travels in a horizontal direction, each character is printed as it reaches a position opposite a magnet-driven hammer that presses the form against the chain. Before a character is printed, it is checked against the corresponding position in the print area of the central processing unit storage to insure accuracy of printer output.

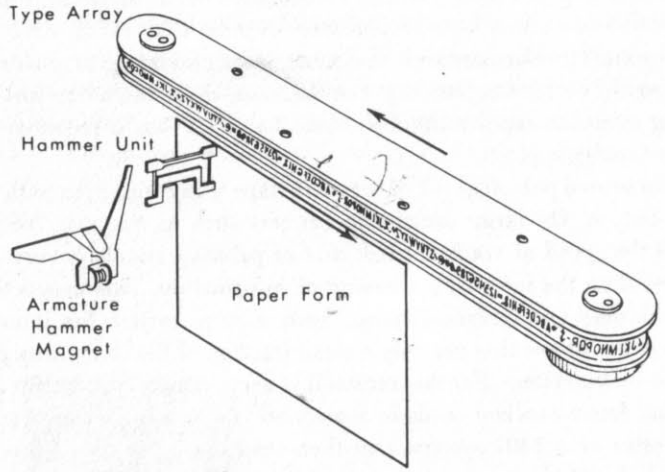

Figure 8. Schematic representation of the chain printing mechanism in the 1403 Printer.

The equipment described so far can be used as a complete computing system, called the IBM 1401 Card System. However, in common with most computers, the computing capacity of the 1401 can be greatly expanded by the addition of a number of other pieces of equipment. This means not only that a considerable range of price and computing power is available to the computer user, but also that the user is able to begin with an inexpensive system and expand it conveniently as his needs grow.

There are many special features that may be added to the basic system, more or less as minor modifications. We shall consider a number of these in the appropriate places in later sections. The following three optional system elements are of a more extensive nature and, when appropriate to the application, increase the computer power of the system by a considerable amount.

Magnetic Tapes. There are two principal uses for magnetic tapes in electronic data processing. The first is to increase the storage capacity of the computing system. The amount of information that can be stored in the central processing unit is limited for economic reasons. When it is desired to store large quantities of data for use during processing, it is necessary to employ some form of external storage. With the 1401, master files are frequently stored on magnetic tapes. It is still not possible to store the entire file within the central processing unit, and the principles of sequential file processing are much the same as with card files. The big advantage is that information from magnetic tape can be read and written a great deal more rapidly than cards can be read and punched. Furthermore, the same information on magnetic tape can easily be read repeatedly. For instance, it is possible to sort a file using magnetic tapes without the card handling that is required using a card sorter.

The second principal use of magnetic tape is in connection with input and output. On large computing systems such as the IBM 7080 and 7090 the speed of reading cards and of printing results is very much slower than the internal processing of information. This means that it can become uneconomical to use such a large system for input and output operations that use only a small fraction of the computing power of the entire system. For this reason it is very common to transfer information from punched cards to a magnetic tape, using either a special converter or a 1401 system, and then read the input data from tape. This is justifiable, as we have noted, in view of the cost of the large system and in view of the fact that tape reading can easily be 50 to 100 times as fast as card reading. The same considerations apply to output. A large system can write problem results on magnetic tape at high speed and then go on to other work while a special converter or the 1401 is used to print the results from the magnetic tape.

As indicated, there are special devices that have no other function but to perform these card-to-tape and tape-to-printer conversions. However, these devices are not capable of anything other than these single functions. The 1401, on the other hand, can be used to do such things as check the validity of the input data as it is being read, develop control totals, and perform other operations that will be described in later sections. Similarly, the large computer system can write its output information on magnetic tape in a condensed form, at high speed. The 1401 can then transform the condensed output into a readable format and print it.

Some 1401 systems are intended primarily for this sort of input-output conversion and editing. The machine system that is intended primarily for output can be obtained without a card reader and punch, and one intended primarily for input can be obtained without a printer.

All IBM computers use the same type of magnetic tape. It comes on a 10½-inch-diameter reel with either 1,200 or 2,400 feet of tape. The tape itself is a plastic ribbon one-half inch wide coated with a magnetic oxide material. A 2,400-foot tape can be used to record as many as 14 million characters. The format in which information is recorded on tape is considered in detail in Section 8.

Magnetic tape is read and written in a magnetic tape unit, of which there are three types. The IBM 729 and 7330 Magnetic Tape Units are pictured in Figure 9. There are two models of the 729 tape unit, designated 729 II and 729 IV. The differences between these three models are entirely in the speed with which they can read and write information. A tape may be written on one and read on any of the others; thus we say that the tapes produced by the three are *compatible*. (Tapes of different manufacturers are generally not compatible, although converters are available to translate from one type of tape to another.)

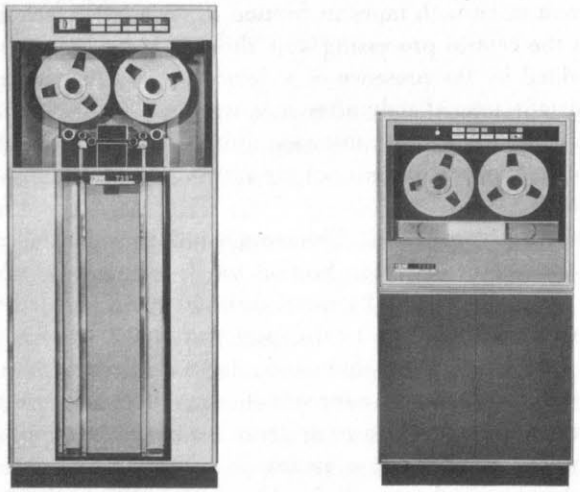

Figure 9. The IBM 729 (left) and 7330 (right) Magnetic Tape Units.

From one to six magnetic tape units may be attached to a 1401 system. Figure 10 shows a typical 1401 Tape System with card read punch, printer, and three 729 tape units. The spectrum of available computer systems runs from a few small computers that do not permit any tapes to large systems which can have over a hundred.

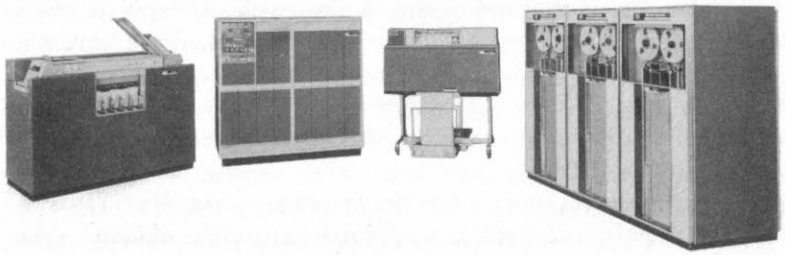

Figure 10. A typical 1401 Tape System with card read punch, printer, and three 729 tape units.

IBM magnetic tape units are provided with a number of automatic checking features to insure the accuracy of transmission of data. Certain additional information is automatically written on the tape to provide part of this checking. This is the subject of parity checking discussed in connection with tapes in Section 8. (A similar technique is used within the central processing unit also.) A second checking feature is provided by the presence of a device to read the information recorded on tape immediately after it is written. This is the *two-gap head* principle that is used on IBM tape units. The third checking feature is involved in the electronics of the reading process, and is called *dual level sensing.*

IBM 1405 Disk Storage Unit. This storage unit provides the random access bulk storage described in Section 1.4. It is composed of either 25 (Model 1) or 50 (Model 2) metal disks which are coated with a magnetic oxide material. The total capacity of a disk storage unit is either ten or twenty million characters, divided into records of 200 characters each. This is in the approximate storage capacity range of a single reel of magnetic tape. However, there is a fundamental difference between tape and disk storage, as we saw in the previous section.

Tape must be accessed sequentially. That is to say, if the tape is positioned at its beginning, there is no way to read a record in the middle of the tape without passing over all of the intervening records. At worst, this can cost several minutes. In computations which are properly organized for the use of tapes, this is not a disadvantage. However, if it is necessary to have access to records on a random basis where access time in minutes would be unacceptable, then the additional cost-per-character-stored of a disk system becomes justified. Any 200-character record anywhere in disk storage can be obtained in at most 0.8 seconds.

Magnetic disk storage is considered to be external storage, as is magnetic tape. Transfer of information between disk storage and the central processing unit must be initiated by the execution of appropriate instructions. This subject will be treated in detail in Section 9.

IBM 1407 Console Inquiry Station. With the batch processing made necessary by the use of card or tape files, all processing requirements are accumulated until the master file is to be processed. With the random access bulk storage, however, it is feasible to set up the system to accept inquiries about the status of stored information on a random basis, whenever an operator requests the information. This facility is provided by the console inquiry station, which is pictured in Figure 11.

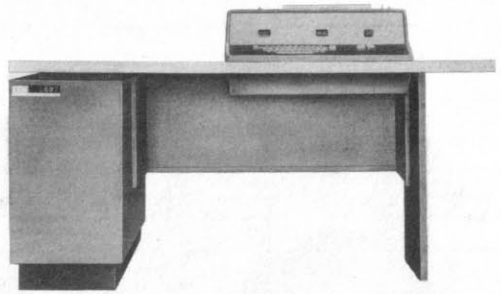

Figure 11. The IBM 1407 Console Inquiry Station.

When the console operator desires information from the system, he presses a request button. With appropriate instructions, the computer can detect the presence of this request and call for the request to be typed in from the inquiry station typewriter. The information typed in must be in a prescribed format established when the system was programmed. Instructions in the computer can then determine from the coded request what information is desired, obtain it from disk storage, and type out the information.

It should be realized that such requests would normally not be the major function of the computer. The computer would be set up to carry out some other primary function; the console inquiry requests would be interruptions of the primary program. Careful planning is obviously required to insure that the main program and the console program do not interfere with each other in any undesirable way.

This facility might be used in an inventory control application where orders are processed on a random basis as described in Section 1.4, when it is necessary to determine whether some urgent order can be filled. Facilities similar to the console inquiry station are available for many computer systems, but not all. In some cases the facilities are considerably more elaborate.

It is useful to picture the various components in terms of how they are related to each other, as shown schematically in Figure 12. We see that the internal storage of the central processing unit is the nerve center of the entire system. All input and output devices communicate with it. It contains the instructions that are used by the control section of the central processing unit to determine the actions of every part of the system. Any data to be processed by the arithmetic section of the central processing unit must be located in internal storage, although the data may previously have been brought to the internal storage from an external storage device such as a magnetic tape unit.

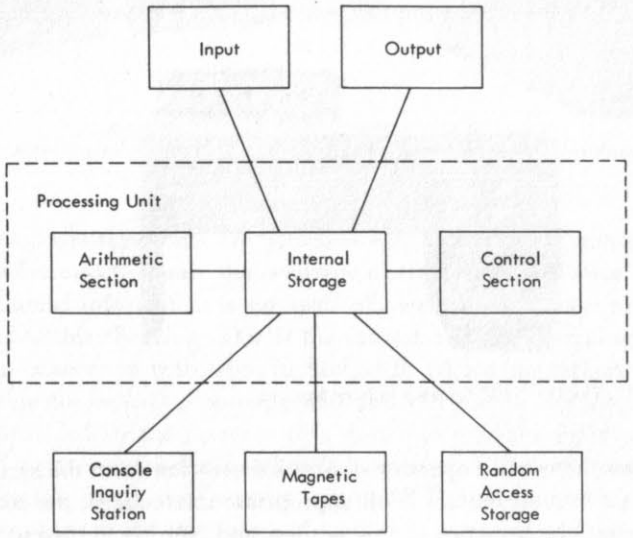

Figure 12. Schematic representation of the relationships among the components of a computer system.

Review Questions

1. *What is a third fundamental function of the central processing unit of a computer, besides storing data and doing arithmetic processing?*
2. *Briefly, how might the procedure of the example in Section 1.3 be modified if the computer has the punch feed read feature?*
3. *How many milliseconds (thousandths of a second) are required to print one line at full speed?*
4. *What are the two basic uses of magnetic tapes? Would both of them likely be applicable to a computer installation consisting* only *of an IBM 1401 system?*
5. *Assuming that an error in writing a tape will be detected when the tape is read, what is the advantage in detecting it* immediately, *as the two-gap head does?*

2.3 The Card Sorter and Collator

The Card Sorter. The card sorter, such as the IBM 83 in Figure 13, can be used for a variety of purposes, the most common of which is to sort a deck of cards into ascending sequence on a key or control field in the card. It may be noted in the figure that the sorter has a card hopper and 13 pockets, which are similar to the stackers on the 1402. As a card leaves the hopper, it moves past the reading station. This consists of a *brush* that can detect the hole punched in any one of the 80 columns in a card. When the brush senses a hole in the column, it directs the card to the correspondingly numbered pocket. If there is no hole in the column, the card is sent to the reject pocket. The 83 sorter can feed cards at the rate of 1,000 per minute.

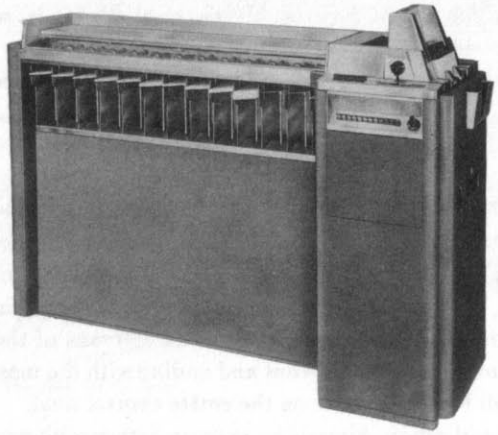

Figure 13. The IBM 83 Sorter.

The principle of the reading brush in the sorter is similar enough to the reading devices in other equipment that we may spend a moment examining how it works. As the card passes through the reading station, it passes over an electrically charged contact roller. While the card is passing *over* the contact roller, it passes *under* a brush. The brush may be set by the operator to read any one of the 80 columns in the card. Figure 14 shows schematically the relative positions of the contact roller, card and brush. As the card passes through the machine, bottom or 9-edge first, the brush is kept from touching the copper contact roller by the card, which acts as an insulator. However, when a punched hole is reached (a 4 hole in Figure 14) the brush drops into the hole and touches the contact roller. This completes an electrical circuit that actuates an electromagnet to direct the card to the proper pocket.

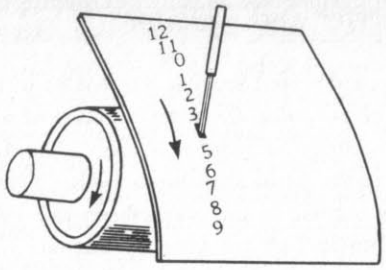

Figure 14. Schematic representation of the relative positions of the contact roller, card, and brush in a card-reading mechanism.

This same principle is used in all IBM card-reading machines, except that other machines have 80 such brushes and read all columns simultaneously. The result of completing the electrical circuit is, of course, different in other machines.

The single brush in a sorter can, of course, read only one column at a time. To sort a deck of cards into ascending sequence on a control field of several digits requires several passes of the deck through the sorter. The first sort pass is made on the *least* significant digit of the control field, after which the cards are picked up from the pockets in sequence. This puts all the cards with a zero in the least significant digit at the front of the deck, all the 1's next, etc. Then the deck is run through the sorter again, this time sorting on the next most significant digit, etc., etc. When the deck has been sorted on all columns of the control field, starting from the least significant and ending with the most significant, the deck will be in sequence on the entire control field.

The reader should satisfy himself by experimenting with an example that it is necessary to start with the least significant digit, not the most significant, remembering that after each sort pass the entire deck is picked up from the pockets and reassembled. That is, it is not normal procedure to keep the cards from each pocket separate after a sort pass.

Sorting cards on an alphabetic control field is somewhat more complicated, requiring either two complete passes on each column or special circuitry in the sorter. It is not too frequently necessary to sort cards on an alphabetic control field, although it is not uncommon to do an alphabetic sort on magnetic tape records, using the computer.

The Card Collator. The IBM 85 Collator is pictured in Figure 15. The collator has two card hoppers, called *primary* and *secondary*, the primary hopper being the one on the bottom. The collator has four pockets which are used in a way somewhat analogous to the stackers on the 1402 Card Read Punch. If we number the pockets from 1 to 4 from the right, cards from the primary feed can be moved to pockets 1 or 2, and cards from the secondary feed can be moved to pockets 2, 3 or 4. Thus, the cards from the two feeds can be merged, which is the most common application of the collator.

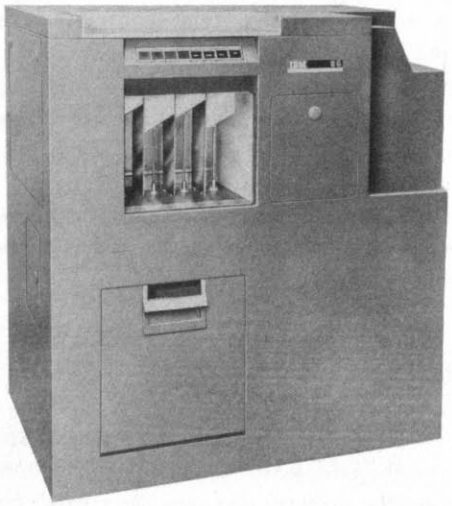

Figure 15. The IBM 85 Collator.

The basic principle of the operation of the collator may be better understood with the help of Figure 16. It may be seen from this figure that there are two sets of brushes in the path which cards from the primary feed hopper follow. These are identified as *primary sequence read* and *primary read*. The secondary cards can be read at only one station. There are 80 brushes at each of these read stations, so that all 80 columns can be read.

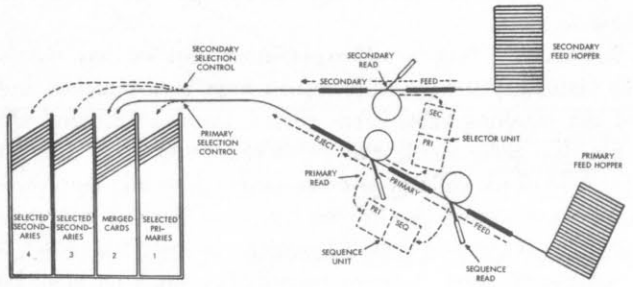

Figure 16. Schematic representation of the card transport mechanism of an IBM collator.

The heart of the collator's operation is based on the *selector unit* and the *primary sequence unit*. Looking first at the selector unit, we see that information can go to it from both the secondary read brushes and the primary read brushes. This selector unit is able to compare the information from the two sources and determine whether the primary field is less than, equal to, or greater than the information in the secondary field. The selector unit output can then be used to control the stacking of the card in either or both feeds and to control the feeding of additional cards from either feed. This, in principle, is all that is required for a merging operation.

The primary sequence unit receives information from the two sets of brushes in the primary feed path. It can thus be used to establish, for instance, whether or not cards coming from the primary feed hopper are in ascending sequence. The result of this comparison might be used to stop the machine if the cards are discovered to be out of sequence.

With these basic functions, and with a control panel by which the operator can select what columns are to be used to control them, a wide variety of operations can be performed.

Review Questions

1. *To sort a deck of cards into ascending sequence on a control field, which column should be sorted on first?*
2. *Can the primary and secondary cards both be sequence checked in a collator?*

2.4 System Components Used in Sequential File Processing

Now that we have a little better picture of the equipment which makes up the IBM 1401 Data Processing System, it would be well to take a new look at the sequential file processing example in Section 1.3. Figure 17 is a flow chart of the sales summarization application, drawn in a more informal style.

This flow chart is largely self-explanatory, but we may note one or two of its features. After the sales reports have been punched and verified and the resulting sales cards sorted, they are merged with the master file. We note, incidentally, that the detail deck goes into the secondary feed of the collator and the master deck into the primary. If there are no unmatched details, the entire merged file will appear in pocket 2; any unmatched details would go in pocket 3. After the incorrect sales cards have been corrected, they must be placed at the proper point in the merged deck. Ordinarily, there will not be many

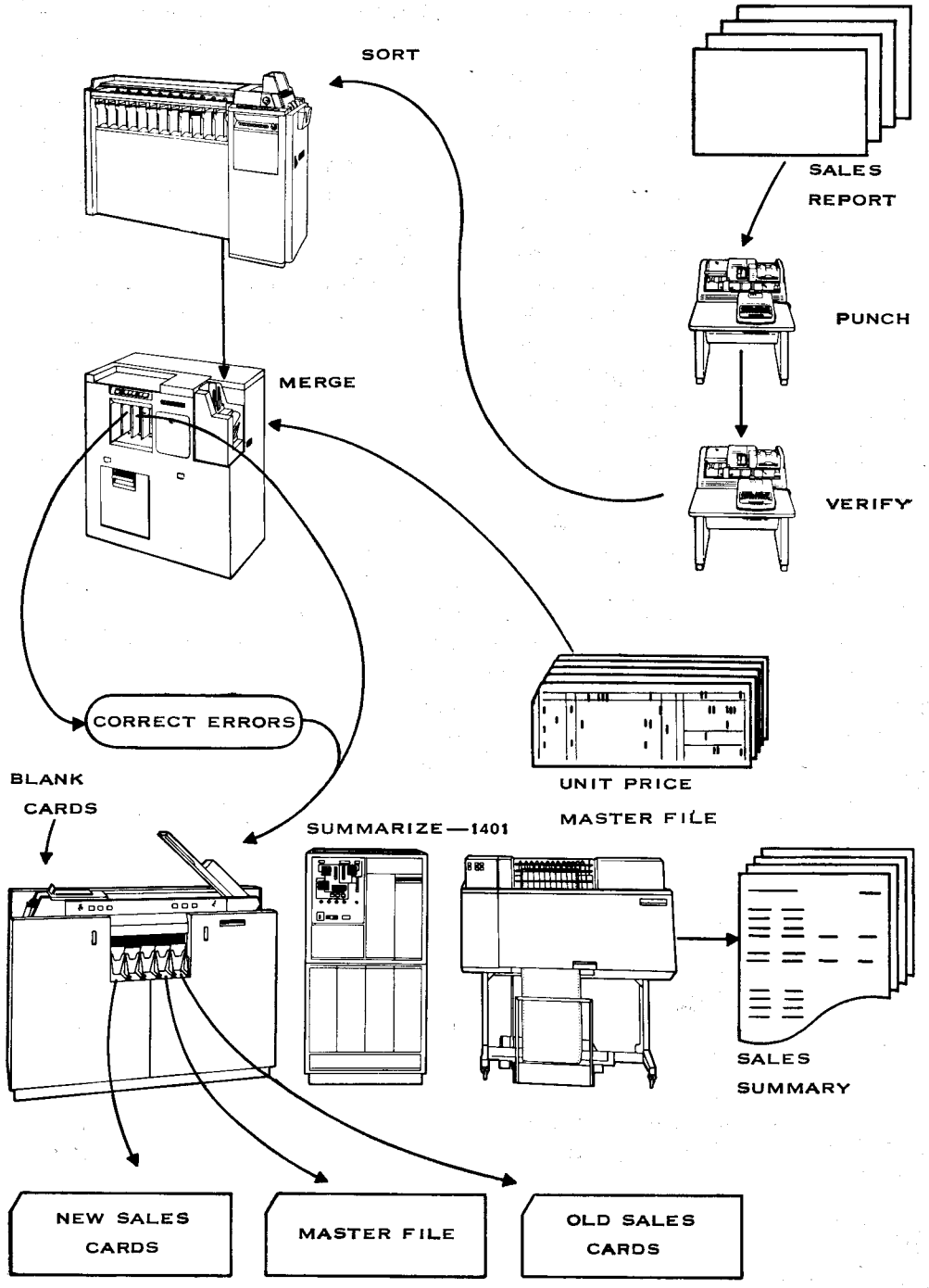

Figure 17. Flow chart of the sales summarization example of Section 1.3, drawn in an informal style.

of these and they can be inserted by hand. If, however, there are many of them, they can be merged by another collator operation. The merged deck is read by the 1402 Card Read Punch and the sales summary written by the 1403 Printer. As the procedure was described previously, new sales cards are punched giving the extended price of each product ordered. These appear in the normal punch pocket of the reader punch. The stacker selection can be used to separate the master file and the old sales cards as the merged deck is read.

The procedure is somewhat simpler using magnetic tapes. Figure 18 shows an informal flow chart of this portion of the application with a 1401 Tape System. The sales reports must be punched and verified as before, but they can then go immediately into the computer. Magnetic tapes can then be used to sort the information on the cards after the cards have been read. When this has been done, the master file (which now is also on magnetic tape) can be mounted on a tape unit and the sorted sales records processed against the tape master file. The sales summary will be printed in the same fashion as before.

There is one difference here: we have not shown the procedure for handling unmatched details. The proper way to handle this problem would depend somewhat on the total size of the job, the expected number of unmatched details, and the use to which the report is to be put. If in a normal month there are only a few incorrect sales cards, then the value of the sales summary may not be diminished significantly by simply ignoring them or by making manual corrections on the report if one or two large orders were omitted. If, on the other hand, it should happen through some sort of consistent error that a large number of sales cards are incorrectly punched, then part of the procedure could be repeated to correct the sales summary. One way to handle the problem would be to punch a card each time a sales record was found to be unmatched.

This discussion of handling of errors is highly relevant to any discussion of the work of programming. It can often happen that such considerations require a significant fraction of the total time to plan the job. To a large extent, such questions are properly the concern of the system designer, who establishes error procedures before the work of the programmer begins. However, error considerations must always be of some concern to the programmer. Furthermore, the programmer who wishes to progress to systems work must be highly conscious of such considerations.

2.5 Representation of Information in a Computer

Information in a computer is represented in a form which requires each storage or processing element to be able to take on only two distinct

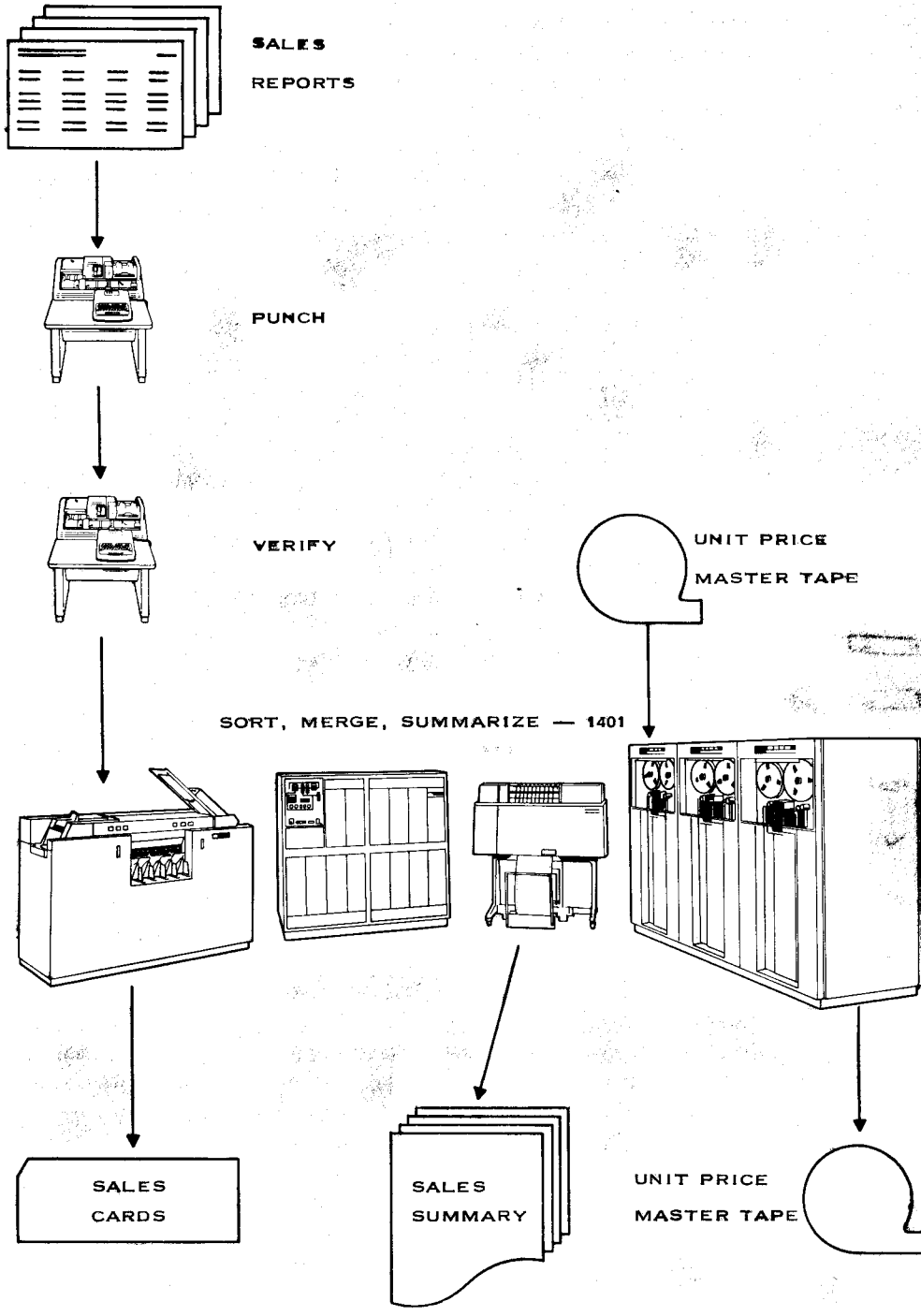

Figure 18. Flow chart of the sales summarization example of Section 1.3 using a tape system, drawn in an informal style.

states. For instance, one of the most common methods of storing information within a central processing unit is by the use of a *magnetic core.* A core is a doughnut-shaped piece of a special ceramic material about the size of a matchhead. It has certain rather special magnetic properties that make it very useful in the design and construction of a storage unit. The storage unit operates so that each core is always fully magnetized in one direction or the other; a core is not allowed to operate so that its magnetization is anywhere between these two extreme conditions. Thus, each core can be used to represent exactly two symbols.

Since there are 48 different symbols (and some other things) to be represented, a character must be represented within the computer by a *combination* of individual cores. Six cores would be sufficient to represent all of the characters, because there are 64 different combinations of the directions of magnetization of six cores. As we shall see a little later, each character is in fact represented by eight cores, the extra two being used for other purposes.

It would be inconvenient to talk for very long in terms of "combinations of directions of magnetization of magnetic cores." We therefore look for some simpler way to describe the two directions of magnetization. Actually any two convenient symbols or terms would do: north and south, on and off, yes and no, etc. The conventional way to represent the two states of a core is to call them zero and one, but this is simply a convenience of terminology. A device or condition which has exactly two possible states is described as being *binary*. It is then said to represent a *binary digit*, which is commonly abbreviated *bit*. We speak of the method of representing the 48 characters with six bits as *binary coding*.

Figure 19 shows the binary coding of the 26 letters, ten digits, and twelve special characters. (A few other combinations used within the computer will be considered later.) We see in the figure that only seven of the eight bits are shown; the eighth is called the *word mark* bit and is discussed below. The seven bits that are shown are seen to have conventional designations: CBA8421. The four rightmost bits (8421) are called the numerical bits, since they correspond in an approximate way to the numerical punches in card coding. The BA bits are for the same reason called the zone bits. The C bit is called the parity bit; we shall consider its function after looking into the coding scheme displayed in Figure 19.

We note that the decimal digits all have representations in which the zone bits are both zero; this corresponds to the fact that their card representations have no zone punches. The letters A-I have zone bits which are both 1; this corresponds in a way to a zone punch of 12. Similarly, the coding of the letters J-R have zone bits of 10, corresponding to an 11 zone punch. The letters S-Z have zone bits of 01 corresponding to a zone punch of zero.

Character	Binary Code C	BA	8421	Card Code	Character	Binary Code C	BA	8421	Card Code
0	1	00	1010	0	O	0	10	0110	11-6
1	0	00	0001	1	P	1	10	0111	11-7
2	0	00	0010	2	Q	1	10	1000	11-8
3	1	00	0011	3	R	0	10	1001	11-9
4	0	00	0100	4	S	1	01	0010	0-2
5	1	00	0101	5	T	0	01	0011	0-3
6	1	00	0110	6	U	1	01	0100	0-4
7	0	00	0111	7	V	0	01	0101	0-5
8	0	00	1000	8	W	0	01	0110	0-6
9	1	00	1001	9	X	1	01	0111	0-7
A	0	11	0001	12-1	Y	1	01	1000	0-8
B	0	11	0010	12-2	Z	0	01	1001	0-9
C	1	11	0011	12-3	&	1	11	0000	12
D	0	11	0100	12-4	.	0	11	1011	12-3-8
E	1	11	0101	12-5	¤	1	11	1100	12-4-8
F	1	11	0110	12-6	-	0	10	0000	11
G	0	11	0111	12-7	$	1	10	1011	11-3-8
H	0	11	1000	12-8	*	0	10	1100	11-4-8
I	1	11	1001	12-9	/	1	01	0001	0-1
J	1	10	0001	11-1	,	1	01	1011	0-3-8
K	1	10	0010	11-2	%	0	01	1100	0-4-8
L	0	10	0011	11-3	#	0	00	1011	3-8
M	1	10	0100	11-4	@	1	00	1100	4-8
N	0	10	0101	11-5	blank	1	00	0000	

Figure 19. Binary coding of the characters in a standard 1401 system.

The word *parity* is used here in the sense in which it refers to oddness or evenness. A careful study of Figure 19 will show that the representation of a character always involves an *odd* number of ones. This is done to provide checking of the accuracy of machine operation at certain crucial points within the machine. The number of ones in each character passing by these points is checked to determine that it is odd. If it is not, an error is signaled on the console and the machine is stopped. Parity checking, therefore, provides a very high degree of assurance that the machine is operating correctly.

The eighth bit associated with each character in core storage is called the *word mark* bit. We shall have to give this matter very careful consideration in later chapters. We may suggest for the time being that the word mark bit is used to signal to the computer the beginning and ending of fields of information in the storage. We saw previously that the assignment of card columns to fields is a matter of interpretation which must be handled by the user of the equipment; in the case of card machines this requires proper wiring of control panels. The 1401 system, however, contains no control panels and some other technique

must be used to signify within the computer where fields begin and end. This is the function of the word mark bits, which we shall be considering in much greater detail in later sections.

Algebraic signs of fields within the computer are indicated by the zone bits of the least significant digit of the field. If the zone bits are 10 (one-zero), the entire field is taken to be negative. If the zone bits are any other combination (00, 01, or 11), the entire field is taken to be positive. Unless there are special reasons to handle the matter differently, a positive field is ordinarily denoted by zone bits of 11.

Review Questions

1. *Can you find a relationship between the card codes for the special characters ($, %, etc.) and their binary representations?*
2. *How many individual cores are required to store 1,400 characters in core storage?*

Exercises

*1. These exercises concern a tape version of the first summarization in Section 1.3. Suppose that the master file is as before, except that it is on magnetic tape. Each tape record gives the product number and unit price of one product; the file is still in product-number sequence. Assume that the sales records are on another tape, and that they are already in sequence on product number. Assume, for this exercise, that there are no unmatched sales records, and that there is only *one sale per product*. There will be unmatched masters, however: there were no sales of some products. Draw a block diagram of the computer operations required to:

a. Read a sales record.

b. Read master records until finding the one having the same product number as the sales record.

c. When it is found, multiply the unit price (from the master record) by the number sold; print the product number and total price of the sale.

2. Extend the block diagram of Exercise 1 to process the entire sales tape. This will require a relatively simple modification of the flow chart to return to the reading of another sales record repeatedly. To stop the process when the sales tape has been completely read, use an end-of-file test: After the last sales record there is a special mark on the tape which indicates that the end of the file has been reached. The end-of-file indicator may be checked each time the sales tape is read.

The indicator will *not* be turned on by reading the last record, but by trying to read the "next" record, which will instead be the end-of-file mark. Thus when this mark is detected, the processing is finished (in this version of the problem). All that need be done is to rewind the two tapes and stop. (We are still assuming only one sale per product and no unmatched details.)

3. Extend the block diagram of Exercise 2 to handle the normal condition of many sales per product. Probably the simplest way to do this is to read successive sales records, summarizing the units sold as long as a comparison shows that sales records for the same product are being read. When a new product number is detected, save that sales record until after finding the master record for the previous set of sales records and completing the processing of that set. Then pick up again with the next sales record (which has already been read, remember). Hints: be sure that the comparison of successive sales records is started correctly; note that when the end-of-file mark on the sales tape is detected, the processing of the last set of sales records has not been completed.

It may still be assumed that there are no unmatched sales records. Do not try to test for errors in sequencing of the two tapes, and do not try to handle the possibility of tape reading errors.