Communicating with Computers

By JIM KYLE

Alphabetic and numeric symbols must be converted into a binary number code which can be operated on by a computer. In order to communicate back to the operator, reverse procedure is followed.

FROM a television network's studios on election night to the payroll department of most large companies, computers have moved into commanding positions. Thousands of words have been printed concerning the functioning of computer circuits—yet all these circuits are valueless until a means is provided for communication between the computer and its user.

It is obvious that communications are necessary, if for no other reason than to provide a way of determining the output of the computer. And since output implies that some input was provided, a second need for communications capability arises. Finally, the computer must be "programmed" to solve any problem given it, and the third communications requirement is for a manner of feeding the program into the machine.

While it might appear that communications techniques would be different for each different model of computer, actually the basic principles are almost identical for all models in current production. In fact, many of the techniques have been standardized within the computing industry, allowing interchangeability from one make of machine to another. This interchangeability has led to development of nationwide hookups between computers and users.

Table 1. Comparison of symbolic codes. Note that both Hollerith and ASCII are in order to allow direct sorting (the irregularity in the left digit for ASCII is due to the parity bit); Baudot is not. Baudot uses same codes for both letters and figures, transmitting a "37" or "33" to indicate which is meant. Once "33" is sent, all codes are interpreted as figures until "37" is sent. Punctuation differs from machine to machine.

CHARACTER	BAUDOT	HOLLERITH	ASCII (incl. parity)
0	26	0	260
1	27	1	061
2	23	2	062
3	01	3	263
4	12	4	064
5	20	5	265
6	25	6	266
7	07	7	067
8	06	8	070
9	30	9	271
A	03	12-1	301
B	31	12-2	302
C	16	12-3	103
D	11	12-4	304
E	01	12-5	105
F	15	12-6	106
G	32	12-7	307
H	21	12-8	310
I	06	12-9	111
J	13	11-1	112
K	17	11-2	313
L	22	11-3	114
M	34	11-4	315
N	14	11-5	316
O	30	11-6	117
P	26	11-7	320
Q	27	11-8	121
R	12	11-9	122
S	05	0-2	323
T	20	0-3	124
U	07	0-4	325
V	36	0-5	326
W	23	0-6	127
X	35	0-7	130
Y	25	0-8	331
Z	21	0-9	332
LTRS	37	not used	not used
FIGS	33	not used	not used
Car. Return	10	not used	015
Line Feed	02	not used	212

Machine-Language Communications

The majority of today's computers use the binary number system for all internal communications. Binary numbers consist of only two digits, 1 and 0. The base of the system is two; and two is written (in binary) as 10. Three would be 11, four is 100, and so forth. Computers use this system because the two digits (1 and 0) lend themselves ideally to electrical representation as high or low voltage levels; if "high" is defined as meaning "1", then four wires can represent up to 16 different numbers. The first wire would contain the lowest binary digit, the second the next lowest, and so forth, so that if the voltages were "high", "low", "high", and "low" (in order) the binary number contained would be 0101, which is equal to five.

A binary digit is known as a "bit" in the computer industry and a group of bits which are associated with each other is called a "word". The previous paragraph contains an example of a four-bit word, 0101.

One key specification of a computer is its "word length", since this is a measure of the largest binary number the machine is capable of processing. Word length in modern computers ranges from 8 bits to more than 40 bits; many general-purpose machines operate in the 20-bit to 30-bit range.

A 20-bit computer word is a difficult item to read or remember since each of the 20 bits must be correct. A typical such word might be "01011001111100100101". To aid in the reading and use of such words, both programmers and technicians make use of *octal* numbers.

Where binary numbers use *two* as a base and have only the two digits 1 and 0, octal numbers use *eight* for their base and have the *eight* digits from 0 through 7. Thus an octal number bears a close resemblance to the familiar decimal number and is easy to remember. The resemblance is only superficial; octal 100 is equal to decimal 64, and decimal 100 equals octal 144.

The only purpose for which octal is commonly used is to aid in the use of binary numbers; no computation is actually performed using the octal numbers.

The usefulness of octal stems from the easy mental conversion which may be made between binary and octal, or *vice versa.* To make the conversion from binary to octal, simply group the bits by threes, beginning at the right. Next, convert each group of three bits to decimal (001 = 1, 010 = 2, 011 = 3, 100 = 4, 101 = 5, 110 = 6, 111 = 7, and 000 = 0). "Push" the resulting decimal digits back together to obtain the octal equivalent for the binary numbers.

As an example, the 20-bit word cited previously is converted as follows:

0̲1011001111100100101							original binary
001	011	001	111	100	100	101	grouped by threes
1	3	1	7	4	4	5	converted by threes
			1317445				octal equivalent

Note that an extra zero had to be added on the left (underlined) to fill the last group out to three bits. "Leading zeros" have no effect on the value of a number; 001 is the same as 1.

Conversion from octal to binary is the reverse process. When the binary equivalents of the eight octal digits are memorized, either method may be done mentally.

Since the conversion is so simple, virtually all operating manuals, reference books, etc., for any computer give *octal* numbers for "machine language" communications. It is important to keep in mind that this is for human convenience; the machine uses only *binary* numbers and the octal numbers must be converted before the machine can use them.

All three of the main communications requirements of the machine are accomplished through binary numbers, composed of bits grouped in words of the length specified by the machine's design. Certain bit combinations are defined by the designers as "machine instructions" (frequently called "operation codes" or simply "op codes"). When the computer's control circuits detect these combinations on the input lines, specified machine actions result. For instance, if octal 7200000 is the op code for "add" in some machine, and the 21 input lines contain binary 11101-0000000000000000, the machine will perform an addition.

The op codes provide the machine-language means of programming, since the programmer may arrange the various operations in any sequence he finds necessary to solve a particular problem. The choice of sequence to be followed consitututes the art of programming, which is an entire subject in itself.

It might appear that the op codes prevent the machine from processing certain numbers, since octal 7200000 also represents a number. This does not occur, however, since *all* input is assumed to consist of op codes unless otherwise specified. The specification of input as data rather than as instructions is usually accomplished by storing data in a different part of the computer's memory than that used by the program. Thus the *location* of the number determines its meaning.

Of course, communications are necessary for the storage of the numbers in the memory in the first place. The op codes which command storage of a number usually contain certain spaces to be filled by the "memory address" of the number to be stored, and the address in which it is to be stored. If the basic "store" op code were 6300000, for instance, the two digits immediately following the "63" might specify the source address, and the final two digits specify the storage location. Thus, instead of sending the op code as "6300000", the programmer might send "6325050". This would cause the machine to take the number presently located in memory address "25" (010101, since all op codes are assumed to be in octal) and store it in address "50" (101000). If the first number sent to the machine is automatically placed in location "00" and each subsequent number is placed in the next higher numbered location, then the source location for each number is known. Input data can be transferred to the higher numbered portion of the memory, leaving only op codes in the low numbered part, and the separation is accomplished.

Output communication is usually by separate lines or indicators. Essentially the only machine-language output communication in most machines is the bank of indicator lights on the operator's console. Each light indicates a "1" binary value, in the corresponding bit position of the register to which the light is connected. Some machines have only one indicating register, while others have huge banks of lights. However, machine-language output is usually used only while troubleshooting the computer in case of breakdown.

The machine-language input communications provided for most computers consists of one or more banks of switches, either push-button or toggle, located (like the output indicators) on the operator's console. These switches are similar to the output lights; each switch controls the value of a single bit, with one switch position representing "1" and the other "0". After all switches are set, operating another switch inserts the setting into the associated computer register.

While it is possible to load a complete program and all its input into a machine in this manner (and a few machines exist which make no other provisions for program inputs), it is a slow and tedious process, open to many errors. For these reasons, direct switch-input communications are usually used only in troubleshooting or during initial "debugging" of new programs.

The "normal" input communications paths, like normal outputs, operate with other than machine language, through intermediate means. Usually one of several of the "symbolic codes" is employed. They will be discussed below.

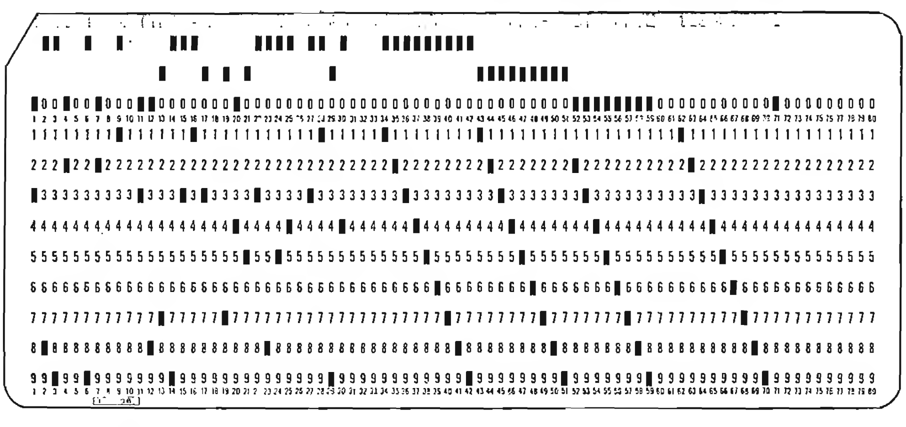
Symbolic Codes

A code is merely a table which matches two sets of symbols, so that one is understood to have the meaning normally attached to the other. Thus it appears that a "symbolic code" is a redundancy; the term is used in the computer industry to emphasize that one set of symbols in the code is the conventional set consisting of the alphabet, the decimal digits, and required punctuation marks, while the other set of symbols is a series of binary (usually written in octal) numbers.

The symbolic code most familiar to communications-oriented personnel outside the computer industry is the "7-unit teleprinter code" frequently called "Teletype® code". This code represents the letters of the alphabet, the numerals, the punctuation marks, and the various control functions necessary on a teleprinter (such as line feed, carriage return, upshift, and downshift) by 5-unit combinations of "mark" and "space" conditions. In addition to the five units which indicate the letter or other data, the code contains a "start" unit which is always a "space" and a "stop" unit which is always a "mark" to indicate the beginning and ending of each character.

Merely replacing the terms "mark" and "space" with "1" and "0" respec-

Fig. 1. Typical punched card, produced on IBM Model 026 key-punch machine with printing attachment. Printed line across top of card is for convenience of users only. Two unmarked rows between printing and row "0" are called "12" (upper row) and "11" (lower row) and are so identified under the Hollerith column in Table 1.



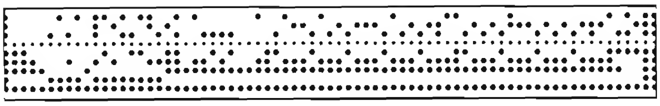


Fig. 2. Typical punched paper tape as produced by Model 33 Teletype equipped with tape-punch attachment. This tape was punched under computer control as part of output; computer was programmed not to produce parity bits. Normally, outside hole on wide side of tape (bottom)) would be parity. Each hole that has been punched indicates binary "1", lack of hole a "0".

tively changes the teleprinter code into a binary symbolic code in which every character begins with a "0" and ends with a "1". The five intervening bits determine the character.

When this change in terminology is made, the code becomes the *Baudot* code widely used by computers. Frequently a sixth "data" bit, called the "parity bit", is added to assist in determining the correctness of each character. The parity bit may be either a "1" or a "0", whichever is necessary to make the total number of "1's" in the six bits together an even number. Then a lost bit may be detected by counting "1's" in each character; if the total is odd, a bit has been lost from that character.

Baudot code may be used with punched-tape reading devices or with teleprinters connected to special "input buffer units" on a computer for either input or output. The conversion from Baudot code to the machine's language is accomplished by special circuits in the reading device or buffer unit; the conversion from alphabetic, numeric symbols to Baudot is accomplished by the teleprinter or tape punch. By this means, the user may type in data directly and need not have any concept of the computer's own internal language.

A code used even more widely than Baudot is that known as Hollerith code. This code is based on the punching of holes in cards at specified locations. From one to three punches may be made in each column of the card, and up to 80 columns may be punched in each card. Each column corresponds to one character; the specific character is determined by the number and location of the punches in that column. For instance, a single punch in row 9 designates the decimal digit 9. Two punches, one in row 11 and the other in row 5, designate the letter "E".

The familiar "IBM card" is a typical example of a Hollerith-coded card. These cards may be read into the computer by special card-reading machines which sense the presence and locations of the holes and convert the code into its binary equivalent for the specific computer with which they are used.

Since each card contains up to 80 characters and cards may be read at rates up to and exceeding 300 cards per minute, the punched-card communications channel is widely used for mass input of programs and data. The computer may also drive a card punch, producing new punched cards as output. This is a frequently used output channel, since many other business machines operate with punched-card data. Additionally, the output obtained during one pass through the computer may be used as input for a later pass if it is taken in the form of punched cards.

A relatively recent symbolic code which is, however, gaining almost industry-wide acceptance is the American Standard Code for Interchange of Information (ASCII). This code provides for 32 control characters and 96 data characters, allowing both capital letters and lower-case letters to be included in the alphabet along with all digits and punctuation marks.

ASCII is especially well suited to some of the more recent applications of computers in the areas of typesetting for printing plants; it remains compatible with all previous codes as well. This code is presently in use (completely, or in part) on punched-tape devices, teleprinters, video-display units, and for communications between remotely located computers. One of its major features is that any ASCII-coded device can communicate with any other, as long as both operate at the same speed. Data transmission speeds are standardized as well, with a choice ranging from 45 bits per second up to 2400 bits per second (equivalent to 40-w.p.m. telegraphy at the low speed up to approximately 200 w.p.m.).

Table 1 compares the three symbolic codes discussed here. Baudot and ASCII are shown in octal notation. The listing for ASCII includes the parity bit, since ASCII includes "odd parity" as part of its standardization. Hollerith code is shown in decimal notation; the hyphen indicates an additional punch location.

The communications channels with which these symbolic codes are usually used have already been mentioned briefly in the discussion of the codes. Now, let us take a more detailed look at each medium.

A typical Hollerith-coded card is shown in Fig. 1. To punch the card, a special key-punch machine is used. This machine has a keyboard similar to that of a typewriter, although the arrangement is sufficiently different from a typewriter that special training is required to operate the key punch.

To input information *via* punched cards, it is necessary to write out the information first, then deliver it to a key-punch operator to be punched into cards. After the initial punching, a second operator "verifies" each card by "re-punching" it in a different machine, from the same copy. If the data on the card is not identical to that indicated by the verifier's keying, an alarm light indicates the error. After verification and correction of any errors, the card is ready for loading into the computer's reader for input.

The verification step is essential; in a recent election tally in the Midwest, key-punch operators attempted to speed their output by accepting vote totals by telephone and punching without verification. The resulting unofficial returns contained errors of as much as 12,000 votes, which were traced in every case to single-digit key-punch errors on the cards. One "lucky" candidate discovered to his dismay when the official returns were posted that he had lost what had been declared unofficially an easy victory.

Because of the special additional equipment and trained operators required, card input is usually used only for large computer installations which process data in vast quantities.

Punched paper tape is similar in many ways to the card process but provides slower input rates since only one character is read into the computer at a time. The tape may be punched with teleprinter equipment, separate special tape punches, or by a tape punch connected to the computer output terminals. Fig. 2 shows a typical punched-tape input, in modified ASCII code, as produced by a Model 33 Teletype machine.

Output from a computer is frequently obtained by means of a "high-speed printer" unit, which is a special kind of typewriter that prints an entire line of data at once. These printers can produce output at rates greater than 800 lines per minute, gobbling stacks of continuous-form paper. These printers do not usually use a symbolic code which is one of the standard varieties; instead, they take binary output directly from the computer and convert it to symbols as determined by the computer's built-in "character set" (which is a special symbolic code of the specific computer, and normally is used only with the corresponding printer units). Frequently the printer's design and that of the computer are so closely related that neither can operate without the other.

Previous mention has been made of the similarity between teleprinter codes and symbolic codes and of the use of teleprinters for communication with computers. This means is one of the least expensive and simplest available for general input and output of data; its major disadvantage is its slow speed, in com- (*Continued on page 77*)

Computers

(*Continued from page* 52)

parison with other input-output devices.

Teleprinters, operating with either Baudot or ASCII codes, may be connected to computers either directly or through standard telephone lines (by use of special digital data sets at each end of the line). Thus they allow a user in New York to communicate with a computer in Los Angeles and permit nationwide networks of users to share a single computer. With special controller units at the computer, all users may communicate at apparently the same time (actually, the computer communicates with only one at a time but switches from one to another so rapidly that the users are not aware of the changing).

The teleprinter keyboard is virtually identical to that of a typewriter, so that anyone can provide computer input through this channel. The printout provided is also similar to that of a typewriter except that only capital letters are used.

Aside from the slow speed, the second greatest disadvantage of the teleprinter as a communications device is the vast amount of paper consumed (which must be disposed of in some manner). Even a brief bout with a program can result in a printout five to ten feet in length—before any usable results are obtained.

To overcome this problem, a number of manufacturers have developed "buffer displays" or "video display units" which operate in essentially the same manner as a teleprinter, except that the output is shown on a CRT screen rather than being printed on paper. This is known in the computer industry as "soft copy" (a paper print is "hard copy"). When the screen becomes cluttered with data, it can be erased and communication continues with a clear screen. Among industry leaders offering such units are *IBM*, *RCA*, *General Electric*, *Raytheon*, and *Stromberg-Carlson*.

Summary

Essentially, then, communication with a computer is accomplished by first coding alphabetic and numeric symbols into binary numbers so that the computer can make use of it. Within the computer, certain numbers in the "program" part of the memory cause similar numbers in the "data" portion to be manipulated. At the completion of the program, numbers in the "output" sub-part of the "data" portion of memory are either recorded into a symbolic code for most output devices, or transmitted directly to such devices as the printer and card punch. Finally, the all-number code (whether in machine language or in symbolic code) is decoded back into alphabetic and numeric symbols for the benefit of the user.

While it is possible to communicate with a computer directly in its own binary language by use of the console switches and indicator lights, the multiple-coding communication techniques add vastly to the speed of operation. Since all the coding is accomplished by machine (or mechanical means such as the self-coding keyboard of a teleprinter or video-display unit), the user gains speed by being able to communicate in symbols which are familiar to him and the computer gains speed by not having to wait for the user.

Present communications techniques are still short of the ultimate. *Bell Telephone Labs* has demonstrated a device which recognizes spoken words and converts them into computer machine language. A number of firms make units which re-convert machine language into speech (most are based on the principle of the time-by-telephone machines used in all parts of the country). Perhaps the day will soon arrive when an article of this nature must be titled "How to Talk to a Computer—and How to Listen to Its Replies". ▲