that are interesting and useful are the notations called *biquinary* (pronounced "by-kwy'nerry") and *binary* (pronounced "by'-nerry").

Biquinary means counting by 2's and 5's; the "bi" refers to two and the "qui" to five. It is the notation expressed in a limited way by the hands and feet of human beings, Roman numerals in ancient style, and the earliest of all calculating machines, the *abacus* or bead-counting frame. For example, in writing the number CCLXXXVIIII (two 100's plus one 50 plus three 10's plus one 5 plus four 1's), we are using biquinary notation. The more recent Roman numerals used IX in place of VIIII but this later style is not biquinary.

Two of the big electrical brains that have been finished by Bell Telephone Laboratories use the biquinary system. Seven relays are used for each decimal digit in the pattern shown in Table I.

Table I—Biquinary Relay System

Decimal Digit	Name of Relay: 00	5	0	1	2	3	4
0	1	0	1	0	0	0	0
1	1	0	0	1	0	0	0
2	1	0	0	0	1	0	0
3	1	0	0	0	0	1	0
4	1	0	0	0	0	0	1
5	0	1	1	0	0	0	0
6	0	1	0	1	0	0	0
7	0	1	0	0	1	0	0
8	0	1	0	0	0	1	0
9	0	1	0	0	0	0	1

(1 means "relay energized"; 0 means relay "not energized.") The first two relays are called 00 and 5 and the others 0, 1, 2, 3, and 4. To represent any digit, either 00 *or* 5 must be energized, and exactly one of the other five relays must be energized. If either of these conditions fails, then the machine gives an alarm and stops at once. When the operator examines the trouble-light panel, he detects the circuit that failed. The machine acts as its own policeman. This machine in months of operation has not allowed a single wrong result, except those due to human errors in instructing it.

There are still simpler notations for representing information in circuit elements having just two positions, such as relays or tubes. One of these is binary notation and its relatives. In pure binary notation, the *base* is not 10 as in decimal or alternately 2 and 5 as in biquinary. The base is 2, and the digits —which are only 0 and 1—report the powers of 2: 1, 2, 4, 8, 16, etc. Numbers 1, 2, 4, 8, etc., are represented by 1, 10, 100, 1,000, etc. Fractions ½, ¼, ⅛, 1/16, etc., are represented by 0.1, .01, .001, .0001, etc. For example, the figure 1,011 now means one 8 plus zero 4's plus one 2 plus one 1, and so equals 11.

Other samples of equivalents of decimal and binary numbers shown in Table II. The last binary number in the table is an example of a "recurring decimal" (or should we call it a *recurring binal?*) in binary notation.

Incidentally, if you, like the author, prefer to pronounce new words and signs under your breath, the binary numbers 10, 11, 1,000, and so on, should be pronounced "one-oh, one-one, one-oh-oh-oh" and so on. This way of referring to them gets away from confusion with decimal numbers.

Table II—The Binary Numbers

Decimal	Binary
0	0
1	1
2	10
3	11
4	100
8	1,000
16	10,000
17	10,001
32	100,000
0.5	0.1
0.25	0.01
0.125	0.001
1/3 or 0.33333....	0.01010101
or $0.\ddot{3}$	or $0.0\ddot{1}$

As shown by these examples, we can convert numbers completely from the decimal system into the binary system. Some electronic brains do just that, for so doing simplifies the circuits of an electronic machine. For example, the electronic brain known as Binac (Binary Automatic Computer), completed in 1949, handles all numbers in pure binary notation.

Adding in Binary Notation

Before we proceed to other forms of notation, let us consider how adding in binary notation is carried out—on paper, and by a relay circuit.

Adding in binary is carried out by this addition table:

	0	1
0	0	1
1	1	10

To add with this table, select one of the numbers you wish to add from the top outside row and the other from the left outside column. Draw a vertical line (imaginary if you wish) from the number selected in the top row down through the column of numbers on the inside, and a horizontal line from the number in the left column across the row of numbers on the inside. You will find the sum of the two numbers where the two lines cross. Thus 0+0=0, 0+1=1, 1+0=1, and 1+1=10, (i.e., 2). No one could ask for a simpler addition table! Instead of the 100 entries of the decimal addition table, the binary addition table has only four.

For an example, let us add 1,011 (8 plus 2 plus 1, or 11) and 1,101 (8 plus 4 plus 1, or thirteen):

```
 1,011
 1,101
------
11,000
```

The result is 11,000 (16 plus 8, or 24). We may obtain it by the following "schoolboy" routine: (column 1, starting from the right) 1 and 1 is 10, put down 0, and carry 1; (column 2) 1 and 0 is 1, and 1 to carry, is 10, put down 0, and carry 1; (column 3) 0 and 1 is 1, and 1 to carry, gives 10, put down 0 and carry 1; (column 4) 1 and 1 is 10, and 1 to carry is 11, put down 1 and carry 1; (column 5) put down 1.

Binary adding circuits

Now let us design a set of circuits which will carry out the operation of addition in binary notation. We shall begin with a preliminary circuit, pre-

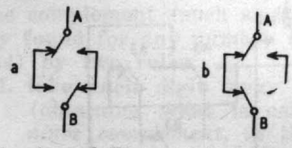

Fig. 3—A and B contact patterns that are used to simplify the adding circuit.

liminary because it uses many more relays and more successive impulses than are necessary. But it is useful because it shows the steps needed to reach a final circuit.

First, two binary numbers to be added, each of four binary digits, are stored in the two registers A and B (called Augend "to be increased" and Addend "to be added"). The relays marked A1 to A4 store the digits in the digit columns 1 to 4, counting from the right—first the one digit, then the twos, the fours, and the eights digits. (See time 1 of Fig. 1). At some previous time these relays have been picked up, and now terminal T1, through the hold contacts of these relays, holds them up while the later circuits operate. For example, suppose 1,011 is stored in the A or Augend register. Then relays A4, A2, A1 are

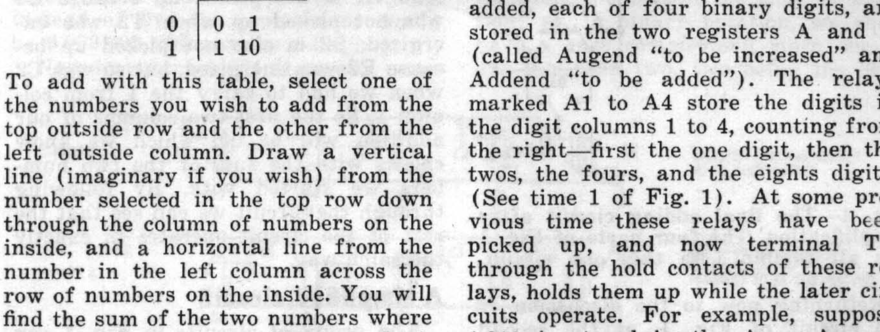

Fig. 2—A 4-column binary carry circuit that will carry 0 as well as carry 1.