The Clear Timing Pulse which we need for line T5 into the multiplier of the last article is a series of control pulses as follows:

Cycle	*Pulse Series*
1	1111 1111
2	0000 0000
3	0000 0000
4	0000 0000
5	1111 1111
6	1111 1111

To obtain this configuration of pulses—and the 1111 1111 can be one continuous signal, not necessarily a series of 8 separate positive pulses—we make use of a flip-flop. We need to set the flip-flop at the start of cycle 1 and cycle 5, and reset the flip-flop at the start of cycle 2. This we can do by appropriate connections through crystal diodes from (1) L1 and L5, and (2) L2, respectively. The reason we are using two set pulses and only one reset pulse is that when the machine is turned on, the first set pulse from line L1 is required.

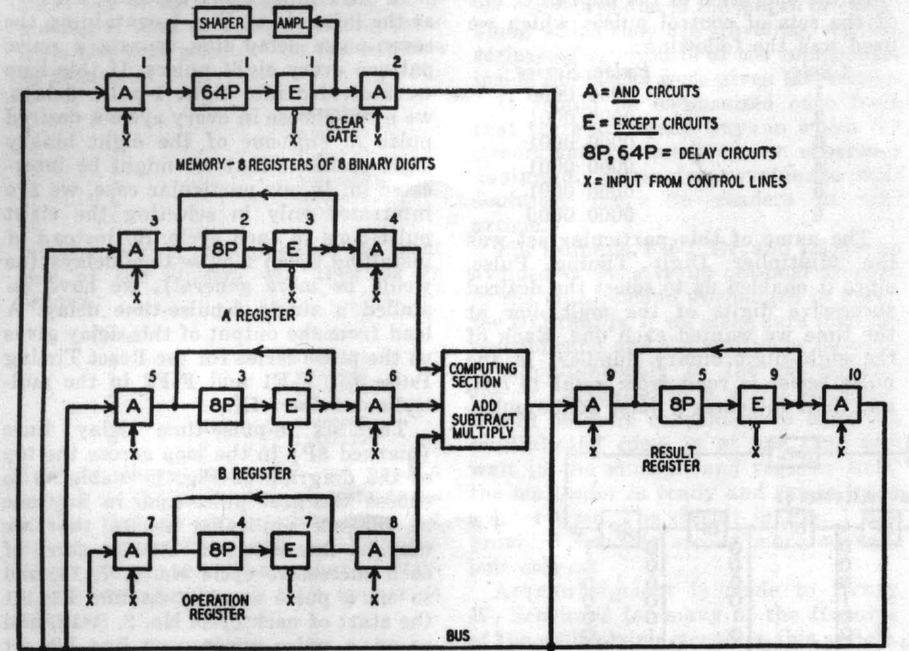

Fig. 3—Storage registers and computing facilities assembled form partial brain.

Ordinary electrical delay lines are good for delays of about 50 pulse-times. After that, the pulses become indistinct. To get accurate control pulses at much longer intervals, the type of circuit shown in Fig. 2 may be used. Here there are two circulating delay line loops, one repeating at intervals of every 25 pulse-times and the other repeating at intervals of every 32 pulse-times. These two numbers have no common factor, so the pulses applying for admission to the AND circuit will match each other only once in 25 × 32 times, or once in 800 pulse-times.

We can now assemble some storage registers and some computing facilities, and begin to obtain a whole electronic computer. In Fig. 3 we have drawn a schematic diagram showing:

a Main Storage, or memory, of 8 registers of eight binary digits each, in a 64 pulse-time delay line;

a Computing Section, which can add, subtract, or multiply;

an A-Register, and B-Register, which can take in numbers to be operated on in the Computing Section;

an Operation Register, which can take in the instruction telling the operation to be performed; these last three are the Computing Section Input Registers;

and a Result Register, which will hold the result of the operation produced by the Computing Section.

At the bottom of the diagram is the *bus*, a line along which numbers can travel from any register in the Main Storage to any Computing Section Input Register, and back again from the Computing Section Result Register to a register in Main Storage. Permission to any number to travel on the bus depends on the opening of the AND circuit. The operation of the circuit hinges on the control lines running to the ten AND circuits and the five EXCEPT circuits. These 15 control lines and the 16th line, the input of the Operation Register, all marked as ending with x, lead to controlled timing pulses and signals, and are related to the programming of an electronic computer.

CHART OF MAJOR AND MINOR CYCLES				
Process *Step*	Major *Cycle*	Minor *Cycle*	AND Circuits *Conducting*	EXCEPT Circuits *Inhibitina*
1	1	2	2, 3	3
2	2	5	2, 5	5
3	2	8	7	7
4	3	1	4, 6, 8	—
5	4	1	9	9
6	4	7	10, 1	1

A typical problem

How does this partial schematic of an electronic computer operate? In the Chart we show how this assembly would carry out a typical problem like:

> Take the number in the 2nd register, and the number in the 5th register, multiply them, and put the result in the 7th register.

The Chart lists minor cycles and major cycles. What are they? A minor cycle consists of eight pulse-times, beginning with the first digit of an 8-binary-digit number, and ending with the last one. A major cycle consists of eight minor cycles, a time sufficiently long for all the numbers in the Main Storage to circulate once completely around their loop. In general, to get any desired number out of Main Storage, we have to wait until it comes round the loop and grab it then.

To carry out our problem, the first thing is to get the number in the 2nd register out of Main Storage. This we do by waiting until the 2nd minor cycle comes along; we then open AND circuit No. 2, and let the series of pulses come out into the bus. But there is no way of storing them there, so we simultaneously open AND circuit No. 3, allowing this number to go into the A-Register delay line. This will do us no good unless we clear out of the A-Register any number already there, so we energize EXCEPT circuit No. 3. In this way we succeed in making the transfer we desire. This is step 1.

In step 2, we proceed in almost the same way, and transfer the number in the 5th register, using minor cycle 5. In step 3, we assume that the command for making the computing section multiply is available at minor cycle 8, and transfer it at that time.

Having filled the input of the Computing Section with the information it is going to use, in step 4 (the time is now minor cycle 1 within major cycle 3) we send the numbers and the operation into the computer, and assuming that not more than 6 or 7 minor cycles are necessary for the multiplication, the result comes out into the Result Register at the time major cycle 4, minor cycle 1.

Since the answer is to be stored in the 7th register, we do not have to wait for the next major cycle, but in this same 4th major cycle we can send in the number at the time of minor cycle 7. So at that time we transfer from the Result Register through the bus into the 7th register of Main Storage by opening the appropriate AND circuits, A10, A1, and operating EXCEPT circuit No. 1 to clear out any previous information in this register.

The chief topics remaining to be discussed are function tables, programming, and input and output. These we shall begin in the next article.

(Continued next month)