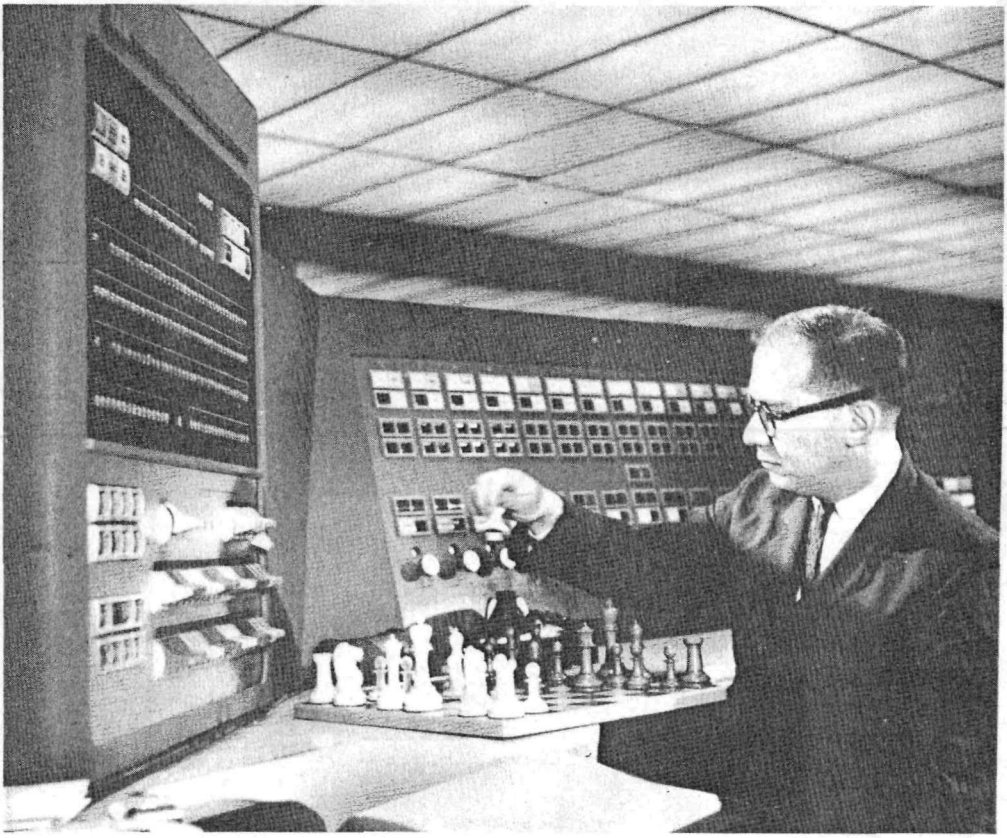
COMPUTERS

by William R. Corliss



U.S. ATOMIC ENERGY COMMISSION
Division of Technical Information
Understanding the Atom Series

The Understanding the Atom Series

Nuclear energy is playing a vital role in the life of every man, woman, and child in the United States today. In the years ahead it will affect increasingly all the peoples of the earth. It is essential that all Americans gain an understanding of this vital force if they are to discharge thoughtfully their responsibilities as citizens and if they are to realize fully the myriad benefits that nuclear energy offers them.

The United States Atomic Energy Commission provides this booklet to help you achieve such understanding.

Edward J. Brunenkant

Edward J. Brunenkant, Director
Division of Technical Information

UNITED STATES ATOMIC ENERGY COMMISSION

Dr. Glenn T. Seaborg, Chairman
James T. Ramey
Wilfrid E. Johnson
Dr. Theos J. Thompson
Dr. Clarence E. Larson

COMPUTERS

by William R. Corliss

CONTENTS

CALCULATING PRODIGIES 1

BIRTH OF COMPUTERS 2
- Sticks and Stones and Abacuses 2
- That Strange Genius Charles Babbage 4
- From Shell Trajectories to Weather Forecasting 7
- An Automated, Computerized World? 12

ANATOMY OF A COMPUTER 13
- The Generalized Digital Computer 13
  - *Primer in Binary* 14
- The Input Unit 17
- Magnetic Memories 19
- Calculations Are Made in the Arithmetic Unit 25
  - *Primer in Computer Electronics* 26
- The Switchyard Magnificent—The Control Unit 32
- Getting Data Out of a Computer 35
- How to Talk to a Computer 38

ANALOG COMPUTERS 42
- Analogs Measure Rather Than Count 42

SMALLER AND SMARTER COMPUTERS 47

APPENDIX The Interest of the Atomic Energy Commission in Computers 51

ANSWERS TO PROBLEMS 53

SUGGESTED REFERENCES 54

United States Atomic Energy Commission
Division of Technical Information

Library of Congress Catalog Card Number: 66-61907
1966; 1967(Rev.)

THE COVER

The photograph on the cover is a scene from the movie "Thinking Machines" (see Motion Picture list on page 55). The IBM computer has been programmed to play chess. The movie describes some of the details of programming and the resulting strengths and weaknesses of the machine's game are analyzed. Chess offers so many possible moves that neither man nor machine could conceivably play a "perfect" game. The machine lost the game.

THE AUTHOR

William R. Corliss is an atomic energy consultant and writer with 12 years of industrial experience including service as Director of Advanced Programs for the Martin Company's Nuclear Division. Prior industrial connections were with the Flight Propulsion Laboratory of the General Electric Company and with Pratt & Whitney Aircraft Company. Mr. Corliss has B.S. and M.S. Degrees in Physics from Rensselaer Polytechnic Institute and the University of Colorado, respectively. He has taught at those two institutions and at the University of Wisconsin. He is the author of *Propulsion Systems for Space Flight* (McGraw-Hill 1960), *Space Probes and Planetary Exploration* (Van Nostrand 1965), and co-author of *Radioisotopic Power Generation* (Prentice-Hall 1964), as well as numerous articles and papers for technical journals and conferences. In this series he has written *Neutron Activation Analysis*, *Power Reactors in Small Packages*, *Direct Conversion of Energy*, *SNAP—Nuclear Reactor Power in Space*, and was co-author of *Power from Radioisotopes*.

COMPUTERS

By WILLIAM R. CORLISS

CALCULATING PRODIGIES

"Multiply in your head 365,365,365,365,365,365 by 365,365,365,365,365,365," Dr. Adams asked ten-year-old Henry Stafford in 1846. The boy bit his hands. His eyes rolled. He smiled and talked. Then, in less than a minute, he gave the correct answer,

133,491,850,208,566,925,016,658,299,941,583,225.

Zerah Colburn, son of a Vermont farmer, was another calculating prodigy. When he was eight, he was asked to compute 8^{16} in his head. Without hesitation, he gave the answer, 281,474,976,710,656, a feat that caused scholars in the examining audience to weep with envy.

We do not know how these mental-arithmetic marvels are performed any more than we know how musical prodigies like Mozart could recall every note in a long symphony. All we know is that such gifts are destined for only a few, and that if this modern world of 3 billion persons is to run smoothly and with dispatch, it must rely upon nonhuman computers to figure bank balances, keep inventories, calculate rocket trajectories, and design nuclear reactors.

This booklet relates how wires, transistors, and human ingenuity are combined to produce machines that surpass all the calculating prodigies that ever lived in speed, accuracy, and stamina, though perhaps not in the matter of mystery.

BIRTH OF COMPUTERS

Sticks and Stones and Abacuses

Man's first conscious mathematical operations probably involved only simple counting: the number of faces in the tribe, the number of cattle in the herd, and the number of wives owned. When the numbers involved exceeded the number of fingers (and perhaps toes), some new form of reckoning had to be invented. Piles of sticks and stones and marks in the sand or on cave walls would have been logical calculating* aids for primitive men. Addition and subtraction could be carried out merely by adding and erasing symbols from the crude tally sheets. Today, modern electronic digital computers count in this same simple way. The cave wall is replaced by arrays of iron rings strung on wires. Instead of drawing a mark on the wall, we now magnetize an iron ring. Erasure is accomplished by reversing the direction of magnetization.

Man's mathematics has progressed far beyond the simple notion of counting to algebra and the calculus. The electronic digital computer, moreover, with counting as its only stock in trade, has followed closely behind mathematical advances. It seems that any calculational problem, whether it is figuring the best move in a chess game or checking airline seat reservations, can somehow be accomplished by counting alone.

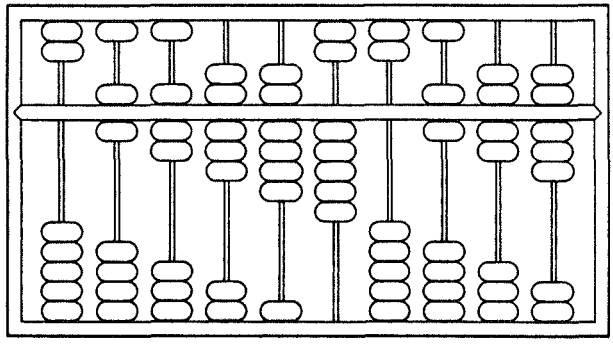
Simple *counting* is not the only way to compute. One can *measure* too. Surely early man must have noticed how the shadows of trees swung slowly around as the sun moved across the sky. The passage of time could be measured by the progress of shadows on such natural sundials. With no trains to catch, primitive people needed no refinements. Later, of course, early civilizations developed more accurate natural clocks. The arrangement of huge stones at Stonehenge, England, for example, told the ancient priests when Midsummer's Day, eclipses, and other key events of religious significance were about to occur. The basic idea

*The English words "calculate" and "calculus" are based on the Latin word *calculus*, which was a small pebble used in calculating, counting, etc.

behind sundials, modern clocks, slide rules, thermometers, and automobile speedometers is the measurement of some secondary quantity that mimics or simulates the thing we wish to know. Thus, distance around the rim of a sundial simulates time, and the height of mercury in a thermometer is the *analog* of temperature.

Two families of computers, the *analog* and *digital* machines, have evolved from the simple concepts of measuring and counting. Most people, however, mean "digital computer" when they say "computer", and this booklet will also concentrate on digital units. We should keep in mind, though, that for every large digital computer there are thousands of thermostats, planimeters, bathroom scales, and other small analog computers that "compute" one thing by measuring another.

No story of computers would be complete without mentioning the abacus. Here, the idea of counting is refined by using sliding beads rather than fingers. In contrast to the desk calculator (or "adding machine") that uses the *decimal*



The Chinese abacus (or suan pan) is a digital computing aid. Skilled operators using small sticks to slide the beans back and forth can keep pace with modern electromechanical desk calculators.

system and our *binary* electronic digital computers that count by twos, the abacus is a *biquinary* device (like two five-fingered hands), using five beads on one side of a wooden divider and two on the other side to count groups of fives. A fast abacus operator, sliding beads back and forth at "lightning speed" with a small stick, can sometimes beat a man skilled in using an ordinary desk calculator. Neither, however, can outrace an electronic digital computer, whose results are nearly instantaneous, or a calculating prodigy.

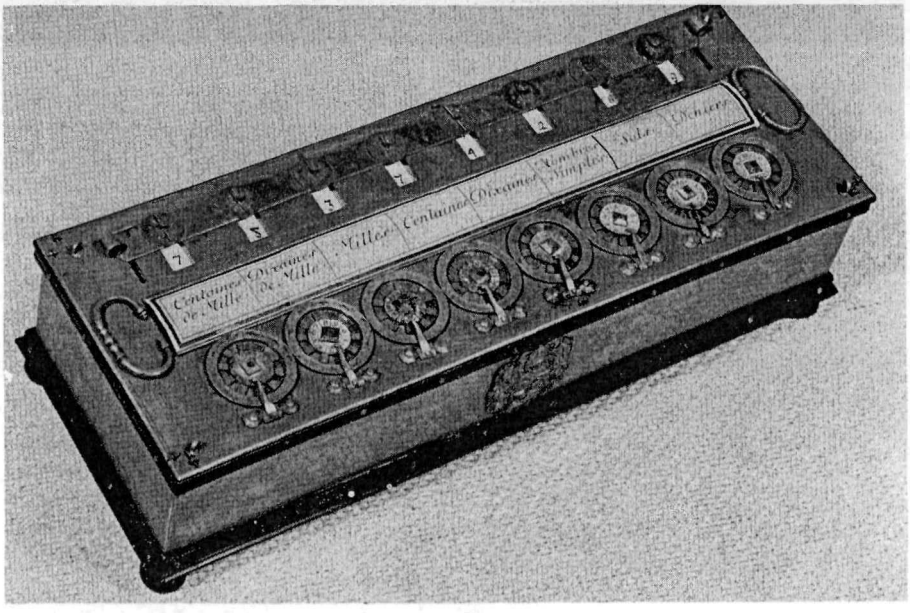
That Strange Genius Charles Babbage

Sometimes history seems to go awry and events happen before they logically should. For instance, Leonardo Da Vinci sketched helicopters and war machines long before they could be built. Charles Babbage was a similar ahead-of-his-time genius in computing. Born in Devonshire, the day after Christmas 1792, this banker's son conceived all the important features of modern digital computers; that is, the arithmetic (pronounced arith-MET-ic) unit, the memory, input and output schemes, and the most inventive concept of all, the idea of automatically sequencing instructions, by which a computer goes from one step to the next without human help.

Babbage was born into an England where mathematics, during the century since Sir Isaac Newton's death, had all but stagnated. The few who were literate could not figure sums accurately. Financial accounts were snarled, logarithm tables were full of errors, and insurance data were grotesquely jumbled. Babbage was incensed at this outrageous state of affairs and resolved to correct it—using computing machines instead of people.

Few computing machines were available. The French mathematician, Blaise Pascal, had devised and proudly displayed a simple mechanical computer in 1642, but it was little more advanced than the modern plastic "adding machines" with telephone-type dials we can now buy in "five-and-dime" stores. Machines can be built, surmised Babbage, that can add and subtract large numbers at high speeds. He set himself to the task.

Babbage conceived his "Difference Engine" in 1820 and completed it in 1822. It was specially designed to compute polynomials, like $x^2 + 2x + 63$, for the preparation of mathematical tables. It was essentially a collection of gears and levers, similar to today's desk calculators, accurate to six places. With this success, Babbage tried to construct a better Difference Engine, accurate to twenty places. He even talked the British government into contributing 17,000 pounds (an enormous sum in those days) to the project—probably because of the military value of a device to prepare good ballistic tables. This project, however, quickly became mired in manufacturing problems. The metal-working in-

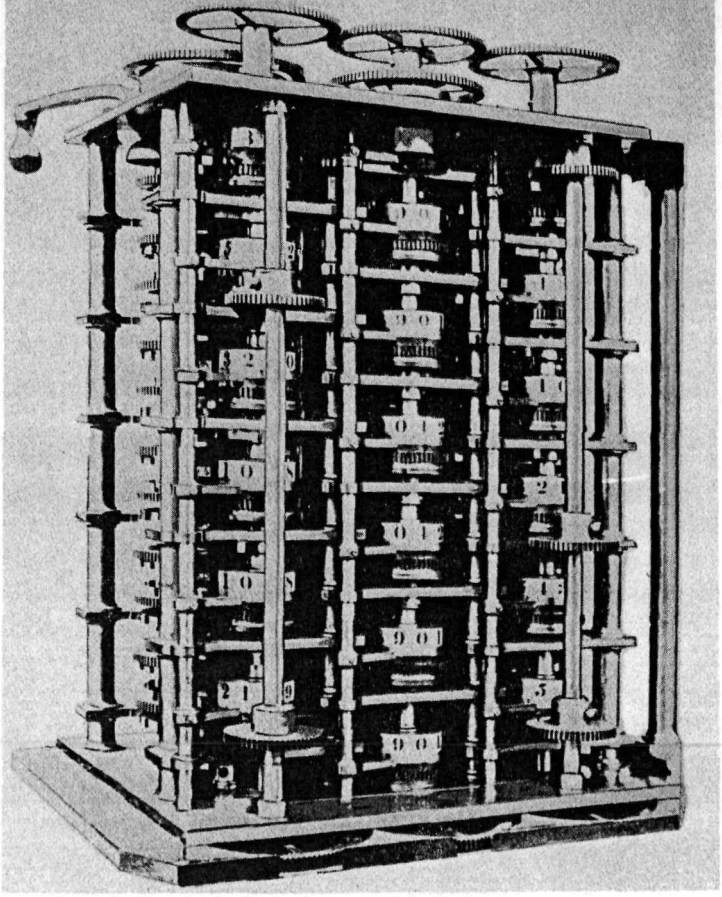


Blaise Pascal designed this mechanical computer in 1642. It is a digital, decimal machine, operated with a stylus. Numbers are carried to adjacent wheels by gears inside the machine.

dustry in the early 1800s could make smooth-bore cannons and good plowshares, but it wasn't competent for the precision gears and linkages described in Babbage's marvelously detailed drawings. So the project died, but not before Babbage had had a new dream and not before he had trained a few machinists to make metal parts with more precision and detail than the world had ever seen.*

Charles Babbage called his new dream the "Analytical Engine". It was designed to do all kinds of computations with the flexibility of a modern electronic computer. A vast assemblage of cogs, levers, and gears would be run by steam power (electricity was still a laboratory curiosity). The Analytical Engine's memory (or "store" as Babbage called it) was to consist of banks of wheels engraved with the ten digits. One thousand 50-digit numbers would be available upon demand by the "mill", where the arithmetic was done. Answers were to be automatically printed out just as computers do today. And, even more prophetic, the

*Some of these Babbage-trained men later founded machine-tool companies that made important contributions to England's industrial capability.



The "Difference Engine" of Charles Babbage. This digital, decimal computer was conceived in 1820. A small model was successfully built, but manufacturing problems prohibited the construction of larger machines.

Analytical Engine was to *control itself internally* by punched cards.

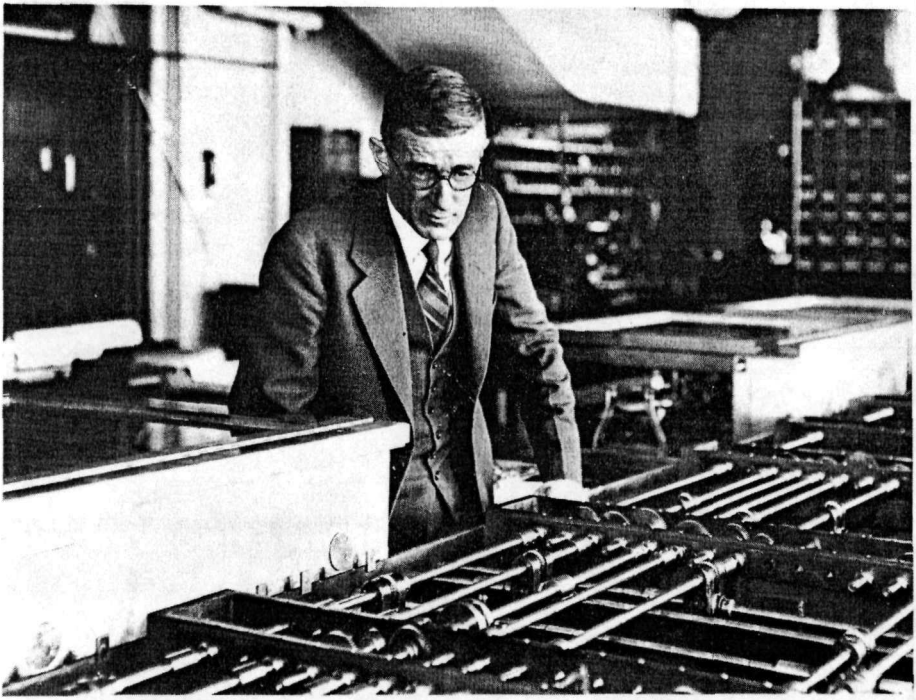
The punched card idea came from the mechanized loom of the Frenchman, Joseph Jacquard, in which punched cards controlled the pattern-weaving apparatus. Lady Lovelace, Lord Byron's daughter and the chief chronicler of Babbage's exploits, put it this way, "the Analytical Engine *weaves algebraic patterns* just as the Jacquard loom weaves flowers and leaves". It was all a beautiful idea, but it was born a hundred years too soon. Despite his ingenuity and the investment of his personal fortune, Babbage could not leapfrog the century of industrial development still ahead.

Babbage died in London in 1871, leaving a legacy we have yet to fully understand and measure. He wrote: "If, unwarned by my example, any man shall succeed in constructing an engine embodying in itself the whole of the executive department of mathematical analysis .... I have no fear of leaving my reputation in his charge, for he alone will be able to fully appreciate the nature of my efforts and the value of their results."

From Shell Trajectories to Weather Forecasting

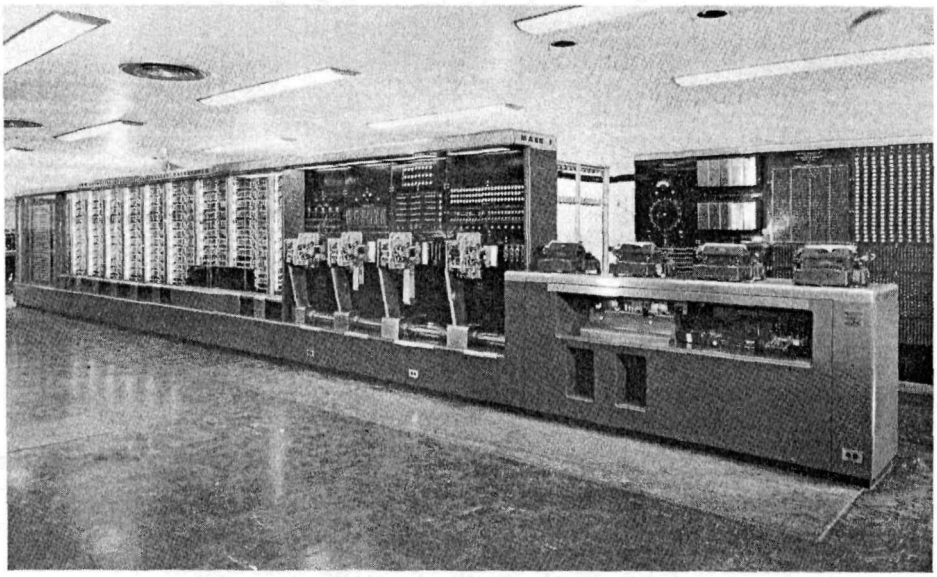
During World War II, the rumor was deliberately circulated that the large analog computer under construction at the Massachusetts Institute of Technology by Dr. Vannevar Bush and his associates had proven impossible to complete. Actually, the machine's motors and wheels were busy spinning out artillery firing tables. As in Babbage's time, much of the impetus for modern computer development was military in origin. Bush had started work on analog computers in 1925. The idea was to simulate a physical quantity,

Dr. Vannevar Bush shown with a portion of his Differential Analyzer. Built at MIT, the Differential Analyzer was the forerunner of modern analog computers.



like shell velocity, with an easily measurable analog, such as the angle through which a gear rotates, just as clock hands simulate time. By present standards, these early analog computers were slow, only about 100 times faster than a human operator using a desk calculator.

In the late 1930s, while Bush worked on his analog computers, only a few miles away a young Harvard graduate student named Howard Aiken was getting "fed up" with the tedious calculations required for his PhD. thesis. To speed his work up, he invented a series of small, very specialized, digital computers. He soon noticed that all his machines had common logical operations and many other similar features, such as memories and control units. In short, Aiken began plowing the same field Charles Babbage had tilled a century earlier. Aiken, however, was more fortunate than Babbage in having electrical gadgets, like relays, to help him. After three years of experiments, Aiken had his attention called to Babbage's pioneering efforts. Later, Aiken said that when he read Babbage's charge to his successors he felt that Babbage was talking directly to him from the past.



The Harvard Mark I digital computing machine. This early computer used relays to store and manipulate numbers.

In 1939, Aiken, with support from the International Business Machines Corporation, began work on a large-scale, digital computer called the Automatic Sequence Controlled Calculator. Charles Babbage would certainly have smiled when the first model, the Mark I, was placed in operation at Harvard in 1943. More ambitious machines followed.

Parallel to and independent of Aiken's work was that of George R. Stibitz at the Bell Telephone Laboratories. Both men took the critical step that makes modern computers possible, the introduction of automatic internal operation, by which a computer itself directs the course of the calculations once it is started. Both the Aiken and Stibitz computers could make decisions between alternative calculational routes—decisions that depended upon the results of preceding computations.*

When the Harvard Mark I was in operation, it sounded "like a roomful of ladies knitting", according to Jeremy Bernstein.† The muted clicking noises were made by thousands of electrical relays opening and closing. (Unlike most modern computers, the Mark I also included electromechanical counters, descendants of those designed by Pascal, however). The early relay computers were about ten times faster than desk calculators, and they could work for days without human intervention. They were slow by present standards because the iron relay contacts took whole milliseconds (thousandths of a second) to open and close, far too long for modern needs.

Fortunately, a much faster electrical switch was at hand: the vacuum tube. During World War II, at the Moore School of Engineering of the University of Pennsylvania, Drs. J. Presper Eckert and John W. Mauchly recognized the usefulness of the fact that a vacuum tube could be turned on and off in about a millionth of a second—thousands of times faster than the sluggish relay. By 1946, backed by the Army but too late for war use, Eckert and Mauchly had the ENIAC (Electronic Numerical Integrator and Calculator) ready. It was the first of the electronic digital computers

---

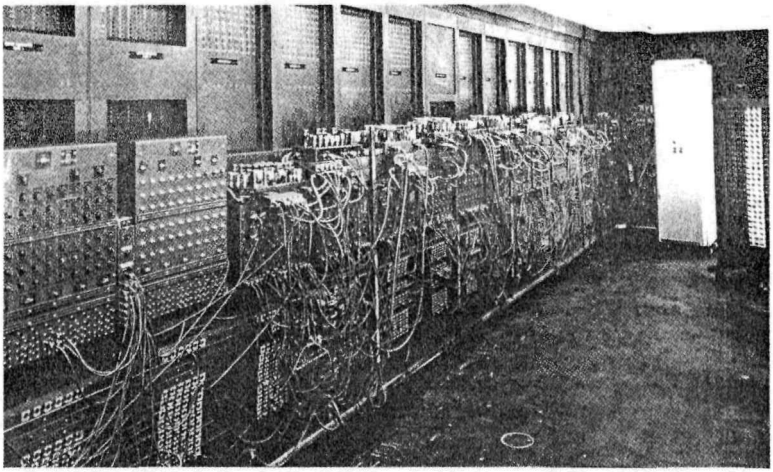
*A bank's computer does this when it looks at your balance and decides whether to send out an "overdrawn" notice.

†See Bernstein's excellent book, *The Analytical Engine*, listed in the Suggested References on page 54.

and the most complex electronic machine in the world at that time.

Dr. John von Neumann

During World War II, Dr. John von Neumann, who later served as a member of the Atomic Energy Commission, was a consultant both to the Army's Aberdeen Proving Ground, for whom Eckert and Mauchly were developing their electronic computer, and the AEC's Los Alamos Scientific Laboratory, which was attempting to compute the characteristics and effects of atomic bombs. In the latter capacity, von Neumann became impressed with the need for rapid, extensive, and repetitive computations, and, in his Army work, he learned about the ENIAC. The two together stimulated him to develop the idea of the internally



The first electronic digital computer was the ENIAC (Electronic Numerical Integrator and Calculator), which was built by the Moore School of Engineering at the University of Pennsylvania for the Army. It used 19,000 vacuum tubes and hundreds of thousands of other electrical parts.

stored program, where the step-by-step directions for computation, called *instructions*, are stored within the computer, and computations progress without need for external guidance. The idea of *instructions*, as distinct from *data*, to be stored in the computer memory was von Neumann's landmark idea. It became a reality when a computer using internally stored instructions was built at the Institute for Advanced Study at Princeton, New Jersey, in 1952.

The manufacture of electronic digital computers has proliferated into a multibillion-dollar business. The transistor has replaced the vacuum tube, and calculating speeds have risen by several factors of ten over that of the Mark I. The bookkeeping requirements of business have stimulated design of computers that range from desk-sized models to giants that fill several rooms.

A modern electronic digital computer. This specially adapted IBM 7090 fills an entire room at American Airlines' electronic reservations center at Briarcliff Manor, New York. All the ticket agents throughout the country are electrically connected to this computer. Reservation requests are processed almost instantaneously.

Even though computers can now multiply two 20-digit numbers together in millionths of a second, there are scientific problems that need still more speed. In meteorology, for example, more rapid forecasts could be made if weather data from ground stations and satellites around the

world could be quickly digested and inserted into the complex equations that describe the motion of cyclones and anticyclones. Faster computers are needed to compute what goes on in complicated chemical reactions and exploding atomic bombs. Some complex scientific problems involve literally billions of computing operations. Already, though, computers are approaching that ultimate limit imposed by the velocity of light, the maximum speed at which information can be transmitted. Since light travels about a foot in a billionth of a second, the distance between computer components results in a speed bottleneck.

An Automated, Computerized World?

You have a number! It may be a Social Security number, bank account number, or car registration number; but somewhere a computer is watching you. Whether figuring TV audience ratings, census statistics, income taxes, or bank balances, the electronic digital computer encroaches already on every citizen's private life. It is pertinent in this connection, to point out that computers can do much more than they do now. Before long, all banks, stores, employers, and governments will be tightly knit together by wires and radio. Machines will keep track of all items of commerce, all freight cars, all telephone calls, and all people. Many highways may be automated with computers controlling the passage of every car. There may be one universal credit card, perhaps bearing your Social Security number, and you might pay bills by inserting it in your home telephone and dialing the bank. This would be your "money key", entitling you to purchase anything anywhere. Should you lose your money key, you might "disappear"—financially speaking—from the face of the earth.*

Computing machines also have their humorous side, for they are not infallible. Some "lucky" people have received weekly paychecks of up to $9,000,000 (only to find, of course, that the check was uncashable). Machines in control of inventories have been known to order many times the world's supply of certain machine parts. One anecdote tells of an airliner of the future that has just become air-

*This might be a joke—or it might not.

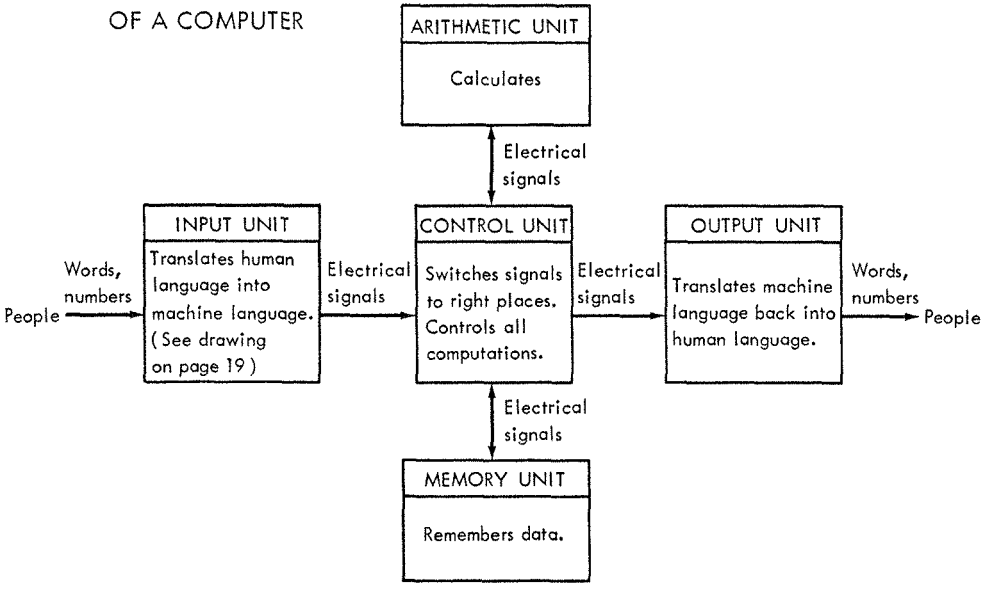
borne. The passengers are treated to this recording: "Ladies and gentlemen, this is a historic flight. For the first time, it has been possible to dispense with the pilot and copilot. A computer is operating this aircraft. It is in complete control . . . . complete control . . . . complete control . . . ." Perhaps the best feature man has built into every computer is the wall plug: It can always be pulled out!

ANATOMY OF A COMPUTER

The Generalized Digital Computer

All digital computers have five things in common. First, there is an *input unit* that digests human-produced information and instructions about what to do with this information. Its food is punched cards and tapes. Once in the computer, information and instructions are sent to the second component, the *memory*, to await recall by the computer *control unit*, which is the computer's taskmaster. The control unit makes sure that data are channeled to the right

BLOCK DIAGRAM OF A COMPUTER



places and that the fourth major component, the *arithmetic unit,* makes the right calculations. Finally, when the computations are complete, the *output unit* transforms the computer's electrical signals into written records that people can read.

Primer in Binary

The memory and arithmetic elements of an electronic digital computer are made so that they are either on or off, lit or unlit, closed or open, or magnetized clockwise or counterclockwise. In a sense, these *bi-stable* elements have only two fingers on which to do their counting. The two-valued way of counting is called the *binary* or *base-of-two* system, in contrast to the *decimal* or *base-of-ten* system we usually use. The number base used in arithmetic is arbitrary and always has been chosen for convenience. The ancient Babylonians used a sexagesimal, or base-of-sixty scheme, that is carried over to our clocks and angle measurements. Some mathematicians advocate that we shift to the somewhat more convenient duodecimal (base-of-twelve) system.

A computer therefore counts in the following manner:

Binary number	Computer memory*	Decimal number
0000	o o o o	0
0001	o o o x	1
0010	o o x o	2
0011	o o x x	3
0100	o x o o	4
0101	o x o x	5
0110	o x x o	6
0111	o x x x	7
1000	x o o o	8
1001	x o o x	9
1010	x o x o	10
1011	x o x x	11
1100	x x o o	12
1101	x x o x	13
1110	x x x o	14

*o = open relay; x = closed relay.

The binary system has blocks, or positions, arranged from right to left, worth 1, 2, and powers of 2 (such as 4, which is 2×2; 8, which is $2 \times 2 \times 2$; etc.). To build the number 3 from binary blocks requires one 1-block, and one 2-block, so it is written 11. To build binary 13 requires one 8 block, one 4 block, no 2-blocks, and one 1-block—written as 1101. The number 49 needs one 32, one 16, no 8, no 4, no 2, and one 1—written as 110001. In the yes-no language of electronic computers, when the binary system writes 100 as 1100100, it is really saying: "A block of 64, yes; a block of 32, yes; a block of 16, no; a block of 8, no; a block of 4, yes; a block of 2, no; a unit, no."

Addition in binary is simple. There are four basic rules:

$0 + 0 = 0$
$0 + 1 = 1$
$1 + 0 = 1$
$1 + 1 = 0$ plus a carry of 1.

When we add columns that generate a carry, we consider the carry a 1, and add it to the next column to the left, as we do with carries in decimal addition. Carries are indicated by the letter "c". A long column may generate more than one carry, in which case all of them are added to the next column.

So,

```
 cccc
  1101      or      13
 +1011            +11
 -----            ---
 11000             24
```

For efficiency, the five computer components must be able to converse with one another rapidly and in the same electrical language. In the electronic digital computer, pulses of electricity flow among the five units at nearly the speed of light. The pulses carry numbers and symbols in

---

Primer in Binary (Continued)

or:

```
 c  c
cccccc          1
  10111   or   23
 +10011       +19
 +11010       +26
1000100        68
```

Multiplication is hardly more difficult. The whole binary multiplication table can be reduced to one rule: "One times one is one; any other product is zero."

So,

```
  101   or    5
 × 11        × 3
  101         15
 101
 1111
```

Binary notation uses a "binary point" corresponding to the "decimal point" in decimal notation. Position values to the right of the binary point are successively halves, quarters, eighths, and so on. So the binary notation for the value "three and five-eighths" is 11.101.

The binary equivalent of any decimal number may be found by subtracting powers of 2 in the following fashion:

To find the binary equivalent of decimal 77, we say:

$77 - 2^6 = 77 - 64$ with a remainder of 13
$13 - 2^3 = 13 - 8$ with a remainder of 5
$5 - 2^2 = 5 - 4$ with a remainder of 1
$1 - 2^0 = 1 - 1$ with 0 remainder.

Thus, $77 = 1 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 1001101$.

PROBLEM 1

The binary number 101 actually is $1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 5$. What, then, is the decimal equivalent of 101.1?

PROBLEM 2

How can the decimal number 6.375 be expressed in binary? Build the decimal up from *negative* powers of 2.

Answers to problems are on page 53.

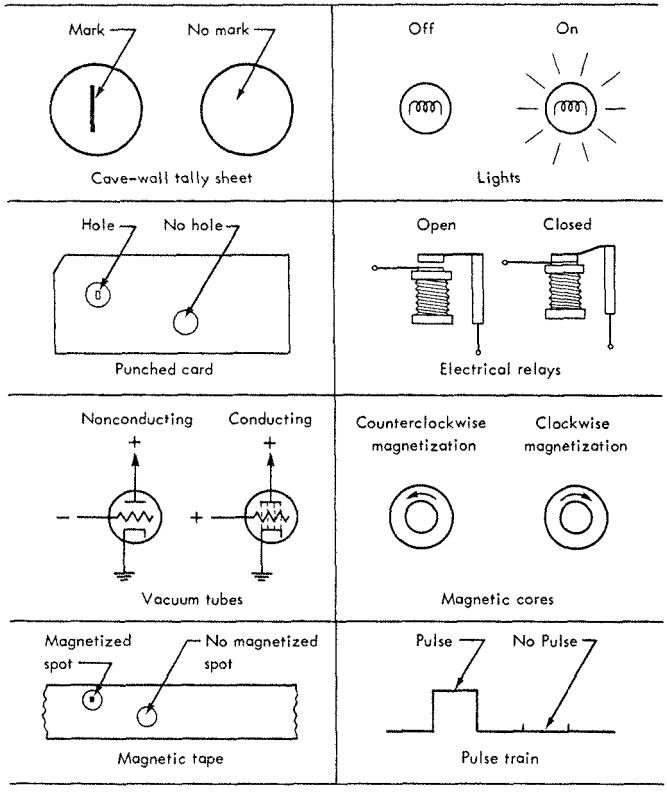
the binary language. A pulse signifies a 1 and its absence a 0. For example:

or 1011 (binary) = 11 (decimal)

The alphabet, punctuation, and mathematical symbols can be coded into similar pulse trains.

So far, only electricity has been mentioned as a message carrier. Actually it isn't necessary; digital computers need only two things:

1. Something that has two easily distinguished states, like relays (open or closed) and coins (heads up or tails up).

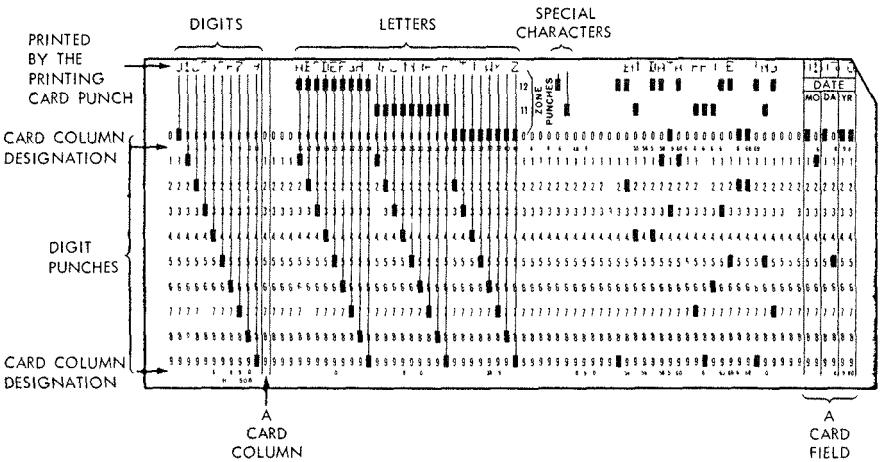


Two-valued or bi-stable *devices frequently used in computing. One condition is arbitrarily chosen as 0, the other as 1. These devices effectively "remember" numbers.*

2. A means for transmitting signals between them. Babbage used rods and cranks to transmit mechanical motion. Water and other fluids are used in some digital computers, but electricity is by far the most common medium of exchange.

The Input Unit

Computer problems always originate in someone's head. How are thoughts converted into the computer's electrical pulse trains, that is, from human language to machine language? For example if the decimal number 3 is to be inserted into the working innards of a digital computer, someone first has to write the numeral 3 on a printed form, which then goes to a key punch operator, who operates a machine that punches holes in a card or paper tape. The number 3 has now moved from a mental abstraction, to a written abstraction—the figure 3—to still another abstraction, a hole, whose position on a piece of paper can



A typical punched card. By using combinations of two holes, the decimal numbers, the alphabet, and several punctuation marks can be fed into computers and converted into electrical signals By using combinations of three holes in each column a great many mathematical symbols can be introduced Punched cards were first used for large-scale data processing by Herman Hollerith in the Census of 1890.

be sensed and turned into computer language: ___⎍⎍ = 0011. The computer now has something it can work with.*

In the long chain of actions involving writing, punching, and card reading, mistakes can be made anywhere—and they sometimes are, particularly by the human elements in the chain. It is at the input end of the computer where mistakes are most likely to occur, and also where information handling is slowest.

A computer's arithmetic unit can gobble up information—data and instructions—much faster than a punched card reader can feed it. To circumvent this possible bottleneck, modern computers first translate the data on punched cards

Table 1—TWO-VALUED DEVICES FOR DATA STORAGE

Device Description	Access Speed	Capacity	Permanence
Magnetic core, magnetic ring is magnetized clockwise or counterclockwise	High (5 μsec)*	Low	Permanent, erasable
Magnetic drum, or *disc*, spots on surface of rotating drum or record-like disc are magnetized or unmagnetized	Medium (10 msec)	High	Permanent, erasable
Magnetic tape, spots on surface of magnetic tape are magnetized or unmagnetized	Low	Unlimited	Permanent, erasable
Relay, open or closed	Low	Low	Nonpermanent, erasable
Punched card or *punched tape*, hole or no hole	Low	Unlimited	Permanent, nonerasable
Electrostatic, bright spot on TV tube or no spot	High	Low	Volatile, erasable
Delay line, sound pulses are reflected back and forth in a pipe	High	Very low	Volatile, erasable
Flip-flop, vacuum tube or transistor is conducting or nonconducting	Very high (5 μsec)	One bit	Volatile, erasable

*The abbreviation μsec indicates microseconds, millionths of a second. Similarly, msec means milliseconds, thousandths of a second.

and paper tapes into magnetized spots on a metal tape using an *off-line*† reading machine, while most of the computer parts, the *on-line* portion, work on other problems. The magnetized spots—moving past magnets—activate electric

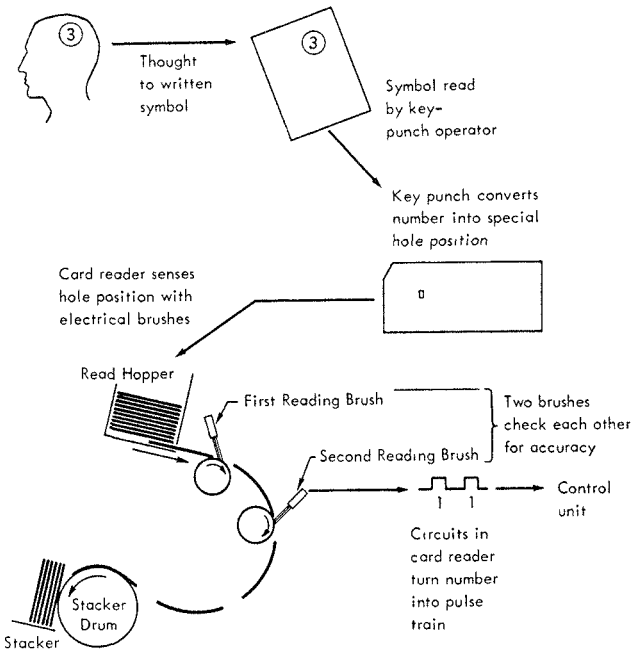
*It would be misleading and anthropomorphic to say that the computer "understands" cards, pulses, or anything else. It's just a machine that does what it's told.

†That is, a unit on a side circuit.

"readers". When the control and arithmetic units are ready for the next problem, the information is "read off" the much faster magnetic tape. Since computer time rents for as much as several hundred dollars an hour, time is truly golden, and ways to make savings like this are important.

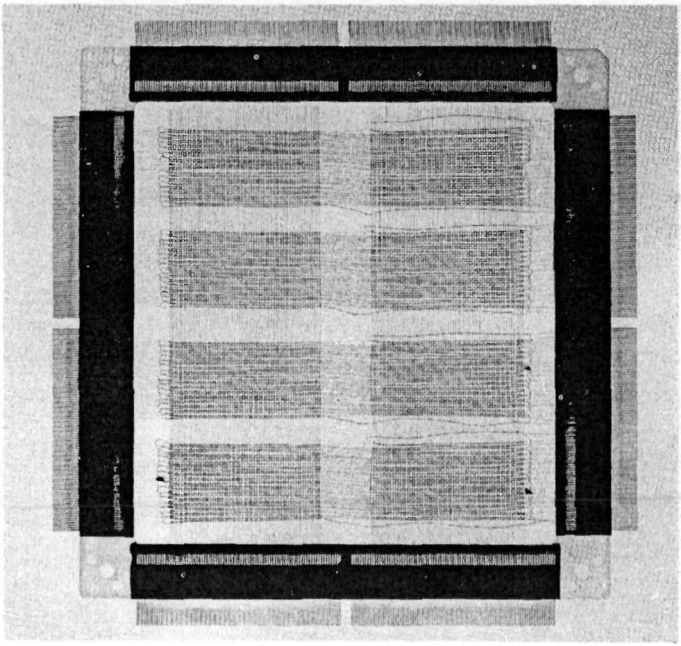
Magnetic Memories

A computer's arithmetic unit can complete a calculation in a few millionths of a second (microseconds) and be ready for the next instruction. This means that the unit must be fed data fast. In principle, a computer can search through a deck of punched cards or a few thousand feet of magnetic tape for the piece of data that it needs, say, your bank balance. But these methods are too slow. For real speed, data must be filed away with a specific "address", where it can be found and retrieved in a matter of microseconds. Table 1 on page 18 shows several on-or-off,



How a number is translated from human thought to the train of electrical pulses that the computer can use in calculations. Note the many different translations from one abstraction to another.

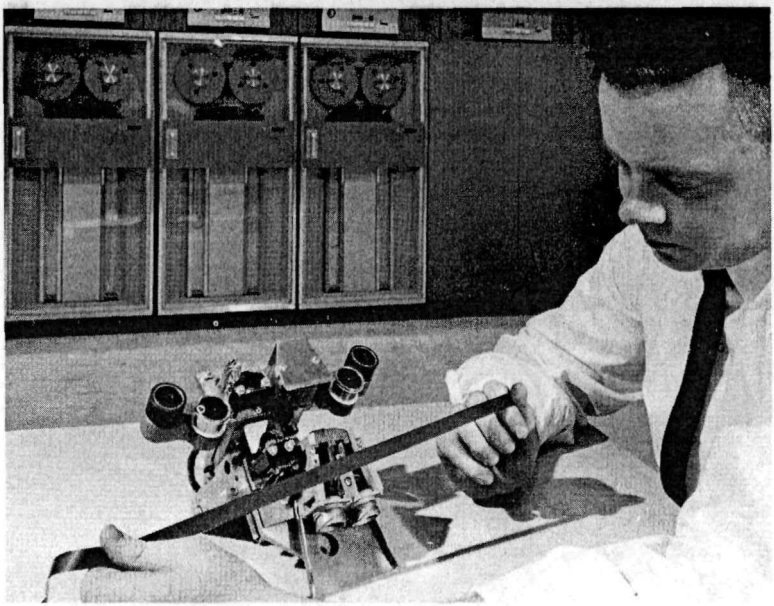
punched-or-unpunched, flipped-or-flopped, and other two-valued devices that can store binary information. For fast memories, today's computers use magnetic cores to store instructions and frequently used data. Magnetic drums are

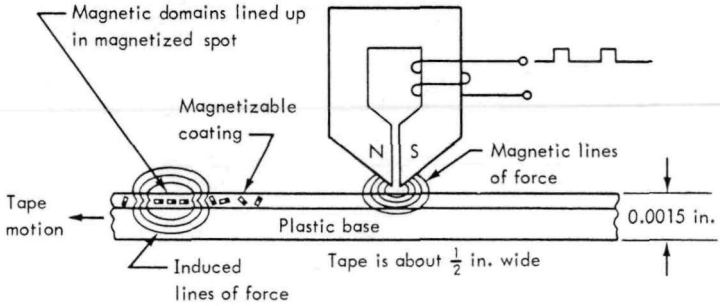


A section of the magnetic core memory of the IBM System/360 computer. This section is $6^1/_4 \times 6^1/_8$ inches and has 16,384 magnetic cores strung on the input-output wires. Each magnetic doughnut is .013 of an inch in inner diameter and .021 of an inch in outer diameter. Information bits can be extracted from any address in .00000075 second (three quarters of a microsecond).

employed where speed is not so critical, and magnetic tapes where speeds may be slower still. Generally, the faster a memory is, the smaller its capacity.

Magnetic drums and tapes store information in the same way: A magnetizable surface passes under a "writer" that magnetizes spots wherever 1's are supposed to occur. A "reader" can afterwards detect the spots and duplicate the original train of data pulses. Of course, the memory should be selective; the reading head must read the information only from the proper *address*. On magnetic drums,

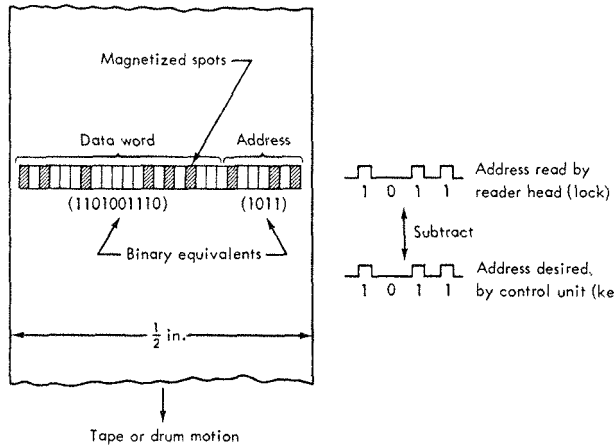




Most digital computers use magnetic tapes for storing large amounts of information. In the photograph, a piece of magnetic tape is shown with the reader-writer head. Three tape units are in the background. The sketch shows how electrical pulses applied to the windings of the writer head are transformed into magnetized spots on the tape.

for example, each piece of data is given an identifying number that is coded as another group of magnetic spots near the data it identifies. As the drum surface whips by (at 10 ft/sec or more) the addresses are read and compared with the desired address. When the correct address appears, the reading head is turned on and the proper data is sent to the computer's arithmetic unit—just like fitting an electrical key into an electrical lock. Searching for the right address obviously takes time. Magnetic drums are

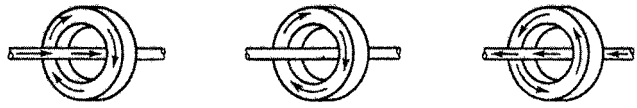
about a foot in diameter and can rotate all their data under the reader in a few milliseconds, but a long tape may take precious seconds to search from beginning to end, unless the data are stored in order, say alphabetically as for making out payroll checks.



"Key-in-lock" portrait-of-address searching with magnetic tape or drum. Address read from tape is subtracted from desired address. When result is zero, the key fits the lock and the reader head will read the associated data word and send it to the control unit for use in computations.

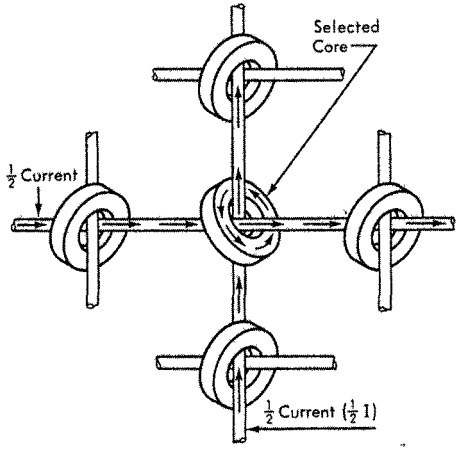
The fastest "memory" system uses magnetizable ferrite "doughnuts" each about half the size of the letter "o" on this page. These doughnuts can be magnetized in a clockwise direction, which we arbitrarily call 1, or counterclockwise, which is 0. Through each of the hundreds of thousands of doughnuts in a typical memory are run three or more separate wires that carry the magnetizing currents and take the stored data back to the arithmetic unit when the doughnut is interrogated.

Take the array of nine doughnuts shown in the figure on page 24. Suppose we wish to store a 1 in the center doughnut. It takes a certain minimum electrical current, I, to switch the doughnut's magnetic field to the desired clockwise direction. By sending currents equal to $\frac{1}{2}$ I down the two center wires, X_2 and Y_2, only the center doughnut, where the effects of the two currents add, will be magnetized in



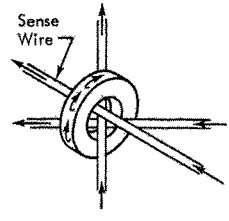
(a) (b) (c)

(a) Electrical current in wire produces circular magnetic field (Ampere's Law), which induces magnetic field in doughnut. (b) When current stops, doughnut remains magnetized (c) When current is reversed, direction of magnetization is reversed



A specific doughnut in a large array is magnetized by stringing the doughnuts on two wires and sending half the necessary current down two of the wires. Where the currents meet, their magnetic fields will add and magnetize the desired doughnut

A third wire, called a "sense" wire is used to sense the direction of magnetization when the magnetic doughnut is "interrogated"

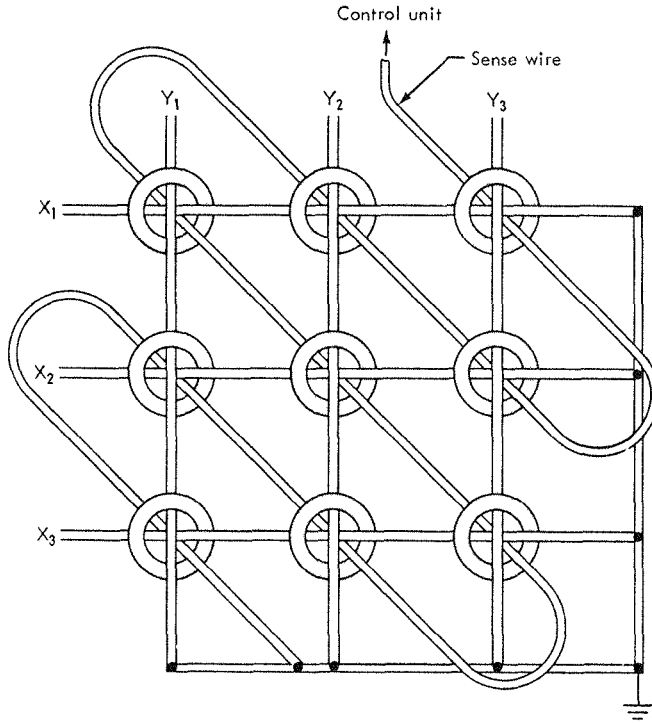


Magnetic cores are magnetized by currents in wires. Similar currents "interrogate" the cores to extract the stored information.

the clockwise state. The other eight cores will remain in their previous states.

Fast retrieval of data from the magnetic-core memory depends upon the *random-access* concept, where the computer can interrogate any doughnut amid that maze of thousands of doughnuts at will, without waiting for tape reels to unwind or drums to rotate. It works like this: Interrogation consists of sending electrical pulses to the address of the doughnut possessing the desired data. In the

illustration, the address of the center ring is X_2, Y_2. To get the 1 stored in X_2, Y_2 out, pulses equal to $-\frac{1}{2}$ I are sent down the wires, X_2 and Y_2. The pulses add to give $-I$ at the center doughnut, creating a counterclockwise magnetic field that reverses the direction of magnetization in the doughnut. The *reversal of the magnetic field* induces an electrical output pulse in the sense wire that is connected to the arithmetic unit. Of course, if the center doughnut contained



This detailed view of magnetic core shows how sense wire is strung through doughnuts on a frame Interrogation pulse of $-\frac{1}{2}I$ down X_2 and Y_2 will produce an output pulse in sense wire if there is a 1 in the center doughnut.

a 0, there would be no output pulse, because the doughnut would already be magnetized counterclockwise. Interrogating a magnetic core is "destructive" in the sense that all doughnuts queried are forced into the 0 state. Computers contain special circuits that regenerate the core memory after this destructive interrogation.

The important features of magnetic core memories are *rapid access*, *random access*, and the fact that information can be *stored permanently* without the need for continuous power.

Calculations Are Made in the Arithmetic Unit

After the information needed for the solution of a problem is translated from human thought into trains of electrical pulses and magnetized doughnuts, calculations can begin.

Suppose there are two decimal numbers, 3 and 2, in the magnetic core of an electronic digital computer and we wish to add them. The computer control unit, to be discussed later, will interpret the pulses associated with the instruction to ADD in such a way that the first number, 3, will emerge from the memory as the following string of pulses and no-pulses: _⎍⎍ = 011. This number will be temporarily stored in a register consisting of three fast-acting, flip-flop circuits. Call them A, B, and C, like this: $\overset{0}{\boxed{A}}$ $\overset{1}{\boxed{B}}$ $\overset{1}{\boxed{C}}$ The second number, 010 = 2, will then be called in from its address in the memory. Now the last signal in the train, a no-pulse, must be combined with the contents of flip-flop C in the register. The second signal, a pulse, combines with the information stored in flip-flop B, and so on, moving from right to left as in manual addition.

$$\begin{array}{r} 011 \\ +010 \\ \hline 101 \end{array} \quad \text{or} \quad \begin{array}{r} 3 \\ +2 \\ \hline 5 \end{array}$$

In other words, the computer uses this addition table:

0 bit* + 0 bit = 0 with no carry bit
0 bit + 1 bit = 1 with no carry bit
1 bit + 0 bit = 1 with no carry bit
1 bit + 1 bit = 0 with 1 carry bit

Certainly nothing could seem much simpler, but relatively complex electrical circuits are needed to carry out even those few operations. If we can solve the addition problem

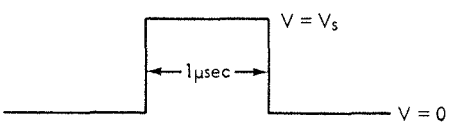
*A bit is a binary digit, 1 or 0.

The OR *Gate*

Three fundamental characteristics of electronic computers are illustrated by the operation of the two-legged OR gate

1. The use of sharp voltage pulses to transmit information
2. The use of the diode as an electronic switch
3. The electronic simulation of the OR logic, page 28.

The electrical pulse used in the following discussion consists of a steep voltage front that rises to its peak value, V_s, in nanoseconds (10^{-9} second). After about one microsecond (1 μsec = 10^{-6} second), the voltage drops rapidly back to zero. Such a pulse represents a binary 1 in this booklet.

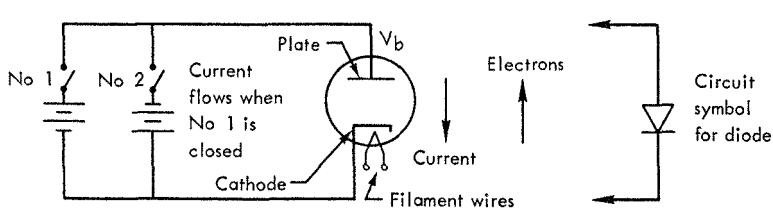


A diode is an electronic device in which current flows well in one direction but hardly at all in the other. The diode radio tube will serve as a good example.

When V_b is positive, electrons boiled off the cathode surface by the hot filament are attracted to the plate. Current flows *forward* through the diode and encounters little resistance. When V_b is negative, however, only a few electrons flow from the cold plate to the cathode; the reverse current is near zero. In effect, the diode has a high electrical resistance in the reverse direction and almost none in the forward direction.

The OR gate simulates OR logic (page 28) with two diodes connected in the way shown below. There are four possible states of the OR gate, corresponding to the four logical questions that can be asked in OR logic:

	The Logic			The Actual Signal		
	A	B	C	V_1	V_2	V_0
State #1	0	0	0	0	0	0
#2	1	1	1	V_s	V_s	V_s
#3	0	1	1	0	V_s	V_s
#4	1	0	1	V_s	0	V_s

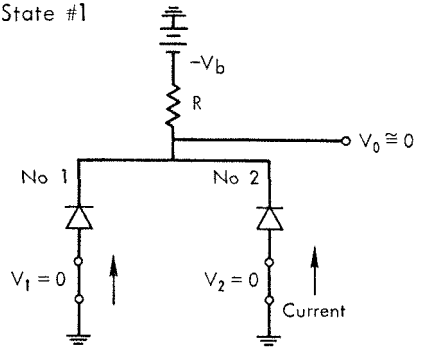


Electronics

The Logic States

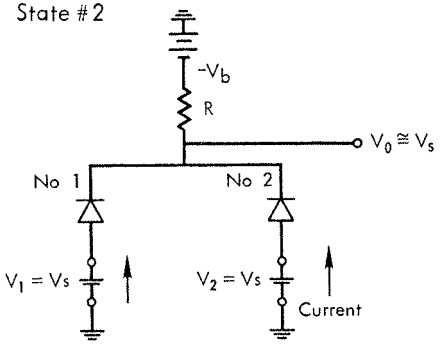
The logic states A, B, and C are represented electronically by the two input voltages, V_1 and V_2, and the output voltage, V_0 The voltage level, V_s, stands for 1, while its absence means 0 Here is how the circuit works

State #1



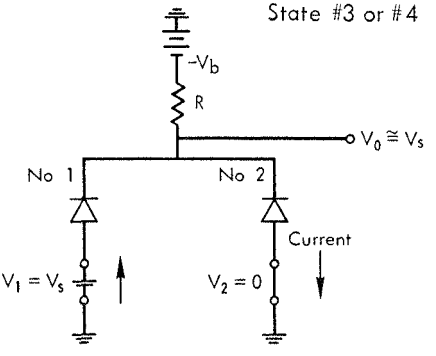
State #1 Input voltages, V_1 and V_2, are both zero Current readily flows through the diodes in the forward direction Since the diodes offer little electrical resistance, there is little voltage drop across them, and V_0 is very nearly 0

State #2



State #2 Input voltages, V_1 and V_2, are both V_s The current flowing through each diode is slightly higher than in State #1 because of the added voltage, V_s The voltage drop across the diodes is still very small though, and V_0 is just slightly less than V_s — it is so close to V_s that subsequent circuits will interpret it as V_s

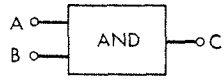
State #3 or #4



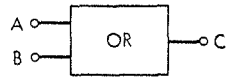
States #3 and #4 are identical except for the position of the input pulses. Taking the case where $V_1 = V_s$ and $V_2 = 0$, the voltage appearing at the top of diode No. 2 and at V_0 will be very slightly less than V_s, due to the almost negligible voltage drop across diode No. 1. The current flowing in diode No. 2, however, will be a *reverse* current because its top terminal is more positive than its bottom terminal. The reverse current is thousands of times less than that flowing through diode No. 1 in the forward direction, but the reverse resistance of diode No. 2 is correspondingly higher so that the voltage drop across diode No. 2 is the same as that across diode No. 1.

electrically, we can then subtract, multiply, and divide because, as we shall see, these operations are only variations of addition.

The entire addition table can be mimicked by combining three basic circuits, called *logical elements*, because their operation adheres to "logical" principles. None of the three logic circuits, AND, OR, and NOT shown here (there are others) can reproduce the addition table by itself, but connected together they can.

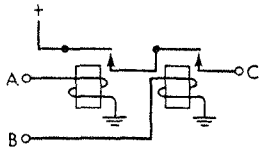


If A = 1 and B = 1, C = 1
Otherwise, C = 0.

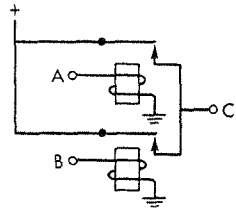


If A = 1 or B = 1, C = 1

If A = 1, C = 0
If A = 0, C = 1



Electrical pulses must be present at both A and B so that both relays will close and permit an output pulse to reach the C.



An electrical pulse at either A or B will close a relay and permit an output pulse to reach C

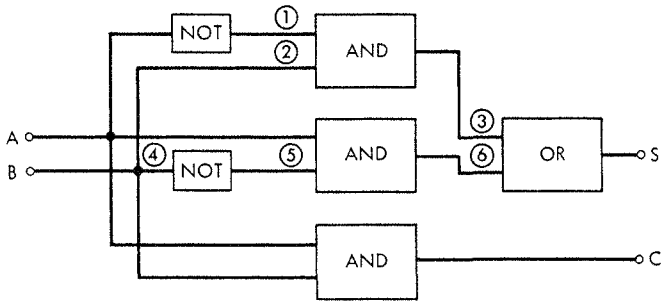
NOT logic circuits (sometimes called inverters) involve vacuum tubes or transistors Their descriptions may be found in the more advanced references given at the end of this booklet

By combining the three simple AND, OR, and NOT circuits, all the major functions of a digital computer may be constructed. Relays are used here for illustration, but transistors are used as switches in modern computers.

The combination of AND, OR, and NOT circuits shown is called a *half-adder*, because it can add only two of the many bits that make up two multibit numbers. The half-adder is one of the most important parts in any digital computer (see figure on page 29).

The half-adder can be constructed from any of many different electrical and mechanical devices, all possessing different speeds, costs, and reliabilities. The fundamental efficiency of the half-adder stems from the simplicity of two-valued (on-or-off, right-or-wrong) logic. Imagine the difficulty in trying to construct an electrical half-adder that

THE HALF-ADDER



A	B	1	2	3	4	5	6	S	C
0	0	1	0	0	0	1	0	0	0
0	1	1	1	1	1	0	0	1	0
1	0	0	0	0	0	1	1	1	0
1	1	0	1	0	1	0	0	0	1

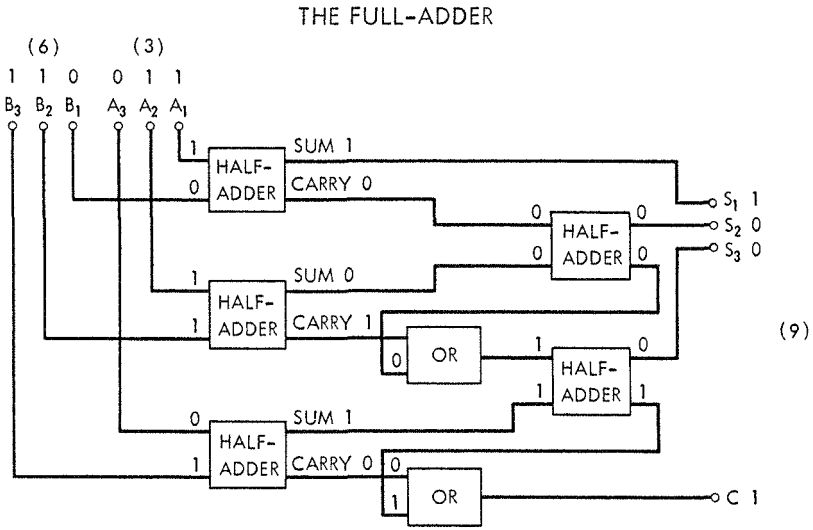
The all-important "half-adder" is constructed from AND, OR, and NOT circuits. In this combination two binary numbers can be added, giving the sum (S) and the carry bit (C) as specified by the addition table shown in the text. The 1's, of course, are represented by pulses in the circuits and the 0's by no-pulses. The reader should verify the table.

duplicates the decimal addition table with 100 entries instead of the 4 needed in binary.*

The next step is the *full-adder,* which is capable of adding numbers with more than one digit. The full-adder is really a series of half-adders connected together so that the carry bits are automatically transferred from one column to another, just as we carry 1's in addition with pencil and

*Decimal adders CAN be built (the common desk calculator is one), but mechanical versions are much too slow and electronic types too unreliable.

paper. The following figure shows three half-adders connected together to make a full-adder capable of handling three-bit binary numbers. A large digital computer may have 30 or more half-adders in each full-adder; the number used corresponds to the number of bits in each computer word.



A "full-adder" for three-digit binary numbers. The full-adder is constructed from half-adders and OR logic circuits. In the example, 6 and 3 are added to give 9. The reader should check through the function of each circuit More half-adders and OR circuits could be added below to add any size number desired. Even a simple full-adder contains many transistors and other electronic parts.

Addition succumbs to electronic circuitry, but what about subtraction? Computer subtraction is perhaps somewhat devious but turns out to be easy to mechanize. First, the train of pulses representing the number to be subtracted from the number in the ABC register is run through a NOT, or inverter, circuit. Every 1 in the pulse train is replaced by a 0, and vice versa. A mathematician would say that the NOT circuit forms the *1's complement* of the number. For example, the 1's complement of 010 is 101, a result also found by subtracting 010 from 111, whence the name, 1's complement. The next step in subtraction con-

sists of ADDING the 1's complement to the number in the ABC register. Finally, the number 001 is added to the result, and subtraction has been completed using only addition. Here is how it works:

$$\begin{array}{r} 3 \\ -2 \\ \hline 1 \end{array} \left\{ \begin{array}{l} \text{(Ordinary} \\ \text{decimal} \\ \text{subtraction)} \end{array} \right. \qquad \begin{array}{r} 011 \\ -010 \\ \hline 001 \end{array} \left\{ \begin{array}{l} \text{(Ordinary} \\ \text{binary} \\ \text{subtraction)} \end{array} \right. \qquad \begin{array}{r} 011 \\ +101 \\ \hline 000 \\ +001 \\ \hline 001 \end{array} \left\{ \begin{array}{l} \text{(Add} \\ \text{1's com-} \\ \text{plement)} \\ \text{Add 1} \end{array} \right.$$

(Note: $a - b = a + \overbrace{(111 - b)}^{\textit{1's complement}} + 1 = a - b + 111 + 001 = a - b.$)

Multiplication, of course, is just a shortcut way of doing repeated addition. Rather than instruct the computer in the intricacies of multiplication, we just design it to do repeated additions. It can add so fast that this is the simplest thing to do. Take the following example as illustrative:

$$\begin{array}{r} 7 \\ \times 5 \\ \hline 35 \end{array} \quad \text{or} \quad \begin{array}{r} 111 \\ \times 101 \\ \hline 111 \\ 111 \\ \hline 100011 = 35 \end{array}$$

All the computer really has to do is add-shift, add-shift, and so on. Shifting a number to the left turns out to be easy to do electronically, and the computer already knows how to add. Division can be accomplished by repeated subtraction in a similar fashion.

One other important computer operation is the comparison of two numbers to see which is larger or if they are equal. Subtraction of one from the other, which yields a +, –, or 0 answer, does this job nicely. Ability to compare numbers gives the computer a basis for making "judgments" and "decisions" whenever the factors involved can be reduced to numbers. An instruction placed in the computer's memory by the human operator can direct the computer to take one calculational path or another depending upon the answers of preceding calculations.

Much of the "intelligence" of computers, whether in chess games or war games, stems from this ability to compare and select among alternatives. The "brains" of a computer are therefore based on only the basic logic circuits we have introduced, and nothing more. Is the human brain also based on similar, simple ideas?

PROBLEM 3

What happens to the last carry bit when a full-adder adds 111 to 111?

PROBLEM 4

Subtract 25 from 31 by converting to binary, forming the 1's complement of 25, adding, and then adding 1. Do the same in decimal, but use the 9's complement of 25 (by subtracting it from 99) and add 1 to the result.

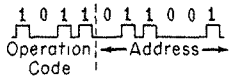
The Switchyard Magnificent—The Control Unit

So far, the electronic digital computer has been shown capable of digesting punched cards, storing the data in a memory, and manipulating numbers at high speed with a few logical circuits. How does a computer come to life once all its critical organs are in place? We find its life originates in the control unit, for here are created the timing pulses that set the pace of computer operations. Here, too, instructions are interpreted, addresses found, and events directed to happen throughout the computer.

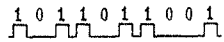
Computer *instructions* were described earlier as coded numbers that could be understood by the computer and turned into action. If instructions are just pulses, what is the difference between them and the real numbers used in the computations? There is, in fact, no physical difference and the two must be kept separate in the memory or the computer will be adding *instructions* or trying to carry out meaningless directions accidentally specified by *data.*

There are many formats for instructions, depending on the computer. The simplest is the single-address instruction. It consists of a long train of bits split into two parts. The first part is called the operation code, a small binary number (string of pulses) that the computer interprets as a

directive to add, subtract, or perform some other operation. The second part is the memory address of the number that is to be added, subtracted, or otherwise operated on. An instruction word might look like this:

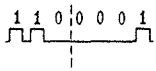
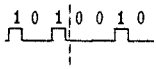
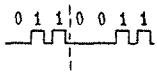


On the other hand, a data word looks like this:



Obviously, the two kinds of words can be distinguished from one another only by their source. One part of the memory must be reserved for instructions and another for data.

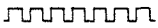
Most modern computers use the single-address instruction. The address may represent any memory location in the magnetic core or on drums. The operation code can tell the computer to perform any of the arithmetic functions as well as store numbers in the memory, shift them, stop calculations, print out answers, and so on. The use of the instruction is typified by the following addition sequence:

Pulse Portrait	Interpretation		Computer Actions
	LOAD	0001	The quantity *a*, previously placed at core address 0001, is recalled and placed in the arithmetic unit register.
	ADD	0002	The quantity *b*, at address 0002, is recalled and added to *a*.
	STORE	0003	The result, $a + b$, is stored at address 0003.

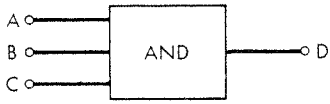
The computer is limited to a few dozen such basic instructions, but, as described later, simple operations can be combined to perform more complicated ones like extracting square roots and solving differential equations. In fact, the secret of electronic digital computers is in doing a few simple things millions of times a second and com-

bining these into more difficult operations, just as multiplication was accomplished by repeated addition.

Now that the role of the instruction has been explained, how, in the electronic sense, does the computer take the pulses that make up the operation code and turn them into, say, the addition operation? Or how is a word physically called in from a specific address in the memory? Two computer components work together to do this job. The first is the computer clock, the heartbeat of the machine. The clock, an electronic one, generates a continuous string of short square voltage pulses like so:



The computer pulse rate (1,000,000 cycles per sec in the case just mentioned) forces the rest of the computer to march in step with the help of special electronic circuits called AND-gates.* The clock sends pulses to all parts of the computer, but most are blocked by closed AND-gates. The only pulses that get through and stimulate action are those that arrive at the AND-gates *at the same time* as a pulse from the operation-code part of an instruction. Thus, the thousands of AND-gates in a large computer remind one of a huge switchyard, in which, by throwing the right switches, we can control and channel to the right spots the entire commerce of action-stimulating pulses. The selectivity of an array of AND-gates is revealed in the cir-



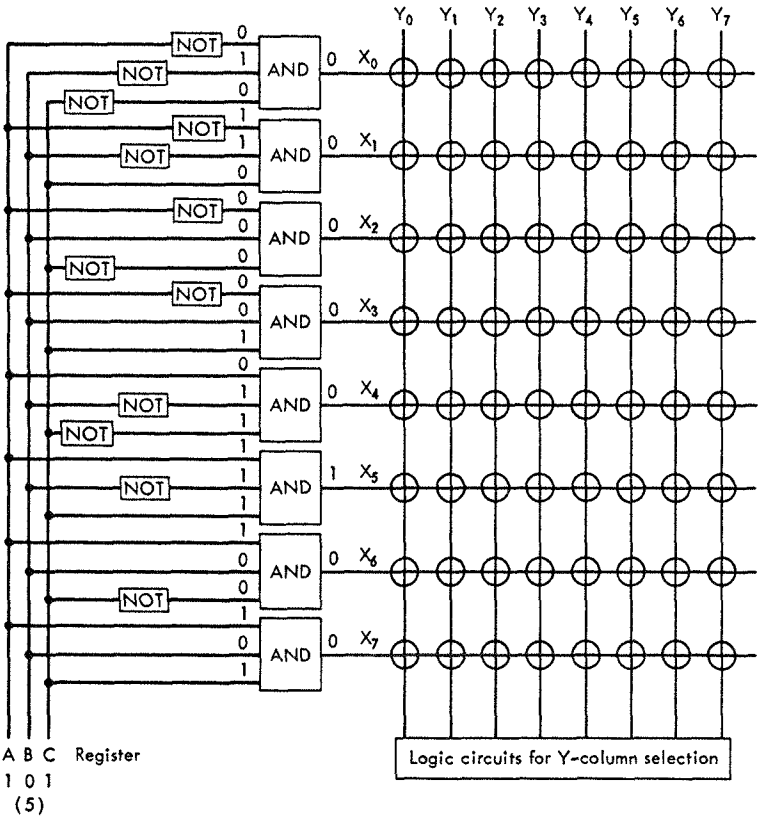
If A = 1 and B = 1 and C = 1, D = 1.
Otherwise D = 0

A 3-legged AND-gate. This frequently used computer circuit requires three simultaneous pulses at its input before it will produce a pulse at its output. See how this circuit is used in finding magnetic-core addresses in the next figure.

---

*This kind of computer is termed "synchronous". Some computers are not regulated by clocks and are called "asynchronous". In addition, by use of overlapping or paralleling circuits, computers can be made to perform many such operation sequences simultaneously. (See page 47.)

cuits of the next figure that shows how a particular address in a magnetic core memory is selected by the address part of an instruction.



These address-finding circuits are typical of digital computers in which complex functions are carried out by means of many simple, interconnected circuits. In the example, the sixth horizontal string of cores from the top is selected by pulses representing 101 or 5 from the address part of an instruction in the control unit register. The arrangement of NOT and three-legged AND-gates acts like a sieve that permits only the correct magnetic doughnuts to be pulsed. Note that the NOT gates invert the incoming pulses so that only in the X_5 circuit does the signal 101 reach the AND-gate.

Getting Data Out of a Computer

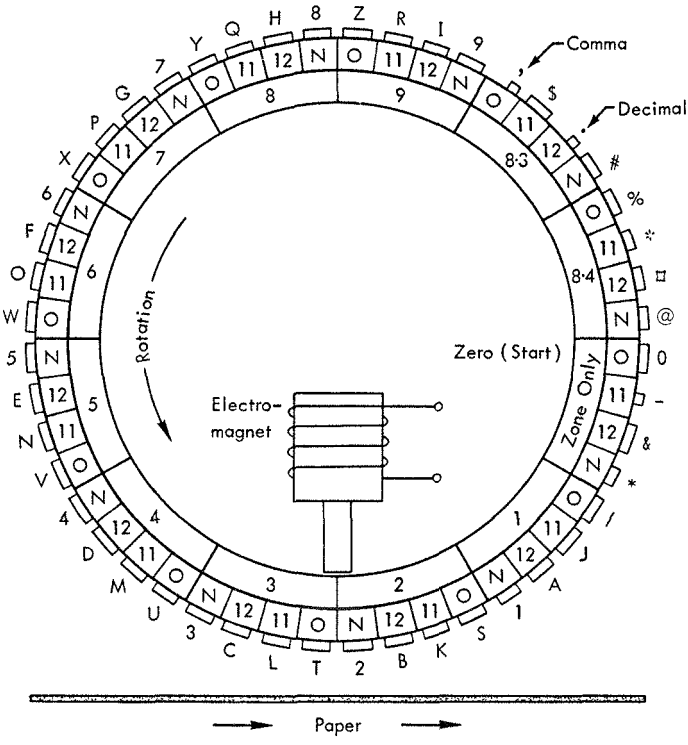
A very common computer instruction is PRINT or PUNCH. When the control unit receives the coded pulses

corresponding to the PRINT instruction, it opens AND-gates that release the pulses representing data stored in the memory location specified by the address portion of the instruction. The pulses are channelled to a printer where they are converted back into decimal and alphabetical characters and printed—usually on equipment something like a high-speed typewriter. If the data are to be used in later calculations, an output instruction may be WRITE and, if a magnetic tape is addressed, the data are rendered as patterns of magnetized spots. Getting data out of a computer involves the same problems as getting it in, only in reverse order.

Output equipment must operate rapidly, but no matter how fast printers print and punchers punch new data cards they cannot keep up with the arithmetic unit, where the pulses of almost inertialess electrons do the actual computing. Rather than slow down the arithmetic unit to match the printer, output data are first written on magnetic tape. Later the tape is "read" by a printer at its own speed while the rest of the computer works on other problems. A similar strategem was used, you remember, to prevent the relatively slow card reader from causing a data log jam at the computer input point.

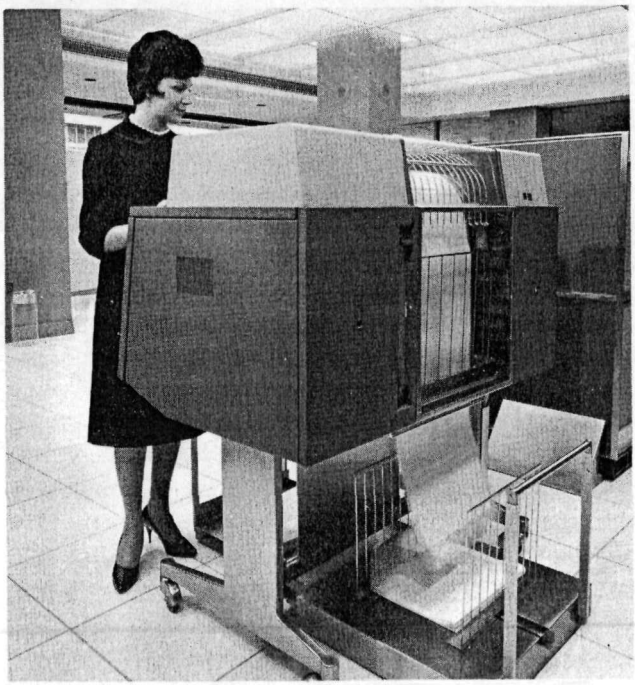
Calling a computer printer slow is correct only in a relative sense, for modern printers can spew out a stream of printed paper far faster than the eye can read it. In fact, the words on this page could be printed in less than two seconds. A web of paper flies through a printer, receiving numbers and letters at a speed of about 20 lines of 120 characters each per second. Anyone who has used typewriters can testify that even the best machines are incapable of typing 2400 characters each second. A computer printer gets around this by printing whole lines at a time. In a typical printer, at each of the 120 locations where a character may be printed, there is a wheel of type that rotates at about 20 revolutions per second. Each wheel contains all letters, numbers, punctuation marks, and mathematical symbols commonly used in business and science. All 120 wheels rotate together. Inside each wheel, there is a hammer driven by a stationary, fast-acting electromagnet. When the rim of the type wheel has moved so that the

letter or number that has been dictated by the computer is between the wheel and the paper, the hammer comes down on the type wheel rim, printing the proper character on the paper.



A high-speed printer wheel. The electromagnet is pulsed when the desired character passes over the paper. A typical wheel rotates 20 times each second. There is one wheel for each of the 100 or more characters in each line. Endless type chains (running horizontally) are sometimes used instead of type wheels.

Because a whole line is printed in $\frac{1}{20}$th of a second, the motion is too fast to follow with the eye. Reflection tells us that, because the type wheels are all synchronized, all the 1's in a line will be printed first, regardless of their position, a split second later the 2's, and so on. The paper web, of course, stops its motion during the instant the type wheels are being actuated, though this cessation of motion is imperceptible. At the end of a one-line printing cycle,



A high-speed computer printer. The model shown can print 10 lines of characters every second. Note how the web of paper emerging at the bottom is blurred by speed.

the paper is automatically moved ahead a line. You cannot see the many thousands of AND-gates opening and closing or the register circuits flipping and flopping in a computer, but you cannot fail to be impressed by its high-speed printer.

How to Talk to a Computer

Inside the electronic digital computer, all conversations are in terms of voltage pulses and the language is binary. Humans don't talk or think in these terms, so an input unit is provided to translate human commands into computer actions. Earlier (page 33), a simple addition problem was traced in which the ADD instruction was presented to the computer on a punched card. The computer then converted the card holes into voltage pulses that opened the proper AND-gates, recalling data from the memory and forcing the

arithmetic unit to add the numbers specified. The instructions for simple addition had to be spelled out meticulously, you recall. In fact, the detail in such a *machine-language* program is much too great for lazy people (a category that includes most people who use computers). We would like to be able merely to write down the equation of interest, say $y = \sin(ax + b)$,* turn it over to a card puncher, and let the computer itself translate the mathematical symbols into the long series of computer instructions needed to direct the indicated mathematical operations. Unfortunately, a computer cannot think in terms of sines and cosines without help. But, if we speak very precisely and use an interpreter, we CAN save ourselves the labor of writing out endless columns of machine-language instructions.

The fundamental idea is this: Invent a language, with a very simple grammar, using English words and universal mathematical symbols that can be translated by the computer itself into machine language through the use of a special translation program (the interpreter).

To illustrate, take $y = \sin(ax + b)$. In machine language, the *program* (or series of instructions) might read like this:

LOAD	0001	The quantity x, previously placed in memory address 0001, is recalled and placed in the arithmetic unit register.
MULT	0011	The quantity a is recalled from address 0011 and multiplied by x.
ADD	0012	The quantity b is recalled and added to ax, to form $ax + b$.

So far, not too hard, but now the lazy man's impasse: Many mathematical functions may be approximated by an *infinite series* of algebraic terms. For example, $\sin(ax + b) = (ax + b) - (ax + b)^3/3! + (ax + b)^5/5! - \ldots$ †, and so on for as many terms of the sine infinite series as we need

*This example will be most meaningful to those who have recently studied trigonometry, but anyone should understand the principle.

†The symbol ! calls for a factorial procedure, so that 3! means $1 \times 2 \times 3$.

for accuracy, perhaps as many as ten. Take the machine language program for just the second term, continuing from above:

STORE	0013	$(ax + b)$ is stored in address 0013 but also remains in the register.
MULT	0013	$(ax + b)^2$ is formed in the register.
MULT	0013	$(ax + b)^3$ is formed in the register.
STORE	0014	$(ax + b)^3$ is placed in the memory at address 0014 for safe keeping.
ONE	0015	The number 1 is stored at address 0015.
ONE	0016	The number 1 is stored at address 0016. Note that the computer will generate a 1 if it is needed, but not other numbers.
LOAD	0015	1 is placed in the register, replacing $(ax + b)$.
ADD	0016	$1 + 1 = 2$ in the register.
STORE	0017	2 is stored at address 0017 but also remains in the register.
ADD	0016	$2 + 1 = 3$ in the register.
MULT	0017	$3 \times 2 = 6 = 3!$.
STORE	0018	$3!$ is stored at address 0018.
LOAD	0014	$(ax + b)^3$ is placed in the register.
DIV	0018	$(ax + b)^3/3!$ —the second term has now been built up.

To avoid reducing ourselves to repeating the above machine-language baby talk, we tell the computer that when we punch SINE on a card, it should look in its memory for the program for calculating the sine by the above infinite series. In "computerese", the machine refers to the sine *subroutine,* which, of course, we have previously placed in its memory as part of the special translation program. Special subroutines have been developed for many complicated mathematical operations. The translation program will have a library of them.

Various sophisticated languages have been created for talking to computers in almost-plain English. The foremost of these is FORTRAN, for FORmula TRANslation. COBOL, COmmon Business Oriented Language, and ALGOL, ALGOrithmic Language, are others. Using a language like FORTRAN, the computer user can rely upon the machine to select the proper subroutines, choose appropriate memory locations for input data and intermediate results. The new language, though more abstract than machine language, makes it much easier to converse with computers.

One valuable characteristic of the electronic digital computer is its ability to perform repetitive computations rapidly in what is called a *loop*. Suppose that we wish to evaluate $y = \sin(ax + b)$ for 50 different successive values of x. When the computation is finished for the first value of x, an instruction tells the computer to go back to the first instruction and repeat the calculation, but this time with the address of x increased by 1. Of course, the computer would go through this loop an infinite number of times unless we also insert a few instructions telling the computer to subtract the x address just used from the address of the 50th x.* When the subtraction yields zero the computer can be instructed to get out of the loop. In all computer programming, every last detail must be written down in unequivocal fashion. The computer cannot be presumed to know anything or exercise any judgment unless we put everything it needs to know there in the first place.

PROBLEM 5 Write a machine-language program for $y = x^4 + 2$.

*Here is a case where we *want* to treat an instruction as if it were a number. See page 39 for a typical instruction format.

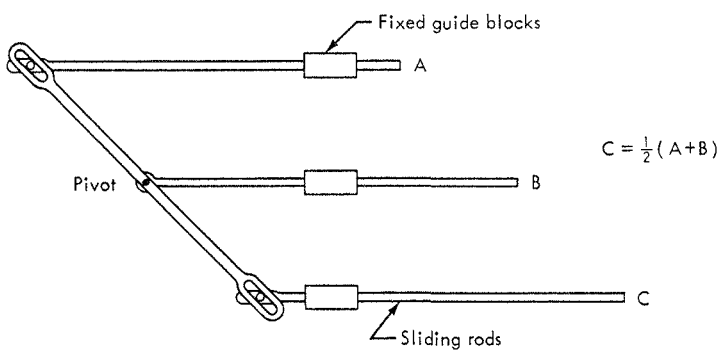
ANALOG COMPUTERS

Analogs Measure Rather Than Count

It is worthwhile repeating the assertion that analog computers *measure* rather than *count*, because therein lie all the differences between the two great classes of computing machines. Analogs copy nature by substituting an easily measurable quantity, like the amount a shaft has rotated, for a difficult-to-measure property like missile velocity. In this fashion, rockets can be "flown" and submarines taken on "trial cruises" before they leave the drawing boards. In other words, analog computers act out or *simulate* nature on small, convenient, and inexpensive stages.

Most analogs can be placed in three classes:

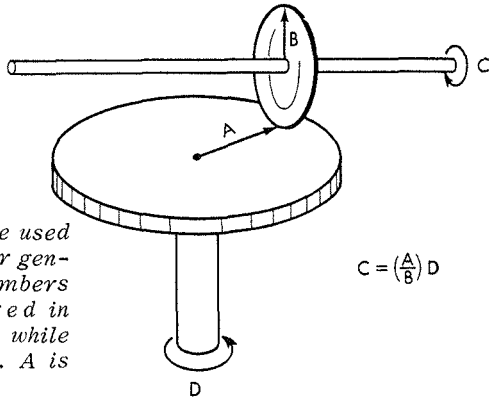
1. Those that simulate numbers
2. Those that simulate physical phenomena
3. Scale models of nature



Mechanical linkage for adding two numbers. The numbers are determined by measuring the distances moved by the right-hand ends of the horizontal bars.

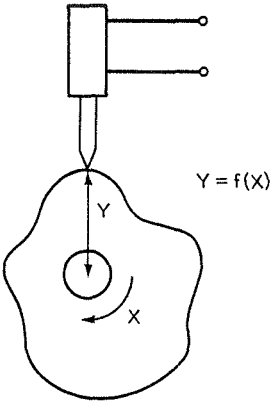
Numbers turn out to be easy to simulate; the slide rule, where distance represents the logarithm table, is an obvious example. Thus, by adding and subtracting distances, numbers are multiplied, divided, and so on. Addition cannot be done on conventional slide rules, but other analogs, such as ordinary rulers, do it easily. Multiplication can be carried out by the common disk and wheel sketched in the next figure.

Electrical and mechanical gadgets can be assembled to simulate just about any mathematical operation desired.



Wheel-and-disk device used by analog computers for general multiplication. Numbers C and D are measured in terms of shaft rotation, while A and B are distances. A is variable.

The answer to problems posed will appear as a voltage level, the angle through which a shaft has turned, or some other physical property. At the computer output, these measurements are converted into numbers. Accuracy is low, typically 1 part in 1000, because the output can be



This two-dimensional cam represents some complicated mathematical function. As it turns, the spring-loaded stylus converts cam radius measurements into an electrical signal. Three-dimensional cams are often used in analogs.

measured with only this much accuracy. In addition, the proportionality, say between number and voltage, may vary or drift during the computation.*

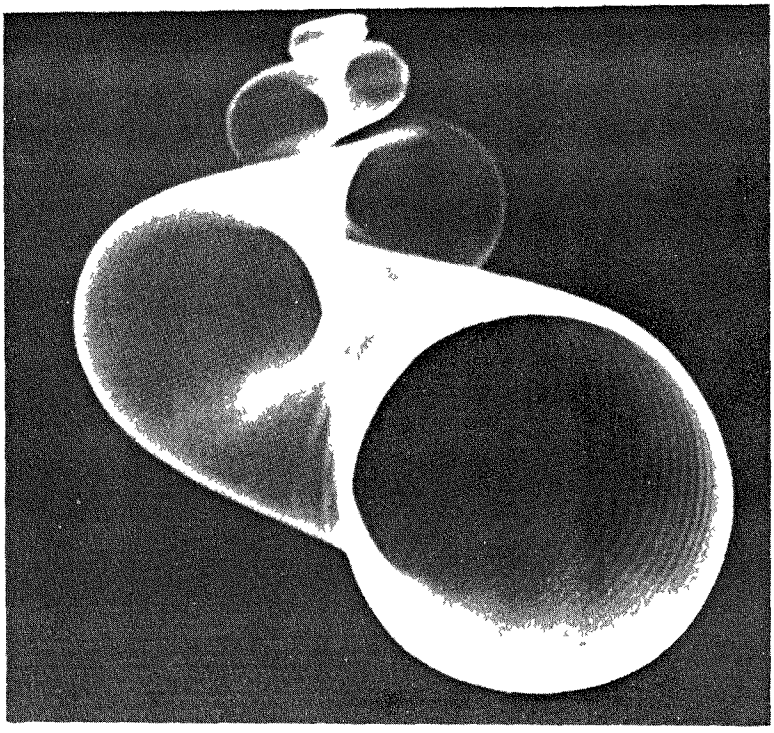
---

*This attrition of accuracy does not occur on digital computers where a 1 is always 1 and not 0.95 or 1.05.

In analog computers, it is particularly convenient to simulate complicated mathematical functions with cams, specially wound potentiometers, and similar devices. Where the digital computer has to compute the value of sin (ax + b) for each x, the analog machine can get the whole sine table from a shaped cam. Analog computers are particularly well adapted to solving differential equations.

The mental picture of the analog computer that may be formed is one of a large mass of gears, shafts, cams, and circuits, each simulating some natural quantity, and all hooked together in such a way that a specific equation is solved or some particular physical phenomenon is acted out. Inherent in this picture are relative slowness and low accuracy. Lack of flexibility is also implied because, to solve a new problem, the whole analog usually has to be rearranged or rewired. The real value of the analog is in solving highly specialized and repetitive problems over and over again. The small analog computer in an aircraft's autopilot that repeatedly solves the same steering equations shows analogs at their most typical and best. In situations such as this they are much faster and more convenient than digital computers.

In the second class of analogs, physical events that are difficult to describe mathematically are simulated. One typical analog in this class uses electricity-conducting paper. The conduction of electricity in the paper closely simulates the conduction of heat, the flight of neutrons in nuclear reactors, or any of several other physical phenomena that are hard to study by equations alone. Simple heat conduction along a metal bar may be easy to describe mathematically, but when there are bars with complex shapes and several sources of heat, equations are not easy to develop. With a conducting-paper analog, though, any shape can be quickly cut out with scissors. Heat sources are simulated by wires that inject electricity into the paper. By measuring voltages at points of interest, one effectively determines temperature. Furthermore, new shapes can be cut quickly and heat sources modified. Sometimes the physical problem is simulated by networks of interconnected resistors, which replace the natural resistance of the paper.



A "computer snail" Photo of an oscilloscope trace generated by an analog computer programmed to simulate the spiral growth common in mollusks and other invertebrates. By varying the inputs all possible shell forms, including those that do not occur naturally, could be generated.

Similar problems can be solved in three dimensions by using a tank filled with a conducting liquid.

Finally, any scale model is an analog computer in a sense. Here, however, only size is changed, while most other physical factors remain the same. Wind tunnels and model boat basins are examples that immediately come to mind. Many aircraft "bugs" are discovered in wind tunnels and eliminated before expensive full-scale models are built. Wind velocities may duplicate those expected in flight; the major difference is in the reduced size of the aircraft. Of course, some factors, such as wing flutter, do not scale down well when physical size is reduced; but with care scale models can be made to yield much valuable information. Digital computers are not helpful in problems like this because the problems are too complex and cannot easily be stated in equation form.

Over $4 billion is spent each year in the purchase and rental of computing machinery. Yet, less than 10% of this money is spent on analog computers. In the following table, we see some of the reasons for this unequal division.

Table 2—COMPARISON OF DIGITAL AND ANALOG COMPUTERS

Characteristics	Digital computers	Analog computers
Cost	High	Low
Accuracy	High, limited by word length to approximately one part in 10^{12}.	Low (one part in 10^3 to 10^4)
Speed	Very fast, because only electronic circuits are involved.	Slow in arithmetic calculations, because slow-moving mechanical parts are often used. Can be fast in specialized problems.
Flexibility	Excellent. Can deal with any problem that can be handled by logic. Letters and numbers can be used.	Specialized. Parts have to be rewired or changed for different problems.
Mode of operation	Performs arithmetic and logical operations by counting discrete pulses.	Simulates nature with electrical and mechanical parameters.

We see from this table that digital computers are readily adapted to handling discrete, countable things like numbers, inventory items, or the hours worked by an employee. On the other hand, analog computers cannot count, though they can add, subtract, multiply, and do a few other procedures with limited accuracy. The big market advantage possessed by digital computers is their usefulness in business operations. They can handle payrolls, inventories, seat reservations, and keep track of millions of details that formerly degraded humans to mere AND-gates and switching circuits. They can monitor and control processes. Furthermore, digital computers can solve many scientific and engineering problems better than analogs. Hence, they are the big sellers.

PROBLEM 6

Show how two gears can multiply any number by three.

PROBLEM 7

Design a slide rule that can add.

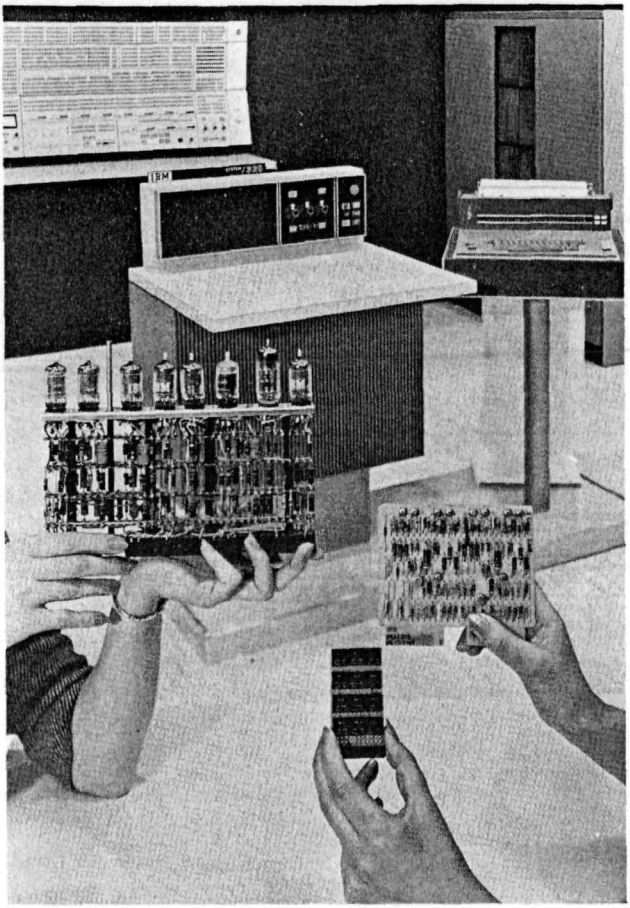
SMALLER AND SMARTER COMPUTERS

As electronic digital computers have progressed from electromechanical relays to vacuum tubes to transistors, their sizes have shrunk while their speeds and memory capacities have grown. Will the next 20 years see advances comparable to those of the last two decades of the computer age?

Looking first at speed, we recall that an initial bottleneck involved the slow input-output equipment, which, because it is primarily mechanical in nature, could not keep up with the electronic portions of the computer. Off-line printers and card readers solved this problem. Computers, however, still do not calculate fast enough for some major scientific problems like computing atmospheric circulation. How can things be speeded up?

Since the speed of computer signals cannot exceed the velocity of light, there is an upper limit on computer speed that we must somehow circumvent. First, notice that the speed-of-light bottleneck is especially serious when there is a long series of operations involving the sending of thousands of signals *serially* between computer components. Why do everything step after step? Why not compute things *in parallel,* where convenient and possible? In the evaluation of $sin\ (ax + b)$ by infinite series, for example, the numerator and denominator of each fraction could be computed simultaneously by separate arithmetic units since they do not depend upon one another. This use of several arithmetic units in parallel seems certain to increase computer speeds a great deal in the future.

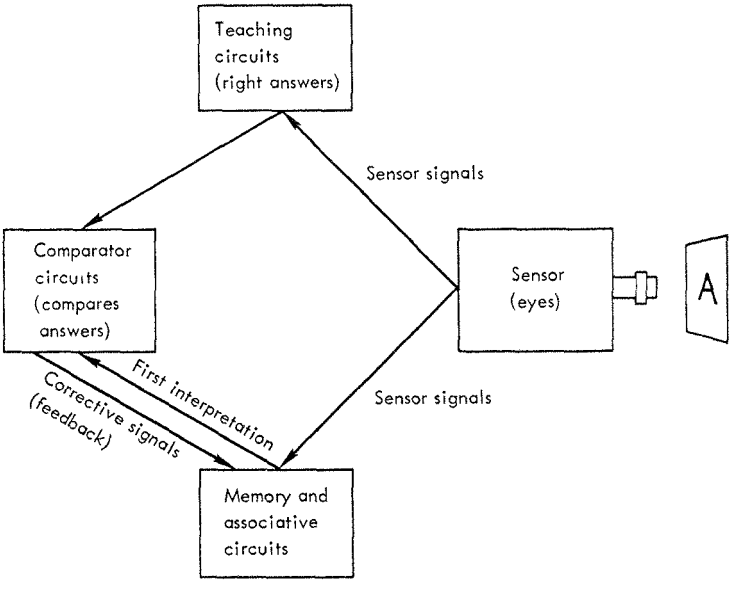
The matter of computer size is important not only because office space costs money but also because many special computer installations, as on aircraft and satellites, are severely weight-limited. The impact of microelectronics and integrated circuits is bound to be important. In microelectronics, transistors, wires, resistors, and other electronic components are deposited in miniature form on tiny chips of silicon and other "substrates". With com-



The major advances that have been made in computer size reduction are exemplified by these three equivalent modules holding logic circuits. Progress has been from the vacuum tube circuits (left), to transistor circuits (right), and finally to microelectronic concepts (center). Power requirements have been reduced as well as size.

ponents measured in millionths of an inch, instead of monstrous, inch-long vacuum tubes, it may be possible to place literally millions of electronic components within a cubic inch. One could then carry a good-sized electronic digital computer around like a transistor radio. This would be useful during school examinations—if the teacher permits.

Admittedly, computers can be made faster and smaller, but can they be made "smarter"—that is, more useful and more flexible? Already we have seen how a few primitive logical functions can be carried out so fast electronically that computers seem superficially to have intelligence. Even relatively "stupid" computers sometimes seem to have human properties, like those on homing torpedoes that have been known to "betray" the ship that launched them, or machines that play tic-tac-toe. What if computers could learn by themselves? What if they were self-



Block diagram of a "learning" experiment The machine is learning to recognize the letter A by viewing it with a TV camera and comparing the electrical signals produced with those from previous experiments in which the correct answers were given. When A is learned, there will be no further corrective signals (feedback)

sufficient, self-reproducing, and self-protecting? Machines CAN be given such qualities to some degree, but whether they will ever be able to *think*, *deduce*, *induce*, or *invent* is a moot point. None of the people who argue about such things have ever found mutually acceptable definitions of intelligence. Looking at facts, computers are doing more and more remarkable things as time moves on, and no amount of semantics will change this.

Take learning, for example. A machine can learn, in much the same way a human learns, if *feedback* is incorporated in its circuits—that is, if it is given a way to recognize its mistakes, remember them, and profit by them. A machine can learn in this sense by using teaching circuits that compare the machine's answers with the correct answers, which are then placed in the memory for future use. In this fashion, a computer with photoelectric eyes can learn to recognize patterns, such as letters of the alphabet, by showing it the letters in various sizes and positions while simultaneously giving it the correct answers. Computers have been taught to play chess and to direct mechanical rats that run mazes perfectly. Of course mistakes are made, especially in the beginning, but, in computers as in other intelligent species, "to err is human". And one might note, computers are just starting their evolution.

APPENDIX

The Interest of the Atomic Energy Commission in Computers*

Because of the importance of computation in the design of nuclear weapons and reactors, as well as for the advancement of basic science, the Atomic Energy Commission from the beginning has supported actively the advance of the art of designing and building digital computers. It has done this by supporting university groups, by conducting research and development in our national laboratories, and by purchasing the most advanced computers offered by the industry, often encouraging industrial development work by contracting to purchase a machine while it was still in the design stage. I feel that the continuing interest of the AEC has been one of the strong contributing influences toward the spectacular development of the computer industry.

There still exist problems in several areas important to the Atomic Energy Commission which tax the capacity of any existing computer, and will continue to tax that of any of the commercial machines which have recently been announced. Many of these are concerned with weapons design and effects computations, such as blast-wave propagation and atmospheric circulation, while others lie in quantum chemistry and reactor physics. Mathematically, many of these problems involve the numerical solution of partial differential equations in three space dimensions and time. This kind of problem demands, within the same computer, both very great computing speed and large memory capacity. Such a development objective constitutes a continuing challenge to the designer of computers.

Another area closely related to digital computers is that of computer-like devices for reading and processing experimental data. For several years now the need for such equipment has been clear. The successful development in the last decade of bubble chambers and of spark chambers,

---

*Excerpted from remarks by Dr. Glenn T. Seaborg, Chairman U. S. Atomic Energy Commission, at the dedication of Faraday Hall, Northern Illinois University, DeKalb, Illinois, February 5, 1965.

and the high pulse rates of modern accelerators, have made it possible for the experimenter to accumulate unmanageable quantities of data. For example, the beam of the Princeton-Pennsylvania Accelerator can be pulsed nineteen times per second. Hence, at this speed, in one eight-hour shift it would be possible to take over half a million triads of bubble chamber photographs. This means that with anything like full speed operation of such a machine it becomes impossible to deal with the data in any way other than automatically.

Present semi-automatic procedures simply are not fast enough. Equipment to scan and measure bubble chamber and spark chamber pictures and to record the data in digital form has had to be developed in our national laboratories and by our contractors, as no commercial equipment existed beyond that of a slow, manually operated type. The scanning, measuring, and recording of data is, of course, only preliminary to the problem of analysis, which must be done on a digital computer. An interesting by-product of these efforts was the discovery at the Argonne National Laboratory last year that a device originally designed to read spark chamber pictures could also be applied successfully to read biological slides.

A related type of use of computers is to control the direct or "on-line" acquisition of experimental data, which are then processed and the relevant derived information is fed back to the experimenter, who thereby acquires the ability of changing the course of the experiment while it is still in progress. A great deal of work in this area is going on at our national laboratories.

ANSWERS TO PROBLEMS

1 5.5

2 110.011; i.e., $6.375 = 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}$.

3 Since the register ends, the carry bit will be lost. A computer usually signals the operator when this happens.

4

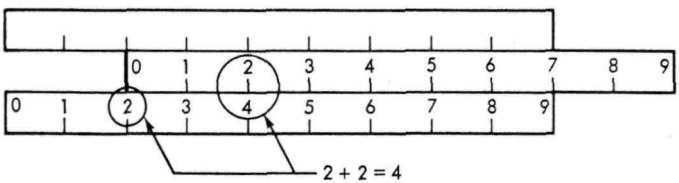
```
 11111   or    11111   or   31   or   31
-11001        +00110       -25       +74
 00110         00101         6        05
              +    1                  + 1
               00110                    6
```

5

LOAD	0001	x
MULT	0001	x^2
MULT	0001	x^3
MULT	0001	x^4
STORE	0002	
ONE	0003	
LOAD	0003	1
ADD	0003	2
ADD	0002	$2 + x^4$

6 When the gear ratio is 3, one gear will turn three times as far as the other. If a rotation of 10° = 1, the other gear will rotate 30°.

7



A linear rather than logarithmic scale is used.

SUGGESTED REFERENCES

Popular, But Sound Articles and Books

The Analytical Engine: Computers—Past, Present and Future, Jeremy Bernstein, Random House, Inc., New York, 1964, 113 pp., $2.95.

The Boundless Age of the Computer, G. Burck, *Fortune,* 69: 101 (March 1964). Part one of four parts.

Mathematical Machines, H. M. Davis, *Scientific American,* 180: 28 (April 1949).

Computers and Their Uses, William H. Desmonde, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1964, 296 pp., $10.00.

Computers, S. L. Englebart, Pyramid Publications, Inc., New York, 1962, 190 pp., $0.75 (paperback).

The Computers of Tomorrow, M. Greenberger, *The Atlantic,* 213: 63 (May 1964).

Charles Babbage and His Calculating Engines, Philip Morrison and Emily Morrison (Eds.), Dover Publications, Inc., New York, 1961, 400 pp., $2.00 (paperback).

The Thinking Machine, John Pfeiffer, J. B. Lippincott Company, Philadelphia, Pennsylvania, 1962, 242 pp., $5.95.

How Smart are Computers?, J. R. Pierce, *Saturday Evening Post,* 234: 24 (November 4, 1961).

How Streams of Water Can be Used to Create Analogues of Electronic Tubes and Circuits, C. L. Strong, *Scientific American,* 207: 128 (August 1962).

Self-Reproducing Machines, L. S. Penrose, *Scientific American,* 200: 105 (June 1959).

More Difficult Books

Giant Brains, Edmund C. Berkeley, Science Editions, Inc., New York, 1961, 294 pp., $1.65 (paperback).

Faster, Faster, W. J. Eckert and R. Jones, McGraw-Hill Book Company, Inc., New York, 1955, 160 pp., $3.75.

Computers and How They Work, J. D. Fahnestock, Ziff-Davis Publishing Company, New York, 1959, 228 pp., $4.95 (hardback), $2.95 (paperback).

Electronic Computers, Henry Jacobowitz, Doubleday and Company, Inc., Garden City, New York, 1963, 274 pp., $2.45 (paperback).

The Collected Works of John Von Neumann, A. H. Taub (Ed.), Pergamon Press, Inc., New York, 1961, 6 volumes, $14.00 each, $80.00 per set.

An Introduction to Symbolic Programming, P. Wegner, Hafner Publishing Company, Inc., New York, 1963, 219 pp., $6.45 (paperback).

Electronic Digital Computers, C. V. L. Smith, McGraw-Hill Book Company, Inc., New York, 443 pp., 1959, $12.50.

Digital Computer Primer, E. M. McCormick, McGraw-Hill Book Company, Inc., New York, 1959, 214 pp., $7.50.

Motion Pictures

Machines that Think, 29 minutes, sound, black and white, 1965. Produced by the Argonne National Laboratory. Research at Argonne into the future scientific uses of electronic computers is shown in this presentation that stresses nonnumerical manipulations of symbols. Computers are taught to make qualitative judgments, to interpret the significance of patterns such as spark chamber photographs, and to control laboratory experimental apparatus. Available for loan without charge from the AEC Headquarters Film Library, Division of Public Information, U. S. Atomic Energy Commission, Washington, D. C. 20545 and from other AEC Film Libraries.

Thinking Machines, 19 minutes, sound, color, 1960. Produced by the Educational Testing Service in cooperation with the National Science Foundation. This film presents approaches and experiments in machine "intelligence". A mechanical mouse that learns by trial and error, a chess game against a giant computer, and a machine that recognizes visual patterns are highlights. This film is available on loan from an audiovisual center in each state. A list of these centers can be obtained from the Educational Testing Service, Princeton, New Jersey 08540.

The following films were produced by KQED in San Francisco for National Educational Television under a grant from IBM. Each is in black and white and is 29 minutes long. They are available on loan from the Audiovisual Center, Indiana University, Bloomington, Indiana 47405.

Logic by Machine (bs-87) serves as an introduction to the subject of electronic computers and to this series of films. The relationship of the symbolic world of mathematics to the real world of objects and events is illustrated. A brief description of the abilities of the computer to calculate, remember, and compare is given.

Universe of Numbers (bs-88) gives a history of computer development beginning with Blaise Pascal's calculator in the 17th century. Explanation of how a computer solves a problem is given and a computer is shown being programmed.

Universal Machine (bs-89) discusses the problem of deciding which universal "machine language" to use. The computer revolution is commented upon in terms of its difference from past industrial advances.

The Control Revolution (bs-90) illustrates the basic elements in a modern, continuous, self-adjusting control system through animation. Computers are shown at work in various industries—from controlling and operating the tooling of delicate machine parts in a factory to recording, storing, and processing data at the Social Security Administration.

Managers and Models (bs-91) outlines the design and simulation capacities of the modern digital computer. This work is explained by describing such computers as the one used at the

Marshall Space Flight Center to simulate in advance the problems encountered in firing a space rocket.

Engine at the Door (bs-92) answers the question "will machines ever run man?".

PHOTO CREDITS

Cover courtesy Educational Testing Service

Page	
5	IBM Corporation
6	IBM
7	Massachusetts Institute of Technology
8	Harvard University
11	IBM
17	IBM
20	IBM
21	IBM
38	IBM
45	David M. Raup, Johns Hopkins University. Photo originally appeared on the cover of Science (March 12, 1965).
48	IBM

This booklet is one of the "Understanding the Atom" Series. Comments are invited on this booklet and others in the series; please send them to the Division of Technical Information, U. S. Atomic Energy Commission, Washington, D. C. 20545.

Published as part of the AEC's educational assistance program, the series includes these titles:

Accelerators
Animals in Atomic Research
Atomic Fuel
Atomic Power Safety
Atoms at the Science Fair
Atoms in Agriculture
Atoms, Nature, and Man
Books on Atomic Energy for Adults and Children
Careers in Atomic Energy
Computers
Controlled Nuclear Fusion
Cryogenics, The Uncommon Cold
Direct Conversion of Energy
Fallout From Nuclear Tests
Food Preservation by Irradiation
Genetic Effects of Radiation
Index to the UAS Series
Lasers
Microstructure of Matter
Neutron Activation Analysis
Nondestructive Testing
Nuclear Clocks
Nuclear Energy for Desalting
Nuclear Power and Merchant Shipping
Nuclear Power Plants
Nuclear Propulsion for Space
Nuclear Reactors
Nuclear Terms, A Brief Glossary
Our Atomic World
Plowshare
Plutonium
Power from Radioisotopes
Power Reactors in Small Packages
Radioactive Wastes
Radioisotopes and Life Processes
Radioisotopes in Industry
Radioisotopes in Medicine
Rare Earths
Research Reactors
SNAP, Nuclear Space Reactors
Sources of Nuclear Fuel
Space Radiation
Spectroscopy
Synthetic Transuranium Elements
The Atom and the Ocean
The Chemistry of the Noble Gases
The Elusive Neutrino
The First Reactor
The Natural Radiation Environment
Whole Body Counters
Your Body and Radiation

A single copy of any one booklet, or of no more than three different booklets, may be obtained free by writing to:

USAEC, P. O. BOX 62, OAK RIDGE, TENNESSEE 37830

Complete sets of the series are available to school and public librarians, and to teachers who can make them available for reference or for use by groups. Requests should be made on school or library letterheads and indicate the proposed use.

Students and teachers who need other material on specific aspects of nuclear science, or references to other reading material, may also write to the Oak Ridge address. Requests should state the topic of interest exactly, and the use intended.

In all requests, include "Zip Code" in return address.

Printed in the United States of America

USAEC Division of Technical Information Extension, Oak Ridge, Tennessee