PROCEEDINGS OF THE I·R·E

COMPUTER ISSUE

Vol. 41 OCTOBER · 1953 No. 10

PROCEEDINGS OF THE I.R.E.®

Published Monthly by
The Institute of Radio Engineers, Inc.

VOLUME 41 — *October, 1953* — NUMBER 10

PROCEEDINGS OF THE I.R.E.

Contents continued, following page

PROCEEDINGS OF THE I.R.E.®

Published Monthly by

The Institute of Radio Engineers, Inc.

ACKNOWLEDGMENT

The reader will immediately recognize that this issue of the PROCEEDINGS differs from a regular issue in two important respects: First, it is a special issue devoted wholly to a single subject, namely, Electronic Computers. Secondly, it has been expanded to approximately three times the size of a regular issue. But perhaps the most noteworthy feature of this issue is the fact that it was assembled with the full collaboration of the IRE Professional Group on Electronic Computers (PGEC).

The Editorial Department is most grateful to J. H. Howard and M. M. Astrahan, PGEC Chairman and past Chairman, respectively, and to the Administrative Committee of the Group for their generous cooperation and helpful guidance in planning this issue. Special thanks are due to the PGEC Editorial Board and Papers Review Committee which, under the chairmanship of Werner Buchholz, reviewed the many papers submitted for the issue, brought about the inclusion of informative introductory and tutorial material, and arranged the papers in a coordinated sequence.

That the PGEC was in a position to render such substantial assistance in preparing this issue is a significant commentary on the extent to which this Group, and indeed all twenty-one Professional Groups, have developed in a few short years. In the two years since its formation, the PGEC has grown into a thriving organization of 2,000 members which issues its own technical publications, annually co-sponsors computer conferences on each coast, sponsors sessions at the IRE National Convention, and has organized local Chapters which hold meetings in conjunction with IRE Sections in Los Angeles, Philadelphia, San Francisco, and Washington, D. C. Additional Chapters are being formed in Boston, New York, and Detroit.

Of these many activities, perhaps the most important is the technical publication, called *Transactions*, which is issued quarterly to all members who have paid the $2.00 Group assessment. PGEC members have found the *Transactions* to be an invaluable source of authoritative information on the latest computer developments.

To these accomplishments of the PGEC must now be added a major share of the credit for the Computer Issue. And as the reader peruses the following pages, it is felt he will be quick to agree that the credit is well deserved.

—*The Administrative Editor*

The Computer Issue

WERNER BUCHHOLZ

The following guest editorial by Werner Buchholz is required reading for every reader of this issue. In it, the author provides the reader with a concise and invaluable guide to the significance of each paper presented herein together with an informative discussion of the growth of Electronic Computers as a branch of the radio engineering field.

Werner Buchholz is admirably qualified to introduce the reader to this issue by virtue of his position as Chairman of the Editorial Board and Papers Review Committee of the IRE Professional Group on Electronic Computers, to which fell the task of reviewing all the papers in this issue.—*The Administrative Editor*

About a year ago the IRE devoted a special issue of the PROCEEDINGS to one tiny component, the transistor, because its development promised to be one of the major advances in electronics. The present issue focuses attention on some of the largest assemblies of electronic components ever made, electronic computers. These machines frequently contain several thousand vacuum tubes each, and they are probably the most complex pieces of equipment of any kind ever assembled which have to operate as a single, centrally coordinated system, where each function depends on the correct operation of almost every component. The design of a successful computer demands a high degree of engineering skill, and the papers in this issue reflect the amount of engineering effort that has been expended in this rapidly growing field.

The urgent expansion of the electronic computer industry can be gauged by the many personnel advertisements appearing in the professional journals and the daily newspapers. The current situation is comparable to the rapid development of radar during World War II. Another sign of growth is the development of the IRE Professional Group on Electronic Computers: it has grown in just two years to be the third largest Professional Group in the IRE, having 2,000 members.

The PGEC co-operated with the Editorial Department of the IRE in the preparation of this special issue of the PROCEEDINGS. The purpose of this issue is two-fold: To provide a set of stimulating and informative articles which would introduce the non-specialist reader to the new and exciting field of electronic computer engineering, and to furnish the specialist with a single volume of reference material on a wide variety of computer subjects. This issue of the PROCEEDINGS thus gives recognition to the work being done by a young, but active, Group in the IRE. In the future the IRE will, of course, continue its usual publication policy: The normal outlet for the more specialized computer articles will be the quarterly *Transactions* of the Professional Group on Electronic Computers, while contributions of more general interest or developments of major significance will appear in the PROCEEDINGS from time to time.

Although electronic computers are a fairly recent development, it is well to point out that automatic computers considerably antedate the "age of electronics." We might choose to ignore the work during earlier centuries of such pioneers as Babbage, who first formulated many of the basic principles of the modern computers one hundred years ago, but whose attempts to build one foundered because current technology proved inadequate. However, we must include the Bush Differential Analyzer and the punched-card machines of the 1920's as the direct ancestors of the machines described in this issue. The first realization in electro-mechanical form of most of Babbage's ideas came in 1944, when the IBM Automatic Sequence Controlled Calculator (the Mark I at Harvard) was completed. Still, the present growth of the computer industry did not start until the results of the enormous development of electronic technology during World War II were brought into the field. It is interesting to note that many computer projects started around a nucleus of wartime radar experts. Electronics not only provided the technological means for greatly increased speed and capacity, and thereby enhanced the usefulness of computers many times, but the availability of cheap, mass-produced components and of engineers trained to use them made it possible to experiment on a greater scale and at a lower capital investment than before. In the late

40's and until the early 50's ideas for new computers and their weird names sprouted in great abundance all over the world.

There was a significant time lag, however, between the announcement of many a computer and its first successful operation. During this time the designers had to face engineering realities. The low-cost components of the radio industry were found to be inadequate for the high-precision demands of a complex computer, and new components and methods of construction had to be devised. The high-speed electronic computer also required input-output equipment of adequate capacity, which meant extensive development of electromechanical devices. Finally, when the machines were finished, better techniques for using them had to be devised. The resultant operating experience led to further design improvements. The contents of this issue reflect the maturing effect of several years of actual operation.

In spite of this period of sober adjustment the industry has continued to grow. Several electronic computers are in commercial quantity production; one type of machine alone accounts for two million tube sockets now in active computer service. Computers have become indispensable for engineering calculations in defense industries, particularly in the aircraft industry. Even more are being built for accounting applications to cope with the problems of an economy of ever-growing complexity. Thus electronic computing may soon affect the average person as much, if not as obviously, as radio and television; it is certainly of vital interest to the electronics engineer.

The Contents of this Issue

The Computer Issue starts with a series of invited articles by A. L. Samuel, M. V. Wilkes, and C. E. Shannon. These well-known authors discuss the nature, uses, and the limitations of computers, and touch on possible further developments in their imaginative and eminently readable papers.

In the next paper electronic computers are looked at from the point of view of the telephone switching engineer, thus providing an interesting parallel (Lewis). One cannot have a real appreciation of the nature of digital computers until he is familiar with the fundamentals of programming a typical machine. A tutorial paper on this subject (Thomas) provides a background which applies to most of the present-day digital computers even though they differ in detail. Basically, programming is a simple, logical procedure, but as the problems to be solved grow, the labor of programming also increases, and the aid of the computer is enlisted to devise its own programs (Hopper and Mauchly). To round out this series of papers there is a comparison of digital and analog computers (Rubinoff).

No computer publication would be complete without an account of some of the important digital-computer projects in existence: their purpose, their design, and current operating experience. The Type 701 Computer, a large-scale machine which IBM is producing in quantity, is described in a set of papers covering the over-all system design (Buchholz), the engineering design (Frizzell), and the arithmetic element (Ross). The SWAC (Huskey, Thorensen, Ambrosio, and Yowell) and the SEAC (Greenwald, Alexander, and Haueter) of the Bureau of Standards, and the NAREC of the Naval Research Laboratory (Sherertz) are representative of the work being done by government agencies. The problems of servicing a large, general-purpose digital computer are eased by the fact that the computer itself serves as its own test instrument; the ILLIAC at the University of Illinois serves as an example (Wheeler, Robertson). From Remington Rand, Inc. come descriptions of two plugboard-controlled computers: the Logistics Computer (Erickson) and the Type 409-2 Computer (Crosman). The last one is an example of a class of much smaller electronic computers of less capacity which are being produced by several companies in larger quantities for accounting purposes. The Bell Telephone Laboratories have constructed two exploratory, special-purpose machines: one to investigate the possibility of an electronically-controlled telephone exchange (Malthaner and Vaughan), and the other to aid in the analysis of relay circuits (Shannon and Moore). Both machines illustrate the fact that computers do not necessarily compute on numbers; these machines apply to switching and combinatorial problems, and techniques intended for numerical work are not very efficient on them. Although the large, "general-purpose" computers are extremely fast on numerical problems, a simple combinatorial problem as determining the best interconnection of a few relays can slow them down to a walk. "General-purpose" is a relative term.

Analog computers are generally described as using continuously variable functions. There is possibly one exception, the Digital Differential Analyzer, which solves problems in much the same manner as an analog differential analyzer, but which uses digital circuits, digital arithmetic, digital programming, and which provides a digital output. An improved version designed at the Bendix Aviation Corporation is discussed (Palevsky). This interesting type of machine bridges the gap between digital and analog computers. It shows that the two techniques do not represent separate and competing schools of thought, as is commonly assumed. One may expect digital or analog methods to be applied where each is best suited.

Next comes a series of papers on theories, techniques, and components useful in designing digital computers. The reader interested in theory will find the foundation laid for the study of a digital computer as a single large network, and the physical realizability of a given logical net is discussed (Burks and Wright). A tutorial paper on Boolean algebra gives the circuit designer a thorough and precise exposition of this important design technique (Serrell). A detailed account is given of a new design of standard circuit packages for the SEAC type of computer (Elbourn and Witt). These circuits have been used as prototypes by other computer projects, and already further improvements in the design of logical switching circuits are reported (Gluck, Gray, Leondes, and Rubinoff). Under the heading of storage there is an extensive survey of memory systems as evaluated by one of the pioneers in digital-computer engineering (Eckert), and another paper gives an idea of the effort being put into developing the still relatively new magnetic-core storage elements into a practical, large-scale, random-access memory system (Rajchman). Photographic techniques, which have been highly developed elsewhere, provide some excellent possibilities for computer storage which are outlined (King, Brown, and Ridenour). They have so far gone largely unnoticed because of obvious technical difficulties, but it should be a fruitful area of research. A binary counter (Ware) and a magnetic-drum recording technique (McGuigan) are presented, shedding some new light on well-explored territory. Two of the biggest handicaps of large computers are their very size and power requirements, for which it is hoped that the transistor, replacing the vacuum tube amplifier, will eventually offer a solution (Vogelsong). Computers operating in the decimal system require a code to represent the decimal digits; the number of variations and different considerations involved is surprisingly large (White). The inclusion in an IRE publication of a paper on the design of a mechanical device may be a little surprising, but it points out the important contribution of good mechanical design to the success of electronic computers (Oster and Wilson).

One use of digital computers is to digest large amounts of raw data produced by measuring instruments, in a windtunnel, for instance, and to put out finished results in tabular form. This data-processing application is closely related to commercial accounting problems, and standard accounting machines are frequently used here for computing, except that the input data are in analog form. This requires conversion of the data from analog to digital form. Various methods are surveyed in one paper (Burke) while another dedescribes a specific conversion technique (Gray, Levonian, and Rubinoff). Yet another use of the digital computer is as a control element, where it must communicate directly with the outside world, usually by continuous variables of the analog type, and be able to smooth out any noise contained in the input (Spero). Here again we see the close relation between analog and digital techniques in certain applications.

In the area of analog computers proper there are three papers dealing with new components: a multiplier (McCool), a magnetic amplifier (Craig), and an input-output unit capable of plotting a graph and automatically reading it back (Vance and Haas). Many different kinds of problems suitable for solving on the electronic analog computers (differential analyzers) in existence are indicated (Meneley and Morrill), the extension of analog computing to partial differential equations by the use of difference techniques is discussed (Howe and Haneman), and an application to noise studies is shown (Bennett). The paper on economic analogs (Smith) should be of wide interest because it shows that economic systems are feedback systems similar to servomechanisms, and analog computers can be used in studying economic models. This opens the possibility that major business policy decisions might in future be based on computer studies just as engineering decisions are today.

Finally the reader's attention might be drawn to the review of a new book on computers. In view of the scarcity of books in any rapidly expanding field, the publication of a new text is always of interest.

Computing Bit by Bit or Digital Computers Made Easy*

ARTHUR L. SAMUEL†, FELLOW, IRE

Since many readers may not be familiar with the technical language used by the computer engineer, particularly in the field of digital computing, the Editors asked A. L. Samuel to write an informal, but informative, introductory paper. Dr. Samuel, who is a Fellow of the IRE, is in charge of physical and chemical research related to the development of computer components at the IBM Laboratory in Poughkeepsie, N. Y. His broad interest and his close association with several computer projects qualify him to interpret the efforts of the computer designer to the non-specialist.—*The Editor*

THE POPULAR PRESS has had a great deal to say about "mechanical brains," so much so in fact that the first task for a paper of this sort might be that of placing the modern digital computer in the right perspective. However, the writer believes that you, the reader, are perfectly able to draw your own conclusions once you are presented with the facts. Your conclusion may well be that we are witnessing a revolution, akin to the industrial revolution, in which mankind will now be freed of the mental drudgery which has replaced the drudgery of unremitting manual labor. Or again, you may conclude that the modern digital computer has been oversold to the public and that there is still a long mental row to hoe before we will reach Utopia. Perhaps the truth lies somewhere in between.

As an aid to the non-specialist, this paper will attempt to explain in an elementary fashion the how's and why's of the modern digital computer and in so doing, to introduce some of the "lingo" of the computer engineer. This should give the reader an initial basis for judging for himself the role the modern computer appears destined to play in the future, and it may help him to read the more highly technical papers which follow. The paper will then briefly review those aspects of scientific research and engineering development as applied to computers in which the greatest opportunities for improvements seems to lie.

THE NEED FOR COMPUTERS

It is hardly necessary to point out the need for automatic calculators in view of the obvious advantages in terms of the reductions in clerical manpower and woman power which result from their use. The fact that such computers make possible a form of numerical experimentation which has never before been exploited, is perhaps somewhat less obvious. We are all aware of the existence of involved mathematical problems which are so lengthy and complicated that mankind has not been able to make even a start on their solution. The accurate forecasting of the weather, which certainly must depend upon the interaction of measurable effects, but in a very complicated and involved fashion, is but one example of a host of enticing problems which demand solution and which are for the first time coming within range. While a great deal of progress has been made in mechanizing office procedures, we all realize that only a small fraction of the entire job has been attempted. And finally, with the increasing mechanization of industry, the need for calculations as a part of automatic control is becoming more obvious. I think that we will all agree that there exists a real need for automatic calculators and that this need is increasing at a remarkable rate.

At first glance, a computer is a very complex device. It can be looked at from two different angles, which Professor Hartree[1] has called the "anatomical" and the "physiological," that is, "of what is it made?" and "how does it tick?" As engineers and scientists you can be expected to know something about its anatomy. It is made up of the same building blocks you know so well; resistors, condensers, magnetic cores, vacuum tubes, diodes, yes, maybe even transistors. These are put together into circuits which differ in detail it is true, but in detail only from many very familiar circuits. The "anatomy" is familiar, but the "physiology," the way these components work together to do computing, this is unique to the digital computer. An "introduction to physiology" seems to be in order.

THE COMPUTER IN FUNCTIONAL TERMS

What then is a computer in functional terms? Any computer can be thought of as an information or data processing device which accepts data in one form and delivers it in an altered form. This is as true for the

* Decimal classification: 621.375 2. Original manuscript received by the Institute, August 12, 1953.
† International Business Machines Corp., Poughkeepsie, N. Y.

[1] D. R. Hartree, "Calculating Instruments and Machines," University of Illinois Press, p. 56; 1949.

modern high-speed digital computer as it is for the adding machine and the lowly abacus. Just as we can understand the use of an adding machine without knowing in detail how the internal mechanism is constructed, we can as easily understand the largest digital computer.

Looked at from another point of view, when the human operator performs a reasonably complicated numerical calculation he is forcing his brain to act as a digital computer. Since all of us are capable of analyzing our own thought processes as we first of all break down the problem into its simple elements and then perform these simple computations, and we can do this incidentally with only the vaguest ideas of the way in which the individual brain cells operate, we should with equal facility be able to understand the operation of a device which is much less complicated than the human brain.

"Yes," you say, "that is all very well, but what is all this about digital versus analogue machines? What is the difference between a binary and a decimal machine? And what, pray tell, do you mean by a stored program?" Well let's just take things one at a time. Let us first of all briefly consider the analogue machine before proceeding to digital machines.[2]

The Analogue Machine

If words mean anything, an analogue machine must solve problems by analogy. Since there are many kinds of problems, and since it is frequently possible to devise many different analogies for any one class, there will be many types of machines which differ from each other in almost every conceivable way. Most of the problems of engineering, and many of the problems in physics deal with continuous parameters. One suspects that there will be parameters in the machine, perhaps shaft rotations or displacements, currents or voltages, which are expected to vary continuously in proportion to the real parameter and so simulate them. Furthermore, since the parameters in the real situation may be determined by relatively complex relationships, the machine must be able to relate various parameters according to these same mathematical rules. Since many of the laws of nature can be expressed in terms of integrations, integrators occupy a fairly prominent place in the analogue machine, at least in the more complicated ones. Sometimes the parameters employed in the machine are not proportional to the simulated parameters, but are related to them in some functional way. The slide rule, one of our simplest analogue machines, is a case in point. Here distances along the rule are not proportional to the quantities to be represented but rather to the logarithms of these quantities. This makes the device particularly suited for solving problems involving multiplication which is then performed by adding lengths proportional to the logarithms of the quantities to be multiplied. In summary, the analogue machine performs a specialized function in a very efficient manner and can frequently outperform a digital computer in the solution of specific types of problems.

Counting on One's Fingers

A digital machine obviously uses digits, so it counts, and just as it is possible to do arithmetic by counting on one's fingers—it does arithmetic. In fact, some of the newer and larger machines are unbelievably naive. They do not even know how to count to ten; instead they can only count up to two. These machines are said to be binary.[3] We will have more to say about this later. An ordinary adding machine is a simple example of a digital computer. Any operation which can be reduced to arithmetic or to simple logic can be handled by such a machine. There does not seem to be any theoretical limit to the types of problems which can be handled in this way. The user may occasionally find, to his chagrin, that it is harder to reduce a given problem to arithmetic than it is to solve the problem by direct methods.

Fortunately, problems requiring the use of the modern digital computer have a habit of falling into classes. Solution methods have already been worked out or are currently being developed for many of these classes so that the work of analyzing a special problem may be limited to the task of identifying the class to which it belongs. This is a situation which will continue to improve with time, and particularly as more and more machines of the same type come into general use. Nevertheless, a great deal of thinking is required when one attempts to make use of a digital computer. Once this thinking has been done the computer makes up for the delay by its speed in arriving at the solution which may easily be 20,000 or 30,000 times as great as that of a man operating a desk calculator.

Can Computers Think?

It might be well, at this point, to dispel some of the fuzzy sensationalisms of the popular press regarding the ability of existing digital computers to think. Over a hundred years ago Lord Byron's daughter, Lady Lovelace, in commenting on Charles Babbage's analytical engine, made a remark which is as true today for the modern computer as it was then. She said, "The analytical engine has no pretentions whatever to originate anything. It can do whatever we know how to order it to perform."[4] More recently Dr. M. V. Wilkes, in his characteristic style, had some very illuminating things to say on this same subject.[5] The digital computer can and does relieve man of much of the burdensome detail

[2] For a comparison of analogue and digital machines see M. Rubinoff, "Analogue vs. digital computers—a comparison," Proc. I.R.E., pp. 1254–1262; this issue.

[3] Even so-called decimal machines usually make use of binary devices and must actually express each digit by its binary equivalent, or at least in terms of some coded equivalent. See G. S. White, "Coded decimal number systems for digital computers," Proc. I.R.E., pp. 1450–1452; this issue.

[4] R. Taylor, "Menebrea," Lf Scientific Memoirs (London Ed.), vol. 3, p. 722; 1842.

[5] M. V. Wilkes, "Can machines think?," *Discovery*; May 1953. (The editors have thought so highly of this paper that it is reprinted in Proc. I.R.E., pp. 1230–1234; this issue.

of numerical calculations and of related logical operations, but perhaps it is more a matter of definition than of fact as to whether this constitutes thinking.

Regardless of what one calls the work of the digital computer, the unfortunate fact remains that more rather than less human thinking is required to solve a problem using a present day machine since every possible contingency which might arise during the course of the computation must be thought through in advance. The jocular advice recently published to the effect, "Don't Think! Let UNIVAC do it for you,"[6] cannot be taken seriously. Perhaps, if IBM's familiar motto needs amending, it should be "Think: Think harder when you use the 'ULTIMAC'."[7]

Information Processing

We will start then by considering a digital computer as being merely an information processing device. This is the only thing that the computer does. *It cannot create any new information* not contained in the original source, although *it may transform the input information* into a very much more useful form. This is an important concept, and a full recognition of its implications will go a long way toward dispelling any mystical feelings with respect to the modern computer.

The input information to be processed may be the statement of a mathematical problem, the rules governing the mathematical operations to be performed, and the input data. It may be the statement of a problem in logic or, if one wishes, the rules for playing some game such as checkers or chess together with the conditions of the board at some particular time and a statement of the logic to be applied to determine the best next move. The digital computer is capable of handling all of these cases and more[8] but in every case it simply processes the input data according to rules specified by the operator. By analogy we might call the computer a mathematical translator in the same sense that a literary translator takes information in one language without adding or subtracting any basic information and renders this information intelligible to someone not understanding the original language.[9]

One aspect of this information-processing concept, as already intimated, has to do with the fact that the computer can, and does, make use of more than one source of information during the processing. One such source of information is the input data. A second source of information is the list of instructions. Still another source is, in effect, a rule book of mathematical and logical operations, which can by analogy be related to the translator's bilingual dictionary. This rule book may be built into the wiring of the machine, or it may be given to the machine, as needed, in the form of instructions.

We might express the idea of Information Processing by saying that a computer simply rearranges information from one spatial relationship at the input to the machine to another spatial array at the output. However, a moment's reflection will show that the machine must translate information not only in space but also in time. Consider a situation in which we tell the machine how to perform a series of operations. It is customary first to give the machine the program (or list of instructions) and, subsequently, to give it the numbers on which the operations are to be performed. The machine must, therefore, be able to remember the instructions at least until it receives the data on which it is to operate. In fact, since most of the required operations cannot be performed simultaneously, a continued use of memory or information storage is required.

Switching and Storage

Getting closer to the language used by computer engineers we can say that two functions are basic to any computer, these being information switching and information storage. By switching we mean changing the arrangement of information in space; while storage can be thought of as changing the time sequence of information. Switching and storage are frequently discussed as separate entities, although it is quite unusual for them to be actually separated, except in their simplest physical embodiments. Most so-called switching devices will on closer analysis be found to contain elements of memory or of information storage; while most storage devices in order to be at all effective must contain certain internal switching mechanisms.

To bring the discussion down to earth let us consider the ordinary electric light switch in your home. This is by definition a switch. It enables one to direct electric current to a lighting fixture at will. Usually there is a detent mechanism which enables the switch to remember what it is supposed to be doing so that once you turn the lights on they will remain on. It therefore has a memory. It is also a binary, or perhaps we should say a bistable device. By way of contrast, the ordinary telegrapher's key is a switch without memory since the key will remain down only as long as it is depressed by the operator's hand. Both the light switch and the telegraph key are binary devices, that is, they have but two operating states. While we are on the subject we might note that most of the switching and storage mechanisms which are employed in modern computers are also essentially binary in nature. Because of this, many of the larger machines operate in the binary number system.

[6] A. F. Draper, "Univac on election night," *Elec. Eng.*, vol. 72, pp. 291–293; April, 1953.

[7] A coined term for the "Ultimate in Automatic Computers." The reader may, if he prefers, insert any name he likes selected from the following partial list of existing machines: *Ace, Amos, Ape(x)c, Arc, Avadac, Bark, Binac, Cadac, Caldic, Circle, Cpc, Deuce, Dyseac, Edsac, Edvac, Elecom, Era-1103, Ferut, Flac, Hurricane, IBM-701, Illiac, Johniac, Leo, Lorpgac, Madm, Maniac, Midac, Mosaic, Narec, Nicholas, Oarac, Omibac, Oracle, Ordvac, Rascal, Raydac, Seac, Sec, Swac, Univac, Whirlwind, Wisc,* etc.

[8] Read C. E. Shannon, "Computers and automata," Proc. I.R.E., pp. 1234–1241; this issue.

[9] This reference to literary translation is something more than a simile since the use of digital computers for translating is already receiving serious consideration. The limited memory capacity of existing machines is the only real deterrent, although the problem of programming and coding is far from simple.

Of course, you can spoil all the fun by asking about the three-way switch on the common floor lamp and so bring up the subject of ternary arithmetic, but we will ignore this complication since no one has yet come up with a ternary machine.

Component Devices

In cataloging the basic mechanisms involved in a computer into switching and storage we have unwittingly discussed the computer as an entity and indeed this is one way to look at the problem. However, when one attempts to build a practical computer out of these two basic kinds of elements one finds it desirable to group the switching and storage elements into subassemblies which perform specialized functions in terms of the flow of information through the machine. Let us consider some of these functions.

In the first place we must get the information into and out of the machine. This calls for some sort of input device and one or more output devices. If the information is not presented to the machine in exactly the most convenient form, both with respect to its spatial and temporal relationships, we can expect the input device itself to contain both storage and switching functions. Then, since the operations to be performed on the input data are either logical or arithmetic there must of a consequence be a logical and arithmetical unit.[10] In some machines the only operation performed is that of addition, in which case this portion of the machine is simply called the adder. We certainly need an input device, an output device, and an adder or a logical and arithmetical unit.

Now let us consider how one would perform a simple multiplication. If you ask an assistant to multiply two, three or four digit numbers together, his first impulse will be to pick up a scrap of paper on which to write the two numbers to be multiplied.

By the way, we are going to run into trouble with nomenclature in the use of the term "number," which can mean either a digit or a group of digits. It might be well to adopt the computer engineer's terminology in which he may speak of "digits" but will use the term "word" to apply to a group of digits. Such an array of digits may, of course, have a non-numeric significance; a good example is an instruction. A purist will even object to the use of the term digit, since this implies a decimal machine and many modern machines are binary where the term "bit" is more appropriate. The most general term and the one obviously required for machines handling alphabetical as well as numerical data is the term "character."

So you have asked your assistant to multiply a couple of three or four-character "words" together. He starts by writing down the given words. He will then perform the operation of multiplication and will probably write down a number of additional "words" on the piece of paper, the so-called partial sums which he uses during the course of the calculation, but which of themselves are not the answer. This calls for a somewhat larger piece of paper than that which would be necessary to write the two given "words" and the "word" which represents the final answer. He is, in effect, using a piece of paper as a temporary storage device or supplementary memory to store "words" needed during the calculation. The scientific computer obviously needs just such a memory and a special portion of the machine is assigned this responsibility and is usually referred to as "the memory." As we will see later, it is customary to make this memory considerably larger than would be required to store temporary information of the type just mentioned and to store in this same memory most if not all of the instructions necessary to perform a rather lengthy calculation, much of the input data and a substantial part of the output data, or at least until enough has accumulated to keep the output device working continuously for a reasonable period of time.

As you may have noted, we have used the terms "storage" and "memory" more or less synonymously. This usage of the term memory to refer to certain types of storage has become quite common and some people have gone so far as to ascribe human attributes to digital computers. Any anthropomorphical implication of this sort in the use of the term memory should be dissipated when I remind you that the printed page is perhaps still our best information storage medium in terms of storage capacity and from this point of view it far surpasses the more glamorized types of memory currently used in computers. By memory, then, we will simply mean a specialized portion of the machine capable of providing information storage.

We have discussed input and output devices, the logical and arithmetical unit, and the memory. The only major part left is the control unit which forces these other components to work as a team. Control will largely involve switching although even here there will be some storage functions. For example, there will be a counting register to keep track of the current computational step being performed, other registers to remember which of the various possible input or output units are currently being employed, etc.

Programming[11]

We are now at the nub of the problem—how does the control work? Well there are a variety of ways of varying degrees of complexity, but for simplicity we will discuss just two types which we will refer to as fixed programming and stored programming. By fixed programming we mean the kind of programming which controls your automatic dishwasher for example. Here the sequence of operations is fixed and built into the wiring of the control or sequencing unit. Once started, the dishwasher will proceed through a regular series of opera-

[10] One such form of unit is discussed in H. D. Ross, Jr., "The arithmetic element of the IBM type 701 computer," Proc. I.R.E., pp. 1287–1294; this issue.

[11] For a more detailed description of the logic of one type of programming, see W. H. Thomas, "Fundamentals of digital computer programming," Proc. I.R.E., pp. 1245–1249; this issue.

tions, washing, rinsing and drying. Of course, if one wished, one could change the wiring to alter the program. In many automatic calculators of the fixed program type this facility is provided in the form of a plugboard on which the wiring can be changed, or as a still further refinement, the plugboard itself is removable and quickly replaced by another previously wired board.

Lest you get the idea that this type of programming is as unimaginative and prosaic as a dishwasher let me remind you that considerable sophistication is possible. For example, one can cause some of the program steps to make tests and perhaps transfer control not to the next step in order but to some other step. Suppose one sets up a closed loop of steps which will do the same thing over and over again, a so-called iteration loop, perhaps to solve for some quantity by the method of successive approximations. It is easy enough to get such a loop started, but it is also necessary to stop it. This can be done by having as one step in the loop a computation of the difference between the last two approximations. Whenever this difference is close enough to zero (being smaller than some preassigned value) one can arrange to terminate the loop. A sort of judgment might be implied in such an operation but on more careful analysis the judgments were actually made by the programmer when he decided to terminate the loop in this way and when he picked the difference value. The machine simply tests as directed and uses the result of the tests in a manner which has been prescribed in advance. The term "programming" is obviously used to refer to those phases of the planning of a problem for a computer which consist of recording the steps required, the tests to be made, and the way in which the solution steps are to be altered as a result of these tests.

The Stored Program

And now we come to the concept of a stored program. . . . Suppose you wished to give your assistant a large number of instructions for manual computations all in advance. You could do this by supplying him with a prepared set of instructions, or you could dictate the instructions and have him write them down, perhaps at the top of the same sheet of paper on which he is later to perform the computations. Two different situations are here involved, although at first glance the distinction appears trivial. In the first case the instructions are stored on a separate instruction form, while in the second case they are stored by the same medium which is used for data. Both situations are found to exist in computing machines. The first case is exemplified by certain machines which use special program tapes. The second situation is becoming quite common in the newer machines and is the case usually meant when the term "stored program" is used.

To make the matter definite let us review some of the steps which might be required to store a program and to execute the program once it is stored. The machine operator must prepare a list of the instructions and a list of the input data, either on punched cards, on a punched tape, or perhaps on a magnetic tape, in any case on some medium which can be read by the input mechanism of the computer. The instructions could conceivably be in ordinary English but, obviously, it will be expedient to adopt some sort of shorthand notation or "code" in terms of numbers or letters which condenses the information as much as possible and presents it in a more convenient form for machine use. It is customary to have these instructions in the form of "words" which are similar in every respect to the "words" used to represent numerical quantities. This process of translating a program into the machine's language is called "coding."

Now consider a typical case where the start or load button on the operator's panel will only cause the machine to read-in the first two instructions and then transfer control to the first of these instructions. The reading operation will cause these two instructions to be stored in the first two positions in memory, but what happens then? Well, we see the need for at least two registers, one in the form of a program counter or register which keeps track of the program step number, or more generally of the address in memory from which the current instruction is to be obtained, and an instruction register into which the current instruction can be transferred in order to enable it to set up the necessary switching so that the instruction can be executed. The first instruction in the case we are considering would be one telling the machine to copy the next two instructions from the input device being used into the third and fourth positions in memory. By continuing in this way we can eventually introduce the instructions needed to execute the problem itself, since each copy instruction will require one instruction but will call in two new instructions. This is really all there is to it, but if the reader is left with the feeling that the matter is still far from clear he will do well to read another paper in this issue[12] which describes a much neater solution to the loading problem and one which is in current use for one type of machine.

Types of Instruction Codes

You will note that the typical instruction must consist of at least two parts, an operation part which tells the machine what to do, and an address part. During the loading operation just described, the operation part specified that information was to be accepted at the input to the machine while the address part specified where the information was to be stored. In other instructions the address part might specify where in memory one must go to get a desired word, while the operation part might specify that the word be sent to the adder, for example, but in any case at least one address must be provided. It should be observed that the instruction always contains the address of the word on which the operation is to be performed, rather than the word itself. This simplifies the substitution of new

[12] W. Buchholz, "The system design of the IBM type 701 computer," Proc. I.R.E., pp. 1262–1275; this issue.

data into a problem since one can change the stored data without in any way altering the instructions. Machines based on this principle are called "Single Address Machines."

Some machines use instructions which do more than this. For example, one might have a "two address code," the first address being that of the word on which the operation is to be performed, and the second being the address at which the next instruction is located. If this is done, the machine does not require a program counter since it does not take instructions in the order they are listed but always in the order specified by these second addresses. As still another example one might have a "three address code," in which the first two addresses specify locations at which two desired words are to be found, and the third address specifies the location where the result of the operation (maybe the sum of the two original words) is to be stored. In this case, since no new instruction address is provided we would again depend on the program counter. Or again, one might design a machine using a "four address code" in which each instruction contains two input addresses, one output address, and the address of the next instruction. It is still a matter of some dispute as to which of these and various other possible systems is the best.

Kinds of Operations

We still have not said anything about the kinds of operations the computer can perform. Here there is even more variation from machine to machine. However, we can classify the instructions into several logical categories. In the first place there must be a few instructions relating to the input and output devices used with the computer. One must be able to specify that an input or output device be operated and that data be copied from the input or to the output. Closely related to this, there must be instructions enabling one to store information in memory or to recall it as desired. Then, of course, there must be a number of mathematical operations such as add, subtract, multiply and divide. And finally there must be a number of control transfer operations which enable one to transfer the control to some specified instruction, either unconditionally or subject to some specified condition. For example, we might have a "transfer on zero" operation which will cause the control to be transferred to the instruction stored at the specified address if the result of the previous operation is zero. This could be used to terminate the iterative loop we mentioned earlier. Other conditional transfer operations frequently encountered are "transfer on plus," "transfer on minus" with obvious significance, or "transfer on overflow": that is if a previous operation has led to a result which exceeds the capacity of the machine.

Virtues of the Stored Program

We can now go back and consider one property of the "stored program" method of operation which is rather unique and which really must be understood to appreciate the full value of such a concept. This property is that of being able to operate on the instructions themselves just as if they were ordinary data. This means that the entire course of a computation can be altered, including the operations themselves, the choice of data on which the operations are to be performed, and the location at which the results are to be stored, and this can all be done on the basis of results obtained during the course of the calculations through the use of conditional transfer instructions. The clever programmer need not write out every operation which he wishes the machine to perform. Instead he works out procedures for portions of the problem and then arranges a master program which calls for these "sub-programs" as required, perhaps over and over again, and which causes the machine to modify and adapt these sub-programs as required during various stages in the solution. Programming, in its more advanced form, thus becomes an art. Much of the apparent complexity of the computer resides in the program rather than in the machine itself. It is interesting to speculate on the way in which improved programming techniques are sure to increase the usefulness of existing computers beyond the fondest hope of their designers and how they are apt to influence the design of future computers.[13]

Limitations of Practical Machines

In concept, the modern digital computer is a universal machine since it can be made to simulate the behavior of any other special purpose machine, or to solve any problem which can be reduced to arithmetic. The practical machine falls short of this, both with respect to data handling capacity and speed.

Data handling capacity obviously refers primarily to the storage functions. It is a truism to say that better memory devices are needed. However, as intimated earlier, most practical storage devices have associated with them certain aspects of switching as well. What is really needed is a larger memory which contains a fast switching mechanism as an integral part. It is a lamentable fact that most memory devices with large potential capacity seem to involve slow switching mechanisms, while the faster switching devices are most easily associated with devices having limited storage capacity.

A complete discussion of the present day methods of handling this problem would lead us well beyond the scope of the present paper. We can note, however, that most modern machines employ a hierarchy of memory devices. A typical array might start with a punched card storage facility with substantially unlimited capacity but which is only capable of being "read" at say 150 characters a second. This will be supplemented with a magnetic tape storage facility which provides for "ordered" access to information at a very much faster rate

[13] For example, see G. M. Hopper and J. W. Mauchly, "Influence of programming techniques on the design of computers," Proc. I.R.E., pp. 1250–1254; this issue.

of around 10,000 characters a second, but which is of limited utility if information is wanted in an order differing from that in which it is stored on the tape. Here again the capacity may be substantially infinite but the larger the capacity the more the time required to sort through many, many reels of tape to select the desired reel and then to read through this tape from end to end to locate the desired bit of information. The next hierarchy might be magnetic drums which would store say 100,000 characters with an access time of several milliseconds. Finally, the machine might have a memory composed of electrostatic storage tubes which can store 40,000 characters with a truly "random" access of say 12 microseconds. This array of different mechanisms is the result of a practical compromise between speed, capacity and cost.

Some very exciting things are in the offing as far as new memory systems are concerned, in the form of improved memory tubes, magnetic cores used as memory devices and, even more speculative, the use of ferroelectric materials such as barium titanate.[14]

The next most serious limitation at present seems to relate to speed, but here the problem is not simply one of speed per se, but rather of speed as a function of cost. It is possible to strike a variety of compromises between these two factors as dictated by the requirements of the particular application. This is convincingly demonstrated by the existence of different types of arithmetic units. For example, a number of machines have been built on the so-called "serial" basis in which a relatively simple but fast type of adder may be used to add two words, bit by bit, usually proceeding from the least significant bit position and progressing with time toward the most significant position. Contrasted with this there are so-called "parallel" machines in which a complete array of individual adder units handle all the digit or bit positions at the same time. Finally, there are hybrid forms of machines, the so-called "serial-parallel" in which the bits comprising the binary code for each decimal digit are handled in parallel, but the digits themselves are then handled in series, and the inverse of this called "parallel-serial" operation. This problem of speed versus cost can, of course, be helped by the introduction of very much faster components. An alternate solution would be to develop components which will greatly reduce the cost, or at least devices which reduce the over-all size of the computer, its power consumption and its complexity, thus reducing its cost.

Here again, we seem to be just around the corner from significant developments. An increase in the use of diodes and the application of transistors are frequently cited as being very promising ways to reduce the size and power requirements. Printed circuit techniques offer possibilities in the way of economies in assembly, and there is a general trend toward miniaturization of almost all of the needed components. Miniaturization may also bring in its wake an automatic increase in speed, because of the lowering of stray capacitances, the decrease in lead inductances, etc. We can certainly look forward to a continual, although perhaps gradual reduction in the size and cost of data processing machines.

Of course, along with improvements of these sorts, we must always keep in mind the problem of machine reliability. As machines become more complicated and faster, error free operation becomes of dominating importance. A typical machine may perform additions, requiring the functioning of hundreds and even thousands of components, at the rate of say 16,000 additions per second. Such a machine has to be well nigh perfect if it is to operate for periods measured in hours or days without an error. The temporary malfunctioning of a single component for a period of less than one microsecond may completely invalidate the final result. As someone has put it, "Every bit counts." Unfortunately, such perfection is not easily achieved.[15] One popular way out of the difficulty is to utilize automatic checking procedures, or to carry along additional information in the form of so-called "redundancy bits" which will detect an error through a lack of internal consistency. In the extreme case, "error correcting codes" may be used which will give the correct answer in spite of the malfunctioning of some components. These schemes are not without drawbacks since the necessary equipment inevitably complicates the machine and may under certain conditions actually result in an increase in the frequency of error. There is no substitute for reliability of the basic components.

Another limitation and in many ways the most serious, since a straightforward engineering approach to its solution is not so obvious, has to do with the increasing demands which computers seem to place on the analytical processes of the men who "program" them or, more succinctly, on the ability of these people to think. Something must be done to transfer some of the load of the routine or semi-routine mental work associated with programming to the machine itself. The path is not clear but we urgently need machines or machine operating procedures which will solve problems that are stated in the more conventional forms of symbolic mathematics, or in terms of standard accounting practices, and thus eliminate the tedium of converting problems to a series of elementary arithmetic operations. A start has been made in terms of "compiling," "sub-programming," "symbolic programming," "program assembly" techniques and the like, but this is only a start.

Finally, we need to provide our computers with devices which will detect intermittent faults as they develop and which will indicate their location and thus reduce the Herculean task of finding trouble in a very

[14] For a survey of some existing memory systems see J. P. Eckert, Jr., "A survey of digital computer memory systems," Proc. I.R.E., pp. 1393–1406; this issue.

[15] The use of "diagnostic programming" as one means of causing a computer to help locate its own faults is becoming quite common. See, for example, D. J. Wheeler and J. E. Robertson, "Diagnostic programs for the ILLIAC," Proc. I.R.E., pp. 1320–1325; this issue.

complex machine. The ultimate along this line, at present only a pipedream, is a machine which is capable of maintaining itself.

Are Computers Here to Stay?

Perhaps it is time to call a halt. The writer cannot forbear a brief reference to the present-day importance of data processing machines. Regardless of what the reader may conclude regarding the long-range future of digital computers, the scientifically trained individual who is mildly interested in these devices and who does not follow up this interest is missing some of the more exciting technical experiences of the present time. In a similar vein, anyone who puts off the use of existing data processing machines because of their rapid development is simply denying himself or his organization the opportunity to profit from their present use and he is not training his personnel in thinking in terms of machine aids; in short, he is not being realistic. Present day computers are very exciting, yet they are down-to-earth, practical devices, and they operate unbelievably well.

This paper will have served its purpose if it has given the non-specialist reader a feeling that he has an understanding of the logical principles on which digital computers are constructed and that he is beginning to be acquainted with the specialized nomenclature which beclouds the issue. We have noted that the only thing really different about computers is their "physiology" since they are made of well known building blocks. A digital computer simply processes digital information. Furthermore, only two basic functions are involved, information switching and information storage, and these are easily explained in terms of such common things as an electric light switch and a piece of paper. It is true that these functions are combined into sub-assemblies such as input and output devices. a logical unit, a memory, and a control unit, but this is largely for convenience and in any case the names are nearly self-explanatory. Understanding programming is perhaps a bit more difficult, but fixed programming is already familiar in a variety of household devices, while stored programming is simply a scheme which permits the program information to be stored and operated on just as if it were any other kind of input data. An interesting aspect of this concept has to do with the way in which the subsequent course of a solution can be made to depend upon preliminary results through the use of conditional transfer instructions. Computers are really not so very difficult to understand after all, since most of the apparent complexity resides in the program rather than the machine itself. Finally, there is a great need for the development of new computer components which will increase the memory capacity, increase the speed and decrease the size and cost, although some very exciting things are in the offing. Computers are here to stay, and it is high time for us to be learning more about them.

Can Machines Think?*

M. V. WILKES†

Since the capabilities and the limitations of computers are topics of wide-spread interest, this paper by a well-known British authority on digital computers is reprinted here, with permission, from the May 1953 issue of *Discovery* (London, England). Prof. M. V. Wilkes was in charge of designing the EDSAC at Cambridge University, England, which had the distinction of being a very early stored-program type of electronic computer. Since then his group has pioneered in developing new programming techniques and in applying computers to a large variety of problems, which include experiments to find out whether a digital computer can be made to exhibit symptoms of "learning."—*The Editor*

THE SUBJECT of this article can be discussed on various levels, ranging from that of a sensational Sunday newspaper to that of a sober philosophical periodical. It arouses deep-seated emotions, and views are apt to be expressed with vigor.

Two contrary attitudes are common. In the first place there is a widespread, although mostly unconscious, desire to believe that a machine can be something more than a machine, and it is to this unconscious urge that the newspaper articles and headlines about mechanical brains appeal.

On the other hand, many people passionately deny that machines can ever think. They often hold this view so strongly that they are led to attack designers of high-speed automatic computing machines, quite unjustly, for making claims, which they do not in fact make, that their machines have human attributes. Such people are often misled by the use of technical terms based on physiological analogies; a good example is the use of the word "memory," for the part of the machine in which numbers are stored.

We must begin by deciding what a machine must be able to do in order to qualify for the description "thinking machine." An extreme view is that of Berkeley, who in his book *Giant Brains* says, after writing about cer-

* Decimal classification: 621.375.2. Reprinted from *Discovery* (England) vol. 14, p. 151; May, 1953.
† Director, University Mathematical Lab.; Cambridge, Eng.

tain automatic calculating machines: "A machine can handle information; it can calculate, conclude, and choose; it can perform reasonable operations with information. A machine, therefore, can think."

The use of the expression "reasonable operations" appears to beg the question, but from the examples that he gives it is clear that Berkeley means actions which depend on the power of the machine to compare two quantities and perform one set of operations if they are equal and another set of operations if they are unequal.

This, for example, would make it possible for the machine to consult a city directory which had been placed beforehand in its memory, and determine the block in which a house having a given address lay. However, other machines, such as automatic railway signaling systems, can perform this kind of "reasonable operation," and Berkeley's definition of what is meant by a thinking machine appears to be so wide as to miss the essential point of interest in the question, "Can machines think?"

An alternative approach is to say that a machine is a thinking machine if it can imitate a human being. There are plenty of existing machines which will do this in a limited field; for example, a doctor friend of mine has a device which will answer his telephone in his absence and repeat a message he has previously recorded. But again, to describe this as a thinking machine would strike most people as playing with words and no more.

Imitating Human Beings

It is, however, undoubtedly the power of some machines to simulate human behavior that people have in mind when they discuss this subject, and a machine which could pass itself off as a human being when given an extended test covering a wide range of different subjects might well qualify for the title "thinking machine." Among other things the machine would have to have some power of learning; for example, it should be possible for the examiner to play a game with the machine, having first explained the rules of the game.

Of course, the machine need not necessarily have the physical appearance of a human being, and in order to avoid any unfairness arising on this account the test could best be conducted with the examiner in one room and the machine in another, the connection between them being provided by a teletype circuit. The test would then take the form of a series of questions put by the examiner, with replies and counter questions automatically transmitted by the machine.

These ideas were first put forward by A. M. Turing in a penetrating article published in *Mind* (October 1950). The following example of a dialogue which might take place between the examiner and the machine is based on one given by Turing.

Examiner: Do you know the sonnet which begins, "Shall I compare thee to a summer's day?"

Machine: Yes.

Examiner: Would not "a spring day" do as well or better?

Machine: It wouldn't scan.

Examiner: How about "a winter's day"; that would scan all right.

Machine: Yes, but nobody wants to be compared to a winter's day.

Examiner: Would you say Mr. Pickwick reminded you of Christmas?

Machine: In a way.

Examiner: Yet Christmas is a winter's day, and I do not think Mr. Pickwick would mind the comparison.

Machine: I don't think you're serious. By a winter's day one means a typical winter's day, rather than a special one like Christmas.

The questions put by the examiner are supposed to be entirely unpremeditated so that there is no possibility of the answer being built into the machine on some sort of record. It is a fantastic suggestion that a machine should be able to carry on a conversation of this sort and many people will be inclined to reject it out of hand. Certainly no existing machine is in the least capable of doing anything of the kind.

I would agree with Turing, however, that the ability to pass this test or something like it is what people mean, or ought to mean, when they talk about the possibility of a machine thinking. If ever a machine is made to pass such a test it will be hailed as one of the crowning achievements of technical progress, and rightly so. Whether everyone will then agree that the machine thinks is another matter; I suspect that to many people thinking means something which can be done by a human being, or possibly by an animal, but not by a machine.

This, however, is leading us from a discussion of what machines can do to an entirely different question, namely, "Is the brain a machine?" This is a question on which science has hardly begun to touch and the answer which would be given to it by any particular person would depend on his philosophical beliefs, particularly as to the relation between mind and matter. Indeed from one point of view, to say that the brain is a machine is equivalent to denying altogether, as some philosophers do, the distinction between these two things.

In some ways it might be said that people now tend more than they formerly did to regard human beings in the same light as machines. For example, the prevalent attitude to the delinquent is to regard him as the victim of circumstances and to say that he requires treatment rather than punishment. This is just the attitude one takes to a machine. One does not get cross with it when it does something wrong, but one looks for and rectifies the fault.

Automatic Digital Computers

The subject of this article is not a new one, but its discussion has been greatly stimulated by the development of high-speed automatic digital computing machines, since these machines are capable of performing operations of much greater complexity than any machines hitherto available. I shall therefore give a description of their mode of operation with special reference to those properties which seem particularly relevant to the present discussion.

The basic operations performed by a digital computer are very simple—addition, subtraction, multiplication and such like—but the machine can be set up so as to perform a long sequence, or *program* as it is called, of these operations one after the other. The complexity which so impresses the layman is in fact a feature of the program, rather than of the machine, and the machine can be changed over from one kind of work to another by changing the program.

For this reason machines of this kind are sometimes known as *universal* machines. Given a suitable program a universal machine can do anything which could be done by a specially built machine; for example, it can simulate the behavior of an analog device incorporating feed-back. It is therefore convenient to discuss the problem of mechanical thinking in terms of writing a program for a universal machine, rather than in terms of building a special machine.

It is, of course, assumed that the universal machine is provided with a suitable output organ for the work it has to do; for the purposes of discussion this can be a printer capable of printing figures and letters. A machine built specially for experiments in mechanical thinking would need to have a very large memory, or "store" as it is perhaps better called, and its basic operations might be different from those of a machine intended for mathematical work, but the relation between mathematics and logic is so close that the differences would be unimportant in general discussion.

Some of the basic operations which can be performed by a digital computer are "conditional"; that is, their nature can depend on whether a certain number (perhaps one that the machine has just computed) is positive or negative, or whether one number is greater or less than another number. The effect of a conditional operation is that a choice is made in regard to the subsequent action of the machine; if the condition is not satisfied the machine follows a course of action specified by one section of the program, whereas if it is satisfied the machine follows a different course of action specified by some other section of the program.

As an example of the use of a conditional operation, I will give a program for calculating a square root by a method of successive approximation. Suppose we want to find the square root of the number N, and suppose we have a first approximation x_0; then a closer approximation to $\sqrt{N}$ is given by $x_1=\frac{1}{2}(x_0+N/x_0)$. A still better approximation can be obtained by applying the formula a second time, using the old value obtained for x_1 as the new value of x_0. This process can be repeated until it is found that the latest approximation does not differ significantly from the previous one, i.e. until x_1 and x_0 differ by a quantity which is not greater than ϵ (say). The numbers N, x_0 (the first approximation) and ϵ must be placed beforehand in the store of the machine. The store of a large machine can hold many hundreds of numbers and it is usual to number, for easy reference, the "locations" in which they are held. It will be supposed that N, x_0 and ϵ are respectively placed in storage locations 100, 101 and 102.

The program then consists of the following sequence of operations:

No. of operation in sequence	Operation
1	Copy number in 101 into 103
2	Divide number in 100 by number in 101, and put result in 104
3	Add number in 101 to number in 104, and place result in 104
4	Divide number in 104 by two, and place result in 101
5	Subtract number in 103 from number in 101, and place result (neglecting sign) in 105
6	Subtract number in 105 from number in 102, and place result in 106
7	If number in 106 is negative, go back to operation 1; otherwise go straight on to operation 8
8	Print number in 101

Here the *seventh* operation is the conditional one. This short program might form part of a longer program in which the calculation of a square root was necessary.

Discrimination

Clearly, conditional operations are very important in the present context since they give the machine a power of discrimination. The possibility of programming a machine to consult a directory, as in Berkeley's example, depends on the use of conditional operations. However, when a machine performs a conditional operation it can be said to think just to the same extent, and no more, that a subway train can be said to think whenever it approaches points which have been set automatically by the passage of a previous train, and goes in one direction rather than another. The use of the word *think* in connection with conditional operations would be justifiable only if the use of a convenient technical term were thereby secured. I know of no computing machine laboratory where the word is used in this way, although the somewhat comparable words *decision* and *discrimination* are sometimes used.

Before a program such as that given above can be put into the machine it must be expressed in a form suitable for the input device with which the particular machine is equipped. Often this means that it must be punched on paper tape like a teletype message; sometimes it must be punched on cards.

In the modern high-speed machines the whole program, or a considerable section of it, is taken into the machine before any of the operations are actually carried out. Since calculations performed on a high-speed machine always involve a great deal of repetition (as does the program given above), this makes for faster operation than would be possible if the input tape or cards had to be read afresh each time a given set of operations were to be performed.