More accurate values could be obtained from NBSCL, *Tables of Probability Functions*, v. 2, 1942. In applications one usually requires differences $\phi(b) - \phi(a)$ rather than values of $\phi(x)$. Such differences can now be obtained by a single reading, putting $x = \frac{1}{2}(b + a)$ and $l = b - a$.

W. FELLER

74[L].—HERBERT E. SALZER, *Table of* $\Gamma(n + \frac{1}{2})$, ms. in possession of the author, NBSCL, 150 Nassau St., New York City.

This is a manuscript of $\Gamma(n + \frac{1}{2})$, $n = [0(1)1000;8S$, guaranteed to 7S]. The table can also be used to obtain $\Gamma(-n + \frac{1}{2})$ by a single division, from the relation $\Gamma(-n + \frac{1}{2}) = (-1)^n\pi/\Gamma(n + \frac{1}{2})$.

H. E. SALZER

AUTOMATIC COMPUTING MACHINERY

Edited by the Staff of the Machine Development Laboratory of the National Bureau of Standards. Correspondence regarding the Section should be directed to Dr. E. W. CANNON, 418 South Building, National Bureau of Standards, Washington 25, D. C.

TECHNICAL DEVELOPMENTS

Our current contributions under this heading appear in the earlier part of this issue. They are "The memory tube and its application to electronic computation," by ANDREW V. HAEFF, and "The logical design of the Raytheon Computer," by R. M. BLOCH, R. V. D. CAMPBELL & M. ELLIS.

DISCUSSIONS

General Design Considerations for the Raytheon Computer

The basic logical units of a general-purpose automatic digital computer are: first, a memory for storing numbers and coded instructions; second, an arithmetic unit for performing addition, subtraction, multiplication, division, and other required operations; third, a central controlling facility for directing the machine in accordance with the coded instructions; and fourth, auxiliary devices required for transforming the input data into a form suitable for the machine, and for recording final results.

In most machines, there are two types of memory provided: a unit of relatively low storage capacity in which any selected item in the unit may be obtained quickly; and another unit, having a much larger capacity, but from which items can be obtained quickly only if they are used in a preassigned order.

In the design of computers, there are a number of basic decisions which together largely characterize the machine. These decisions relate to the manner of representing numbers and instructions inside the machine, the basic or built-in operations provided in the arithmetic unit, and the framework within which machine operations are scheduled.

Representation of Numbers.—In digital machines, the radices 2 and 10 are almost universally used for number representation. Normally the digits used are non-negative, except possibly during the performance of certain operations such as multiplication and division.

The use of the radix 10 has the advantages that numbers and arithmetic processes are in a familiar form. The normal form of the digits of the decimal representation is not necessarily preserved, however, since in many machines it is necessary to use a coded decimal notation. Thus, if bi-valued storage elements are used for number representation, a set of at least four such elements must be used to obtain 10 distinct configurations. The binary system makes for simpler arithmetic processes than does the decimal or coded decimal and frequently requires less equipment for the storage of numbers.

If a digital computer is used as part of a control device, it may receive its numerical input data from instrument readings. Such readings, if in continuous form, must be converted to discrete form before being introduced into the machine. The conversion to binary should be no more difficult than the conversion to decimal. For control purposes it may be required to convert the output of the computer back into continuous form. Here again, using the binary system rather than the decimal within the computer creates no additional conversion problems.

By contrast, a digital computer operating from data which are provided in decimal form, and required to deliver a decimal output, must have facilities for automatic conversions in scale of notation if the binary notation is used internally. Methods of conversion will be discussed later.

In any scale of notation there are two commonly used methods for representing numbers. Let the radix used be r. Then, in the first, or fixed-point method, all numbers X in the machine are of the form $X = \frac{X}{|X|} \sum_{j=a}^{b} x_j r^j$, where a and b are constants of the machine,[1] and $0 \leq x_j \leq r - 1$. For example, in a decimal machine, a and b might assume the values -10 and -1 respectively, and all numbers in the machine would lie in the range: $-0.99999\ 99999 \leq X \leq 0.99999\ 99999$. In the second, or floating-point method, numbers are represented as $X = X_0 r^n$. Here, $r^b \leq |X_0| \leq r^{b+1} - r^a$, where a and b are integral constants, and n is an integer such that $c \leq n \leq d$, c and d being integral constants. In a decimal machine, typical values for a, b, c, and d are -10, -1, -15, and 15 respectively. In this case, $0.10000\ 00000 \leq |X_0| \leq 0.99999\ 99999$ and $10^{-16} \leq X < 10^{15}$. It should be noted that special provision must be made for representing the number zero in the floating-point arrangement.

In either of the above methods of representation, the precision of the representation is $b - a + 1$ digits or columns. Many machines are provided with equipment for operating at two different precisions—a low precision which is usually standard, and a high precision, double the low one, which is the less frequently employed mode of operation.

A negative number can be stored either as an absolute value and a minus sign, or as the complement of the absolute value. The complement of a positive number X is $Z = K - X$ where K is a constant. Two commonly used values for K are r^{b+2} and $r^{b+2} - r^a$. The use of complements may require some modification of the rules of arithmetic.

In fixed-point digital machines the use of scale factors (or, in many cases, linear transformations) is essential, and may for some problems involve many complications. The scale factors are needed to insure that all variables employ as large a fraction of the useful range of number representation as possible, without exceeding this range. If, for example, a variable X is restricted to the range $0 < X < 100$, the transformation $X' = \frac{1}{50}(X - 50)$ might be employed, such that $-1 < X' < 1$. Floating-point operation greatly reduces the need for scale factors, but complicates the operations of addition and subtraction.

The position of the radix point in the fixed-point representation is a matter of some importance since it determines what combinations of operand values are permissible for multiplication and division. If $b \leq -1$, the product of any two permissible numbers is likewise a permissible number; with $b < -1$, however, precision is needlessly lost. For $b \geq 0$, certain products are excluded. Thus, from the standpoint of multiplication alone, $b = -1$ is the best value available. In the case of division, the number of excluded cases decreases as b is increased. For $b = -1$, all cases are excluded for which the divisor is not larger than the dividend; larger values of b are thus suggested.

It should also be mentioned that even the integer 1 cannot be placed in the machine without using a scale factor if $b < 0$.[2]

Arithmetic Operations.—So far, little attention has been devoted to the decision as to which operations the arithmetic unit should be designed to handle. In this connection, a distinction must immediately be made. Some operations can be initiated by a single signal from the central control unit—these will be called built-in operations. Others will be compounded of such built-in operations. If a basic set of built-in operations is furnished, a wide class of problems can be handled by choosing a proper sequence of coded instructions.

A general-purpose automatic digital computer evidently requires some means of performing—(1) addition; (2) subtraction; (3) multiplication; (4) division; (5) extraction of square roots; (6) calculation of algebraic and transcendental functions; (7) selection of one number from a set of numbers according to the value of a controlling argument; (8) sorting, ordering, and collating of data. Moreover, the machine must have the ability to modify instructions in accordance with partial results obtained in the computation. Finally, if the machine operates in a nondecimal scale of notation, it may also be required to perform conversions from one scale of notation to another.

The minimum list of built-in operations which could be considered reasonable for most applications[3] might be adding, complementing, shifting right, and selecting. These may be defined symbolically as: adding: $X + Y = Z$; complementing: $K - X = Z$, where K is a constant; shifting right: $X \cdot r^{-1} = Z$, where r is the radix; selecting: if $X \geq 0$, select Y, if $X < 0$, select 0—call the result Z. Multiplication and division, as well as all other operations, would be compounded from the above four. For convenience in preparing problems for a general-purpose machine, however, a much larger list of basic operations should be provided.

Regardless of which operations are selected as built-in operations, the four operations listed above are usually the basis of the design of the arithmetic unit. Thus, subtraction is usually performed by adding the complement of the subtrahend to the minuend; multiplication—particularly in a binary machine—is simply a combination of additions, shifts, and selections.

Division, the extraction of square roots, and the calculation of simple functions can all be performed in a number of different ways in terms of the above operations. Division, for example, can be performed by subtraction, selection, and shifting. Or the reciprocal of the divisor can be computed by an iterative process. If X is the number whose reciprocal Y is desired, and Y_0 is a first approximation to Y such that $0 < |Y_0| < 2|Y|$, and if Y_0 and Y are of the same sign, then the iteration of the process $Y_{n+1} = Y_n(2 - XY_n)$ converges to Y.

If floating-point operation is included, either as a built-in or as a compounded operation, additional features are required in the arithmetic unit. This is also the case if numbers of twice normal precision are to be handled.

If conversion in scale of notation is required as a machine operation, it can be done by several methods. Consider the conversion of a number from the scale of notation with radix r to that with radix s. In the first scale of notation write the number as

$$X = \frac{X}{|X|} \sum_{n=n_1}^{n_2} x_n r^n (n_2 > n_1)$$

and in the second, as

$$Y = \frac{Y}{|Y|} \sum_{m=m_1}^{m_2} y_m s^m (m_2 > m_1).$$

The additions and other computations required in the conversion can be performed in either scale of notation. If they are performed in the r notation, the fractional part of X may be converted by successive multiplications by s expressed in the r notation. After each multiplication, the integral part is equal to one of the required digits y_m of Y; this integral part is truncated before the next multiplication. The integral part of X is converted by a method which is similar to the above but employs divisions by s in place of multiplications by s.

In order to perform the calculations in the s notation, one uses a slightly different method. One converts the digit of lowest order in the fractional part of X—viz., the digit for $n = n_1$—to the s notation, and then multiplies by $1/r$ expressed in the s notation. Next one adds the equivalent of the digit of second lowest order, and again multiplies by $1/r$. The alternate addition and multiplication continues until all of the digits of the fractional part of X have been used. The integral part of X is converted by alternate addition and multiplication by r. Here one starts with the digit of highest order.

Alternative conversion processes are available which make use of either the s-equivalents of $x_n r^n$ or of the r-equivalents of $y_m s^m$. The equivalents, in this case, are supplied to the calculator as part of the input data for each problem.

Representation of Orders.—Not only input numbers, but also coded instructions must ordinarily be introduced into the computer at the start of the problem. When superimposed upon the logical structure of the machine, these instructions must complete the formulation of the problem.

Consider the computation of $Z = (AY + B)X + C$, in which the numbers A, B, C, X, and Y are stored in the memory, and the result Z of this composite operation is to be sent to the memory. The computation requires the locating of the five input numbers in the memory and their transfer to the arithmetic unit, where operations are performed upon them. It also requires the specification of two multiplications and two additions. The quantity Z must be sent to the proper memory location, since that location will serve to identify Z for future use. Finally, all of the foregoing processes must be performed in the correct sequence. The performance of these multiplications and additions, as well as number transfers between the memory and the arithmetic unit, requires the use of certain coded instructions which must be selected from the memory in the proper order and must then be used to activate the necessary controls.

In any computing process, the required operations are usually performed as a succession of rather simple steps. The entire amount of instructional data needed to formulate each step is selected from the memory as a single unit, which is called an order. A number of different schemes can be used for the breakdown of the computation into steps, and the corresponding breakdown of the instructional data into orders. If the basic step consists of a transfer between the memory and the arithmetic unit, together with the possible initiation of an arithmetic operation, the computation of Z above requires six steps: five to send the input data to the arithmetic unit, and one to return the result to the memory. Another type of basic step would include a complete multiplication or addition and the necessary number transfers. Then the computation of Z would consist of four steps. Still other basic steps could be constructed involving, for example, a pair of number transfers, or even a pair of arithmetic operations, such as an addition and a multiplication. The various choices of basic steps will evidently correspond to different types of orders.

Orders as well as numbers must be selected from the memory. Orders are then sent to the central controlling unit of the computer. The sequence in which the orders are chosen from the memory can either be preassigned as part of the logical structure of the machine, or it can be specified by the orders themselves. If a preassigned sequence is used, provision must be made for a departure from this sequence under special circumstances. The storage of orders in the internal memory generally permits the use of either method of order selection. Order storage in the external memory, however, requires that the orders be used in a preassigned sequence—namely, that in which they are recorded on the external storage medium.

Orders must contain at least two types of coded instructions—an instruction governing the transfer of a number between the memory and the arithmetic unit, and an instruction initiating one of the built-in operations of the arithmetic unit. To specify a transfer, one must locate each terminus and state whether the number is to enter or leave the memory. All of this information need not be explicit in the order, however. The specification of a location may be called an *address code.* The codes for these locations in the internal memory are conveniently assigned as consecutive integers expressed in the same scale of notation as that used for the actual numerical quantities. Data are normally taken from or routed to the external memory sequentially; hence, the address code of a given location need only specify which of several memory units is involved. Special instructions for nonsequential use of the external memory should also be included.

Since it must be possible to modify orders in accordance with partial results obtained, it is desirable to handle orders in the arithmetic unit in the same manner as numbers; this, in turn, requires certain similarities in their respective modes of representation.

The simplest type of machine organization is based upon completely sequential operation. The several processes controlled by each order take place in succession, and no two orders are operative simultaneously. Such serial operation uses a minimum of equipment and simplifies the operation and testing of the machine.

Higher speeds can be obtained through the scheduling of two or more operations simultaneously. Thus, for example, two transfers, or a transfer and an arithmetic operation, can be scheduled simultaneously. Composite computers, containing two or more arithmetic organs, enable several arithmetic processes to be performed concurrently.

Another type of choice is available in the scheduling of machine processes. Each basic step can be allocated either a length of time which varies with the nature of the step, or one which is fixed. The latter mode may simplify the controls for certain types of machine construction but is somewhat wasteful of operating time, since all steps must be allowed the maximum length of time whether it is needed or not.

A sample logical design will illustrate the nature of the alternatives just discussed. Each order will be taken to specify one number transfer and, when required, the initiation of an arithmetic process. Successive orders are normally taken from successive locations in the memory; hence, no explicit instructions for the locations of orders need be given, except in special cases. Each order consists of a memory address code and an operation code. The latter, taken in conjunction with the address, completes the specification of the transfer. Using this type of order, the sample calculation $Z = (AY + B)X + C$ could be represented by the following order sequence:

Number of Order	Address	Operation Code
1	Address of A	Transfer to arithmetic unit.
2	Address of Y	Transfer to arithmetic unit; multiply by A.
3	Address of B	Transfer to arithmetic unit; add to above product.
4	Address of X	Transfer to arithmetic unit; multiply by above sum.
5	Address of C	Transfer to arithmetic unit; add to above product.
6	Address to which Z is to be delivered	Transfer above result from arithmetic unit to memory.

It is assumed in the above sequence that the arithmetic unit is capable of remembering partial results (e.g. the product $A \cdot Y$) between arithmetic operations.

Checking and Errors.—Four kinds of errors (apart from human errors) may occur in digital computation: 1. Errors due to inaccuracies in the numerical input data; 2. Round-off errors occurring in multiplication and division, and possibly also in other operations; 3. Errors due to the inadequacies of the computing procedure (often called truncation errors); 4. Errors due to malfunctionings of the machine.

Only errors of the fourth type will be considered here, and the term error will be restricted to cover this case only.

If a computer is to be of value, its error frequency must be sufficiently low, and the average time to diagnose and repair troubles sufficiently short, to enable the machine to be operated successfully a large fraction of the time. When the computer does make an error, it is essential that the error be detected, so that no wrong results are obtained. A low error frequency and a short repair time can be achieved by having a minimum amount of equipment, and by careful engineering and packaging. The detection of errors (i.e. checking) and error diagnosis can be conveniently considered together.

Two general types of checking may be distinguished: first, checking which requires changes in the computing routine but practically no increase in machine equipment; and second, checking which depends primarily upon the addition of extra equipment. It should be observed that the use of either type of checking tends to cause an increase in the number of errors per problem.

In applications where machine stoppage is extremely serious, even if the failure is detected and diagnosed, the use of extra equipment or extra computation for checking purposes may not be desirable unless it is essential to know whether a failure has occurred. If, however, the error frequency can be kept to a reasonably low figure, and a temporary shut-down of the equipment is not too serious, checking is ordinarily very worthwhile. Mathematical checks (i.e., those involving only an increase in computing time) seem to be of somewhat limited application. If the mathematical formulation of the problem is made self-checking, little diagnosis is ordinarily obtained, and in many cases the volume of the computations is doubled. Test problems can be made diagnostic, but unfortunately these are not necessarily adequate checks; all of the conditions relevant to the behavior of the machine on the main problem are not necessarily reproduced in the running of the test problem. Thus, if both checking and diagnosis are desired, there appears to be a distinct advantage in building the checks into the equipment.

In the extreme case where it is desired to have the machine not only locate its own errors but also correct them and continue automatically, a rather considerable amount of extra equipment may be required. On the other hand, a reasonable amount of checking and diagnosis, without the feature of self-correction, need not involve large quantities of equipment.

In a self-checking machine each number-word must evidently contain information not only as to the value of the number represented, but also as to its correctness. The set of all possible words contains a subset of valid words—i.e., words in which the criterion for correctness is satisfied; all other possible words are invalid. An elementary example of a self-checking method of number representation involves using complements. Each valid number-word consists of an n-digit binary number, together with its n-digit complement. If the second number is not the complement of the first, the word is invalid. There are thus 2^{2n} possible words, of which 2^n are valid.

A method of checking the storage and transfer of words has been devised which detects a large fraction of the machine errors but does not require much extra storage capacity. Let an unchecked number X of 35 binary columns be expressed as

$$X = \frac{X}{|X|} \sum_{n=0}^{34} x_n 2^{n-35},$$

where x_n is either zero or unity. Consider the auxiliary number formed by taking the weighted sum of the digits[4] x_n of X, viz.,

$$X_c = \sum_{n=0}^{34} x_n w_n 2^{-35}.$$

The auxiliary number X_c is stored with the original number X to form the complete number-word and will be termed the weighted count of X. The validity of a word after storage and transfer is checked by obtaining a new weighted count and comparing it with the one obtained previously and stored in the word. The weighted count is most conveniently calculated using weights which are all powers of 2, although not necessarily the same power.

By a judicious selection of the weights w_n, an arithmetic check can also be obtained. Let the w_n be 1, 2, 4, 8, 16, 1, 2, 4, 8, 16, 1, 2, 4, 8, 16, $\cdots$ for $n = 0, 1, 2, 3, \cdots$ respectively. Thus the weights repeat in groups of five, and within each group are just the successive powers of two. It can easily be seen that the weighted count X_c is merely $\sum_{m=1}^{7} y_m 2^{-35}$, where the quantities y_m are the digits of X when X is written in the scale of notation with radix 32. It therefore follows that $X - X_c = 31K \cdot 2^{-35}$, where K is a non-negative integer. Thus, the weighted count of the sum of two numbers differs from the sum of their weighted counts by a multiple of 31. This fact can be used to check the addition process in the arithmetic unit; the other arithmetic operations can also be checked by means of the weighted count.

The checking of selection can be accomplished briefly as follows. The machine is so arranged that whenever one device in a group of N is selected, that device sends back a tag or identification signal. The tag can be compared with the portion of the instruction which governed the original selecting process, and any discrepancy will indicate a false selection.

R. M. Bloch, R. V. D. Campbell & M. Ellis

[1] In some machines, provision is made for changing the parameters a and b between problems.

[2] See S. LUBKIN, "Decimal point location in computing machines," *MTAC*, v. 3, p. 44–50.—EDITOR.

[3] It is here assumed that to build up all processes from the basic operations of logic would require far too many orders in the formulation of problems.

[4] The algebraic sign of the number is also taken into consideration in forming the weighted sum X_c; this has not been indicated in the formula, however.

BIBLIOGRAPHY Z-V

1. ARTHUR W. BURKS, "Electronic computing circuits of the ENIAC," Inst. Radio Engineers, *Proc.*, v. 35, 1947, p. 756–767; illustr. 27.9 × 21.6 cm.

The ENIAC "is a very large device (containing 18,000 vacuum tubes) compounded out of a few basic types of computing circuits. The design principles that were followed in order to insure reliable operation of the electronic computer are presented, and the basic types of computing circuits are analyzed.

"Most of the design work on component circuits was devoted to constructing reliable memory circuits (flip-flops) and adding circuits (counters). These are treated in detail.

"The ENIAC performs the operations of addition, subtraction, multiplication, division, square-rooting, and the looking up of function values automatically. The units which perform these operations, the units which take numerical data into and out of the machine, and those which control the over-all operation are described.

"The technique of combining the basic electronic circuits to perform these functions is illustrated by three typical computing circuits: the addition circuit, a programming circuit, and the multiplication circuit."

2. GREAT BRITAIN, POST OFFICE ENGIN. DEPT., Research Report no. 12719: A. W. M. COOMBS & W. W. CHANDLER, *Automatic Computing: An Analysis of Arithmetical Operations*. Post Office Research Station, Dollis Hill, London, N. W. 2, Aug. 1946. 29 p. + 1 plate. Mimeographed. 22.2 × 27.9 cm.

3. STIG DJURE & EINAR WELIN, "L. M. Ericssons Nya Automatiska Kalkylator for Regleringsändamål," [L. M. Ericsson's New Automatic Calculator for regulating outputs], *Teknisk Tidskrift*, 1944, fasc. 48, 32 leaves, 55 figures; French resumé. 20.3 × 29.2 cm.

Given a certain number of functions of time, $a(t)$, $b(t)$, $c(t)$, the computer under consideration yields the values of certain functions $f_1(a, b, c \cdots)$, $f_2(a, b, c \cdots)$, etc. when t varies continually, and without appreciable time loss. The analytical expressions of the "f" functions can contain derivatives, integrals, elementary functions; they can also be given as implicit functions of $a, b, c, \cdots$ or as the solutions of a system of equations. An example of a system treated on a machine of this type is presented.

It must be understood that a machine adjusted to solve a certain system of equations will solve only this system. The possible standardization of the machine elements will, however, make it adaptable to other similar systems.

4. ADELE K. GOLDSTINE, *Report on the ENIAC* (developed under the supervision of the Ordnance Department, United States Army), University of Pennsylvania, Moore School of Electrical Engineering. *Technical Report* I, bound in 2 v. (v. 1—chaps. I–VI; v. 2, chaps. VII–XI) June 1, 1946, xxxvi, 301 p.; 122 figs. Mimeographed. 21.6 × 27.9 cm. See *MTAC*, v. 2, p. 97–110, an article by H. H. GOLDSTINE & A. K. GOLDSTINE.

Without going into the details of the circuits, this Report covers the functions of the various units of the ENIAC and their integration. It thus serves as the logical introduction to the machine. Those who are not interested in circuit details nor in maintaining an electronic computer need only read Technical Report I for a complete understanding of the logical design of the ENIAC. The programming of the machine is quite thoroughly covered with several illustrative problem setups.

The 6 chapters in v. 1 cover, besides an introduction, the initiating unit, the cycling unit, the accumulator, the multiplier, and the divider and square root unit. A brief general program control with relation to the various units is discussed concurrently with the units themselves. The organization of the material, therefore, differs from the more recent discussions of computers, such as the one by BURKS, GOLDSTINE, & VON NEUMANN, *Preliminary Discussion of the Logical Design of an Electronic Computing Instrument* (see *MTAC*, v. 3, p. 50–53, 128). In this latter the major divisions are input, output, memory, arithmetic unit, and control. The ENIAC itself is not "organized" into these units. It is probable that the logical separation of a computer into these five organs was not made until the ENIAC was nearly complete.

Numbers are represented in the ENIAC by parallel trains of pulses. Ten decimal digits, the normal precision, are represented by trains of from 0 to 9 pulses along 10 wires. An eleventh wire carries either no pulses, representing a positive sign, or nine pulses indicating that the ten digits represent a complement with respect to 10^{10}. Thus, to subtract X from y, $10^{10} - X$ is added to y. The 9 on the sign wire is more properly called a complement indicator. The pulses in the unit train "step" a 10-stage ring counter. When the unit counter steps from 9 to 0, a carry-over to the 10's counter takes place. The complement procedure eliminates the necessity for these counters to step backwards as well as forwards.

There are 20 accumulators, which provide high-speed storage as well as facility for addition and subtraction. Each accumulator has five numerical input channels through any one of which it can receive a 10-digit signed number. There is also provision for shifting so that an accumulator can receive a number which has been shifted a fixed number of places to the left or right. Since repeat controls are provided for additions involving the numbers 1 to 9, multiplication by a constant can be effected without the use of the high-speed multiplier.

The sequencing of the ENIAC is controlled both locally and centrally. All the units of the machine, except parts of the input and output, operate synchronously, based upon a cycle of 20 pulses of the central clock, which emits pulses at the rate of 100,000 pulses per second. A program pulse must stimulate an accumulator to transmit or receive. When an operation is complete, always an integral number of cycles after starting, a program pulse is emitted, causing the next operation to begin. Thus, a sequence of operations can be carried out by plugging together various units and associated program controls. Since there are multiple channels, several of these sequences can be carried out simultaneously. These program sequences are tied together through the use of the master programmer. Hence, the final program pulse of a sequence may be delivered to the master programmer, and another sequence can pick up its initial pulse from the appropriate output of the master programmer. Sign or digit pulses can be used to stimulate program controls. This means that the machine can make a decision between two sequences based on earlier computation. This ability to link program sequences by discrimination makes the ENIAC control system extremely flexible.

The high-speed multiplier uses two of the accumulators so that the complete product of two 10-digit numbers can be obtained. In order to obtain speed, a one-digit multiplication table is provided. That is, if a 7 comes in on the unit line from one number, and an 8 on the unit line from the second number, a 6 is transmitted out on the unit line and a 5 on the 10's line.

The ENIAC is provided with a division unit, which operates in conjunction with a group of accumulators to hold the divisor, dividend, and quotient and do the necessary shifting. Division is accomplished by successive subtraction and testing for over-draft. The same unit can also extract the square root of a number. The square-root procedure utilizes the relation: $a^2 = \sum_{i=1}^{a} (2i - 1)$.

This volume is interesting reading for those concerned with large-scale electronic digital computers, even though several features of the ENIAC have been discarded in designing some of the computers yet to be constructed. However, the technical discussion is not recommended to the casual reader because of its detailed treatment of all units. For a summary treatment the reader is referred to, among others, an article by Professor D. R. HARTREE, in *Nature*, Oct. 12, 1946 (see *MTAC*, v. 2, p. 222; also p. 97–110).

V. 2 of the report contains detailed operating instructions for several units not discussed in v. 1. The contents of each chapter are briefly reviewed below.

VII. Function Table.—The ENIAC contains 3 function-table units, each of which can be used to store 104 12-digit functional entries against a normalized independent variable or location parameter. The stored information is entered manually on switches and can be made available to other units of the ENIAC in as little as 5 addition times (1 millisecond). A single reference to the table may be made to yield not only the tabular value corresponding to the argument specified but also the 4 neighboring entries as well. This characteristic facilitates the use of polynomial interpolation formulae of fourth degree or less. The author gives explicit setting-up instructions for carrying out quadratic and biquadratic Lagrangian interpolation. Higher-degree approximations are practicable but require more elaborate programming.

The storage of ballistic drag functions in the Function Table is described in some detail. Various artifices must be employed to circumvent difficulties which arise on account of the anomalous behavior of such functions in the neighborhood of sound.

The possibility of storing programming instructions in the Function Table is touched upon only briefly in this report. Experience subsequent to the publication date (June 1946), however, has shown that storage of instructions in the Function Table can achieve considerable savings in machine set-up time and labor, although at the cost of somewhat slower machine operation.

VIII. Constant Transmitter.—The Constant Transmitter unit, operating in conjunction with an IBM card reader, provides a special auxiliary memory for the ENIAC. As many as 80 digits may be read into the Constant Transmitter from a standard IBM punch card in about one-half second. Once stored in the Constant Transmitter, these data (in addition to 20 digits and 4 signs manually set up on switches) may be referred to repeatedly by other units of the machine, the access time in each case being 1 addition time (200 microseconds). Extensive set-up details are given for utilizing this valuable feature of the machine.

IX. Printer and IBM Gang Punch.—The output equipment associated with the ENIAC consists of modified IBM units. Detailed operating instructions for plug-board wiring, switch setting, etc., are provided.

X. Master Programmer.—The over-all direction of the ENIAC's operations is under the control of the Master Programmer. This central control unit supervises the progress of the computing program and issues instruction signals which stimulate the various subsidiary organs to perform their specialized subroutines. For example, the Master Programmer is needed for controlling the iteration of a sequence of operations constituting a program chain and for linking various chains together to form subprograms of higher order. The Master Programmer can be made to link programs together either in some preassigned order or on the basis of magnitude or sign discrimination. As a typical example of the latter process, the writer describes a step-wise integration trajectory computation. In this case detailed printed results are desired only at the summit of the trajectory and at the ground. At each step the shell's altitude coordinate and its first (time) derivative are tested to see whether they lie inside a certain range close to 0. If either one is found to fall within specified critical limits, the Master Programmer is made to issue signals which cause the program to enter into the appropriate new phase.

XI. Transmission Systems.—In this final chapter is a discussion of the three principal types of dynamic communication between units of the ENIAC. These are (1) communication by means of synchronizing signals; (2) transmission of numerical data; (3) transmission of program instructions. Detailed charts and block diagrams are provided to illustrate various typical and special interconnection schemes.

General Remarks.—V. 2 is essentially an operating instruction manual for setting up those phases of the ENIAC's problems which involve the units discussed above. The emphasis throughout the volume is on logical relationships, not electronic circuitry. In fact, little or no acquaintance with electronic nomenclature is required on the part of the reader. Because of the vast wealth of detail presented, however, it seems unlikely that the report will attract many thorough readers outside of the group actually engaged in performing manipulations on the ENIAC. Nevertheless, the reader with less specialized interest who is familiar with desk computing machinery should find even a cursory inspection of v. 2 quite rewarding. For one thing, it will convince him that not even the superhuman speed and versatility of the ENIAC can relieve the human operator of the burden of tedious manual routine ordinarily involved in carrying out an extended program of numerical calculation. A brief examination of the report will make it clear that the operator is able to convert the machine to his purpose only at the cost of much tedious preliminary labor. The prodigious speed and industry of the ENIAC, however, repay this initial expenditure many thousandfold in terms of desk-machine output. Nevertheless, it is encouraging to know that the newest electronic descendants of the ENIAC will be able to achieve at least equivalent results with only a tiny fraction of the effort and attention that the ENIAC currently demands of its keepers.

J. T. PENDERGRASS
A. L. LEINER

Navy Dept.
NBS

EDITORIAL NOTE: Although these v. contain many figures, tables, and drawings, reference is made to several drawings which are not included. Hence it is preferable to have the complete file of ENIAC drawings available while studying the text.

5. HARRY D. HUSKEY, *Report on the ENIAC* (developed under the supervision of the Ordnance Department, United States), University of Pennsylvania, Moore School of Electrical Engineering. *Technical Report* II, June 1, 1946, 163 p.; 12 figs. Mimeographed. 21.6 × 27.9 cm. Restricted (not for distribution).

This volume is the fifth of a 5-v. *Report on the ENIAC* (see no. 4, for v. 1–2). It is a step-by-step description of the operation of the circuits of the ENIAC written to accompany the block diagrams and circuit diagrams. These functional circuit descriptions prepare one for the *ENIAC Maintenance Manual* (v. 2), but they do not analyze the factors affecting decisions in engineering design. Voltages, tube types, and component values are given; but currents, powers, stray capacitances, rise times, wave forms, etc., are not. Absorbing the contents of this volume should be a very time-consuming task, worthwhile only for one who is to service the ENIAC.

ROBERT D. ELBOURN

NBS

6. IBM, (a) *IBM Selective Sequence Electronic Calculator.* 16 p. printed brochure, p. 5–6 = 6 p. paster; illustr. 22.9 × 30.5 cm.; (b) "IBM selective sequence electronic calculator is dedicated to science," *Business Machines*, v. 30, no. 11, 15 Mar. 1948, p. 1–12; illustr. 45.7 × 58.4 cm.

NEWS

American Physical Society.—On Friday morning, April 30, at the NBS, members attending the Washington meeting heard Mr. H. ZAGOR of the Reeves Instrument Corporation speak on the REAC. This machine is being developed under contract with the Office of Naval Research.

Mr. Zagor described a Reeves Electronic Analog Computer (REAC) employing direct-current computing elements and servos for the solution of initial-valued nonlinear total differential equations. Solutions of some differential equations were shown, as well as several interesting slides demonstrating the machine components.

The basic component of the REAC is a direct-current amplifier, of which each standard machine has a total of 20, consisting of 7 integrating amplifiers, 7 inverting amplifiers, and 6 summing amplifiers. The servos make it possible to carry out multiplications of variable quantities, to perform resolution of vectors, and to introduce arbitrary nonlinear functions. Thus, each standard REAC can solve differential equations up to the seventh order. However, several REAC's can be connected together to increase the capacity of the equipment where higher-order problems are involved. Solutions are recorded on a six-channel recorder providing graphical presentation of the effect of changing one or all of the parameters of the problem being solved. Insertion of problems is carried out by plugging of telephone cords to bays on the front of the cabinet so as to connect the input and output circuits of each amplifier in any desired manner. In order to permit continuously adjustable coefficients, 24 micropots with micrometer dials are provided. The over-all accuracy of the computer is limited by the resistors, capacitors, and drift in the power supplies. A REAC prototype has been in operation for over a year on Navy problems.

On Saturday afternoon, May 1, at the Department of Commerce Auditorium, the following papers on computing machines were presented:

"Computing machines" by Professor Howard Aiken, Harvard University.

"Status of large-scale computing devices in the United States" by Mr. R. L. Snyder, Jr., University of Pennsylvania.

"The IBM selective sequence electronic calculator" by Dr. W. J. Eckert, IBM.

International Scientific Radio Union (American Section) & Institute of Radio Engineers.—At this joint meeting in Washington, D. C., May 5, 1948, a talk on "Basic digital arithmetic circuits" was delivered by Mr. T. Kite Sharpless, Technitrol Engineering Co., Philadelphia, Pa. Mr. Sharpless presented some basic digital adding circuits in block and schematic diagram form. Both serial (time division) and parallel types were included. Adders using binary base and decimal base arithmetic were treated, with some discussion of the advantages and disadvantages of each. He also took up methods and circuits for taking care of subtraction or the algebraic addition of signed numbers. Two multiplying circuits, the repeated addition type for binary numbers, and a multiplication table type for decimal numbers, were presented and discussed. The problem of division was covered, but no specific division circuits were shown, since this operation can be achieved rather easily by combinations of the other basic operations. All the above-mentioned circuits use standard electronic tubes and circuit components. However, the paper concluded with a brief presentation of some special types of electronic tubes suitable for use as adders.

NBSMDL.—Each week members of the staff of the MDL at the NBS meet with other interested persons to discuss various existing and projected electronic computers.

On March 11, 1948, Mr. R. R. Seeber, Jr., of the IBM gave a detailed description of the recently completed IBM Selective Sequence Electronic Calculator. With the aid of slides Mr. Seeber explained each part of the machine and outlined the role it plays in the performance of a computation. The discussion was of interest not only to those who had not yet seen the new machine, but also to the many present whose brief acquaintance with it at the opening ceremonies left them with an incomplete knowledge of its intricacies.

On April 13 the group heard Dr. Samuel Lubkin speak on the proposed Reeves Arithmetic Computer (REEVAC). A description was given of the various features and expected performances of the machine, which has not yet been constructed. Since most of those present were acquainted with comparable features of the two electronic computers designed under the direction of the NBS by the Raytheon Manufacturing Company and the Eckert-

Mauchly Computer Corp., the talk was followed by a discussion of the relative merits of the three models.

Dr. HARRY D. HUSKEY of the MDL gave two talks at Ohio State University on May 24 at the invitation of the Graduate Mathematics Club and local chapter of Pi Mu Epsilon. In the afternoon talk, entitled "Mathematical aspects of electronic computing machines," newly planned high-speed digital computers were briefly described; the responsibilities of operating personnel were mentioned; and the impact of such machines on applied mathematics and engineering was considered. The evening talk, "Engineering aspects of electronic digital computers," described typical computer components and gave a brief indication of probable future trends of development.

On June 1 Dr. Huskey gave a one-hour invited address on "The present state of automatic digital computing machinery" at the semiannual meeting of the Association of Mechanical Engineers in Milwaukee. The history of the development of digital computing machines was covered as an introduction to the subject, and the various large-scale digital computing machines were discussed. The proposed new machines which are just now in the manufacturing stage were described in some detail, with special mention of their implications in the fields of applied mathematics and engineering and with particular emphasis on points of difference from the older machines. The generality of the new machines was stressed, and the interpretation as a general model was explained, along with some of the difficulties which must be overcome before this ideal of a general model can be realized.

As a sequel to Mr. Seeber's survey of the IBM Selective Sequence Electronic Computer, on June 15, at the NBS, Mr. KENNETH CLARK, also of the IBM, presented an informative treatment of the coding of an actual problem. In the course of this discussion many essential machine features not previously mentioned were brought out.

Personal.—STIG EKELOF, professor of Electro-Technique at Chalmers University, Göteborg, has recently returned from a trip to the United States, where he reviewed the digital computer program. Professor Ekelof is constructing a mechanical analog computer (differential analyzer).

OTHER AIDS TO COMPUTATION

Electric Root-finder

The analogy between the functions of a complex variable and the flow of electric currents in a conducting sheet has been used extensively to solve problems about currents. The inverse procedure does not seem to have been exploited as profitably as it might.

One useful function of a complex variable is the characteristic determinant whose roots are the natural frequencies of a dynamical system. Unfortunately, it is a tedious calculation to find these roots by any numerical process, when the degree of the determinant is high. An electric root-finder based on the current-flow in a conducting sheet was devised by F. LUCAS,[1] 1888–1890, but it suffers from two practical defects. First, the sheet must theoretically be infinite, which means in practice that it must be very large; and second, the current density at certain current sources must be large.

These defects can be avoided by a conformal transformation of the plane which carries the upper half-plane into a bounded region, but expands the areas surrounding the current sources.

To find the roots of various polynomials in z, we select a set of real values of z, say z_1, z_2 etc., according to any convenient scheme. The number of these values must be greater than the highest degree of the polynomials to be