CHAPTER XI

Calculator Organization and Control

The first serious thought involving the use of machinery to perform calculations seems to be credited to Blaise Pascal, who in about the year 1642 built some elementary machines to assist in the computation of taxes. In succeeding years various other mathematicians pondered the problems of reducing the drudgery of routine calculating and of increasing the speed and accuracy of the work. A few of them reached the stage of constructing models, but it was some 200 years after Pascal's first work before the production of calculating machinery was placed on a commercial basis. Punched-card accounting and calculating equipment had its beginnings in the 1890 census, although punched cards had been used years previously in the control of weaving complex patterns on looms. After about 1890 the development of adding machines, desk calculators, bookkeeping machines, and punched card accounting equipment proceeded at a rapid pace. Even so, the machines appearing on the market as late as the 1920's might be considered crude compared with the advanced models available today.

Some of the ideas presented in the various chapters of this book pertain in a vague fashion to adding machines, desk calculators, and machines of similar categories, but for the most part an entirely new class of machines has been in mind. Scientific and accounting problems of any complexity require long sequences of operations in order to reduce the input data to calculated results in the desired form. With the desk calculator category of equipment, substantially only one operation at a time can be performed. For each addition, subtraction, multiplication or other operation, the operator must depress some keys on the keyboard. The results of intermediate calculations must be recorded externally to the machine. Furthermore, the operator must make all decisions regarding the selection of alternate calculating routines when the arithmetic operations to be performed are dependent on the nature of the data or the values obtained at intermediate points in the calculations. With punched-card equipment many of these decisions can be mechanized through the use of sorters and collators, but even then the operator must at least handle the cards at each step in the process. The outstanding feature of the calculators which are the real subject of this book is their ability to perform long sequences of arithmetic operations and "logical decisions" in an automatic manner.

The construction of the first successful calculator capable of performing long sequences of operations was started in 1939 by the International Business Machines Corporation in cooperation with Harvard University and was completed in 1944. It is known as the Automatic Sequence Controlled Calculator or the Harvard Mark I. One previous attempt at the construction of such a machine had been made by the now-famous Charles Babbage of England at the incredibly early period of approximately 1830 to 1840. Parts of Babbage's machine were completed and made to work, but the project as a whole failed largely because it was too ambitious in view of the relatively primitive engineering techniques available at the time. In spite of the facts that rather detailed descriptions of Babbage's work had been recorded and that substantial parts of his incompleted machine had been preserved, the passage of the years caused his ideas to be largely forgotten. It is believed that the Automatic Sequence Controlled Calculator owes its existence to a study of Babbage's work and to a recognition that components and engineering techniques had been developed to a point where a large-scale calculator would be feasible.

Most of the components in the Automatic Sequence Controlled Calculator were developed by IBM for their regular line of business machines and were electro-mechanical in nature. While very useful large-scale calculators can be assembled with mechanical or electromechanical components, the more outstanding achievements in calculator technology have been made through the use of electronic components because of the extremely high speeds of operation which are attainable electronically. The first electronic calculator, known as the ENIAC, was built at the Moore School of Engineering in Philadelphia and was completed in 1946. The construction of the ENIAC must have required great courage for it was started at a time when the electronics industry was having considerable difficulty making the approximately 100 tubes in a radar set function simultaneously for long enough periods to be useful. From tube life data available at the time it was a simple problem in arithmetic to prove that tubes in a large-scale calculator would fail faster than the defective tubes could be found and replaced. Nevertheless, in spite of its 18,000 tubes, the ENIAC did operate successfully.

A basic requirement of sequenced calculators is the ability to store numbers and other information. Even in the more elementary sequenced machines the storage of intermediate results must be accomplished by some means or other; in the more sophisticated calculators, number storage is used for a variety of purposes. In the ENIAC, a bistable circuit known as the Eccles-Jordan trigger or flip-flop, which was first described in 1919, was employed as the storage means. The very large number of tubes used in the ENIAC was largely a consequence of the fact that only one binary digit can be stored in one flip-flop.

During the summer of 1946 the Moore School held a special course on the subject of digital calculators, and lecturers and students from a variety of organizations interested in the field were invited to participate. At this course the idea was presented that the information concerning the sequencing of operations was data that could be entered into the calculator and stored through the use of the same components employed for the entry and storage of the numbers undergoing the calculations. This concept contained a number of attractive features, the most outstanding of which was the possibility of using the program to alter itself. (Here, the "program" means the series of operations which the calculator is to perform.) Through the use of a "stored program" calculator, as this type of machine is called, a great variety of tricks could be performed which would aid in the solution of many kinds of problems. In particular, the using of portions of the program over and over (sub-routines) became easy to accomplish, whereas this result had been achievable only with considerable difficulty on previously existing calculator designs. Further, new means of electronic storage such as ultra-sonic delay lines, magnetic drums, and electrostatic storage tubes had been recently invented, and the practicability of using large amounts of electronic storage for programming was thereby enhanced immeasureably. It is hardly possible to give an accurate estimate of the amount of influence that the summer course had on progress in calculators; nevertheless, it was shortly afterwards when several projects were started to build calculators that were organized along the lines which had been described.

The first/ stored-program machine to be completed was built by ~~group to complete their machine was~~ a group headed by M. V. Wilkes at Cambridge University in England. Although their machine, known as the EDSAC, was modest in size and capabilities when compared with other large-scale electronic calculators being built at the time, Wilkes and his associates are well known for their contributions to programming techniques and procedures. Specifically they were the first to make extensive use of a library of sub-routines with a system for easily assembling them to make programs for the solution of new problems.

Another pioneering organization in the field of digital calculators was the Bell Telephone Laboratories. It was found that many of the components developed by the telephone and telegraph industry were suitable for the construction of a calculating machine. The arithmetic and control portions could be comprised largely of relays and other pieces of telephone switching equipment while teletype machines with their associated punched paper tape facilities served well for input and output equipment. As early as 1940 Bell Laboratories had completed a small calculator specifically intended to perform computations with complex numbers, which arise frequently in telephone engineering problems. Subsequently, a series of larger, more versatile, machines were built, with the last one, known as Model VI, being completed in 1950. All were of a type described as "relay calculators" because, of course, of the extensive use of relays in their design. An outstanding feature of Models II to VI of the series was the elaborate error-detecting facilities which were incorporated into them. Although components would occasionally fail, the errors created could almost always be detected automatically. Upon sensing an error the machine would stop so that appropriate corrections or repairs could be made by the operator. The machines gained an excellent reputation from the standpoint of reliability of the computed results. Also, the floating decimal point idea seems to have appeared first in the Bell Laboratories Model V, which was completed in 1946.

While relays and other electromechanical components are still used in certain portions, notably the input and output mechanisms, of most calculators, interest in the design of new "relay calculators" has diminished almost to the vanishing point, especially in the realm of the so-called large-scale calculators. The reason is a simple economic one; for a given amount of money more calculating can be accomplished with electronic calculators because of the much higher speeds which are attainable. However, even the electronic calculators are composed largely of components which were developed by another industry, the radio communications industry. In fact, it is literally true that with a little ingenuity a quantity of radio sets could be disassembled and then reassembled in the form of a calculator of quite respectable abilities. Although the designs of many electronic components have been altered to suit the needs and some new components such as magnetic drums and magnetic storage cores have appeared, the debt of the calculator industry to the communications industry is great indeed.

<u>Externally Programmed Calculators.</u> The phrase "externally programmed" is intended to mean that each operation which the calculator is to perform is under the control of some device external to the components and wiring of the calculator itself. It is in contrast to plugged program or stored program calculators, which will be discussed in later sections. Examples of external programming means are punched cards and punched paper tape. In the case of cards, one instruction might be punched in each card in a coded form of some sort. Then, with the deck of program cards in the hopper, the calculator would cause a new card to be sensed for each successive operation to be performed. To anyone at all familiar with more sophisticated methods of operation it will be clear that external programming of this nature has severe limitations. However, because the method has considerable historical interest and because the idea still has usefulness in some special applications, it will be described briefly.

Any calculator which is to perform long sequences of arithmetic operations must be capable of storing a multiplicity of numbers which are the initial data, the intermediate results, and the final results. Conceivably, this storage could be entirely in the form of punched cards or tape of the same type as used for program controls. In some of the earlier machines punched cards or tapes ~~was~~ were used for auxiliary storage (later machines use the higher speed magnetic tape), but they all employed at least some "built-in" storage. In the earlier machines the built-in storage was made up of decimal registers, each consisting of a set of decimal counters in parallel.

In order to perform addition, at least one and frequently several of the storage registers were equipped with carry handling facilities to form parallel accumulators, as described in the chapter on decimal addition and subtraction. To add two numbers, it was necessary to place at least one of the numbers in one of the accumulator-type registers and then cause the other number to be sent to this same accumulator. For shifting, multiplication, division, and other more complex operations the numbers were sent to one of a set of arithmetic units specially designed for the purpose, and the result of the operation was then returned to one of the storage registers.

The job of the program cards or tape was, therefore, largely reduced to a matter of controlling the transfer of numbers from one place to another. The problem is somewhat similar to the problem of telephone switching; it must be possible to connect any unit (phone) to any other unit. However, there is one important difference. In a telephone system it is desirable to allow several conversations simultaneously, but in the calculator the program cards call for only one operation at a time. Therefore, a common set of wires, or "bus," may be used to connect all storage registers and arithmetic units. Each must be equipped with an "in" switch and an "out" switch. An instruction on the program card, therefore, causes one "in" and one "out" switch to be closed. Then when the storage registers are sensed, such as by applying ten pulses to each counter and thereby rolling it around through zero to its original state, a set of pulses representing the digits are caused to appear on the bus from the unit which has its "out" switch closed and be transmitted to the unit which has its "in" switch closed.

Many, many more details must, of course, be considered in order to gain a thorough understanding of the system, but more important than the details is an appreciation of the limitations of programming by external means. In the preparation of programs it is frequently desirable to repeat some of the operations a number of times. A typical example of this requirement is the calculation of some quantity by an iterative process. The portion of the program can be repeated easily as many times as desired merely by punching enough program cards or tape, but this procedure is clearly unattractive. It is particularly unattractive when the number of times the iterative process is to be applied is not known in advance. The automatic returning of cards to the hopper or the back-spacing of tape by the required amount would accomplish the desired result, although apparently no one has ever considered this solution practical enough to build the required mechanism. Another solution is to put the part of the program which is to be repeated on a separate piece of program tape with ends joined to form a loop. This tape loop may be placed on a separate tape reader which is called into operation by the main program tape, and control is returned to the main tape when the calculated results indicate that a sufficient number of iterations of the loop have been completed. This "sub-routine" tape may, if desired, control other sub-routine tape loops as sub-sub-routines. Clearly, when many sub-routines are involved, the controls become complex and the number of tape reading units can become prohibitively great.

A more severe limitation of external programming is encountered when the sequence of operations is not continuous, but instead, it "branches" according to the results obtained at intermediate points in the calculations. Many examples are encountered where one set of arithmetic operations is desired if a given number is positive and a partly or entirely different set is desired when it is negative. Simple cases can be handled by a multiplicity of tape readers, but as more complex cases of branching are considered, this solution rapidly becomes impractical.

Further, it is not possible for the program to alter itself in the same sense that it is possible in a stored program machine. Many useful programming tricks, some of which are explained in the chapter on programming, are thereby not available on externally programmed calculators.

Plugged-Program Calculators. A second general method for controlling the sequencing of operations is through the use of a plug board (control panel). With a plug board the actual physical wiring of the calculator is changed for each new set of calculations to be performed by the machine. Clearly, it is important to devise a systematic arrangement for connecting together the various plug hubs by means of plug wires in order to reduce the required time and the likelihood for mistakes when preparing the plug board for a given sequence of operations. Even with the best designs, the amount of effort required to prepare a plug board is likely to be excessive when compared with inserting a deck of program cards especially in commercial applications where the same array of connections must be assembled and disassembled frequently. IBM solved this problem by making the entire plug board removable, and for each sequence of calculations which might be repeated frequently, a separate plug board is maintained with all plug wires in place. The outstanding limitation of a plugged program calculator is the limitation on the number of steps which can be accommodated in the sequence. The limitation is a practical one and not a theoretical one; for many steps (say much over 100) the amount of equipment becomes excessive and the mass of plug wires which are required becomes unwieldy.

In explaining how a plugged program calculator functions, no attempt will be made to describe some imaginary "generalized" calculator. Instead, one specific design of machine, IBM's Type 604, will be used as a pattern although liberal variations in the details and nomenclature will be made in an effort to illustrate the basic principles with a minimum of confusion from information that is not pertinent. However, it should be understood that the 604 arrangement is by no means the only one possible. Any group of imaginative engineers at the task of designing their own machine would probably incorporate many of their own ideas and arrive at a design subtantially different from this one. Incidentally, an interesting commentary on the 604 is that when the first model was built it was thought that the sales department would have to work hard to place as few as ten such machines in industry because no need for that much calculating could be visualized. In only five years after it was first announced in 1948, the 604's in service were numbered in the thousands.

A block diagram of the major units in the arithmetic and storage portions of the calculator is shown in Fig. 11-1. There are two eight-wire number busses (each indicated by a single line) in the system. One is used for entering numbers into the various registers, and this operation is called "read-in," which is abbreviated to RI. The other bus is for "read-out," which is abbreviated RO. Besides the registers intended specifically for storage, there are two special registers, one the multiplier-quotient (M-Q) register and the other is the accumulator. As the name implies, the M-Q register is for storing the multiplicand when multiplying and for storing the digits of the quotient as they are generated when dividing; however, this register may be used for ordinary storage if desired. All actual arithmetic is performed in the accumulator. The accumulator is similar in many respects to the other registers except that facilities for handling decimal carries are included. The decimal digit storage devices in all registers are electronic decimal counters

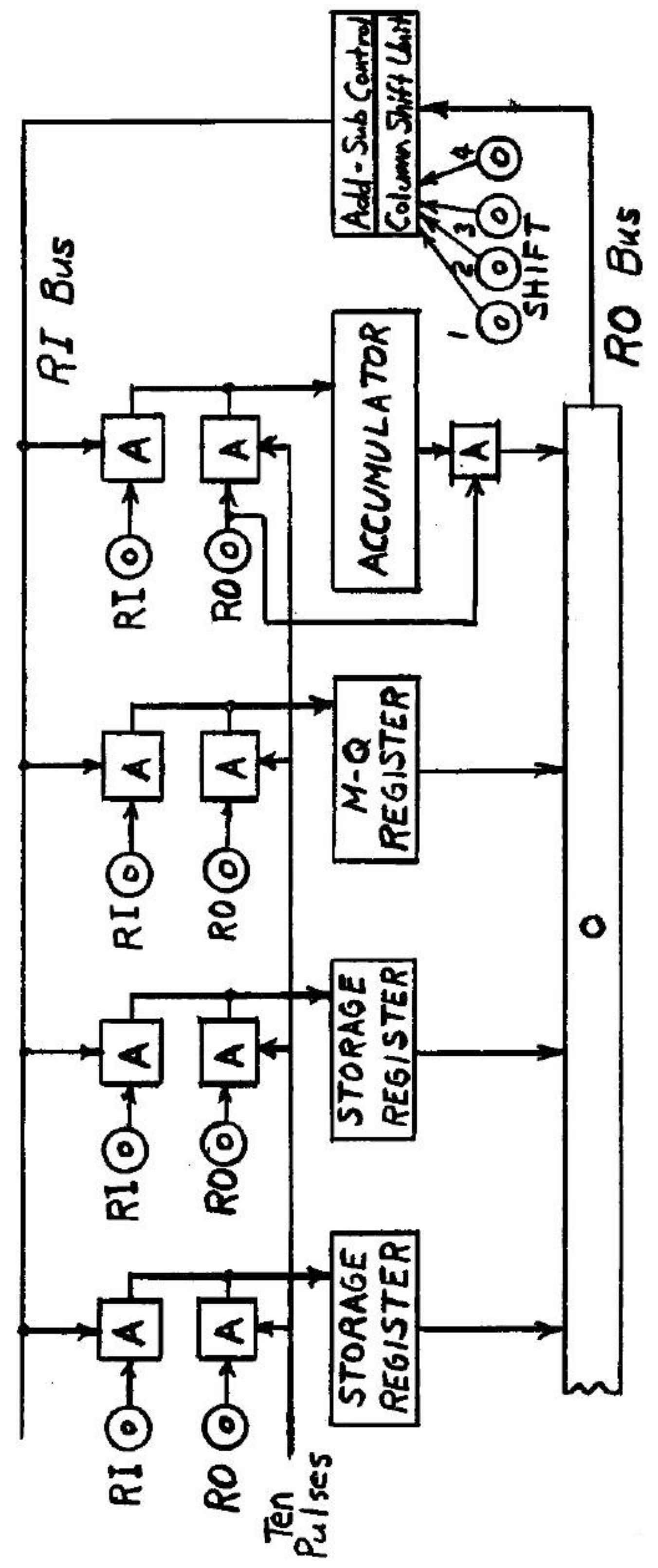


Fig. 11-1
Organization for Plugged
Program Calculator

made up of four binary flip-flops with connections to eliminate six of the sixteen possible stable states in accordance with one of the schemes described in the chapter on counting. Then, to read-in a digit, a number of pulses equal to the value of the digit are applied to the counter. To read-out a digit, ten pulses are applied to the counter and a pulse is caused to appear on the corresponding wire of the RO bus at the time the counter arrives at zero. After the tenth pulse the counter is, of course, back in its original state.

All arithmetic operations involve the transferring of numbers from one register to another, and all transfers occur through the "column shift unit" and the "add-subtract control." The column shift unit is a switching network capable of shifting the connections between the RO and RI busses under the control of external signals which are applied to it. For example, if a shift of two places is called for, the units RO signal will appear on the hundreds RI wire, the tens RO signal on the thousands RI wire, etc. The add-subtract control converts the timed pulses on the RO bus wires to series of pulses which the counters can count on the RI bus wires. Either the true or 9's complement representation can be obtained for addition and subtraction purposes in the manner described in the chapter on decimal addition and subtraction. To cause a number to transfer from one register to another it is necessary to apply a steady-state signal to one RI plug hub and one RO plug hub during the time of the transfer. In Fig. 11-1 each "and" and "or" switch actually represents eight such switches because of the eight wires in the bus. Also, control signals must be applied to the column shift unit to indicate the desired shift and to the add-subtract unit to indicate whether an additon or subtraction is to be performed. The set of "and" switches on the output side of the accumulator is for the purpose of preventing output pulses from reaching the RO bus when reading into the accumulator. Corresponding "and" switches are not needed on the outputs of the other registers because these other registers are always reset to zero before reading into them, with the result that the counters will never go past nine during read-in.

The basic source of pulses in the machine is a continuously running multivibrator which drives a 23-stage ring counter. The signals obtained from the various stages in the ring are used either directly or in conjunction with other flip-flops to generate control pulses and "gate" signals which may be needed throughout the calculator. This ring circuit, together with its associated equipment is sometimes referred to as the "clock." Time is measured with respect to the status of the ring. The instant when the last trigger in the ring goes off and the first one comes on is known as 1-time. The time of the second trigger's coming on is 2-time; the third, 3-time, and so on. The pulses generated from

1-time to 10-time are used for a variety of purposes including the resetting to zero of the appropriate storage registers, multiplication and division control, and others. The ten pulses used for the transfer of one storage location to another are generated from 11-time to 20-time, inclusively and the carry gate signal occurs from 21-time to 1-time of the next cycle. While a more detailed understanding of the "clock" would be desirable for some purposes, it will not be described further because an understanding of the sequencing of arithmetic operations does not depend upon it.

The signals controlling the sequencing of operations are obtained from another ring counter, called the "program ring." The program ring may have as many stages as desired with one stage, or step, being required for each operation. The program ring gets its stepping pulses from the first stage in the clock ring with the result that each time the clock ring goes through one complete cycle of operations the program ring is advanced one step. The output signals from the program ring are brought out to hubs on the plug board. Therefore, during each entire arithmetic operation (during each program step) a steady-state type of signal is available to control the various functions in the calculator. Actually, each output from the program ring is brought to three separate hubs, called the program exit hubs, and each connection is made through a separate electronic driving tube in order to prevent back circuits. The reason for the three output hubs is that it is frequently necessary to control three things during one program step.

To illustrate the functioning of a plugged program calculator, a simplified plug board wired to do the calculation, $x(x - 10y)$, is shown in Fig. 11-2. Assume that x is initially in storage register 1 and y is in storage register 2. During the first program step steady-state signals appear on the three program exit hubs corresponding to this step. A plugged connection is made from one of these hubs to one of the storage RO hubs corresponding to storage register 1. Note that there are four of these latter hubs shown in the figure. The line connecting them indicates that they are electrically wired together behind the plug board. This arrangement allows the convenient plugging of this RO function four times in the preparation of the plug board. The same remarks apply to other hubs which are similarly connected together. A second plug wire is used to connect program step 1 to an accumulator RI + (add) hub. The operation which occurs as the clock executes one cycle during the first program step is the transfer of x to the accumulator (refer also to Fig. 11-1). It is assumed that the column shift unit passes numbers without a shift unless a plug wire is used to control it otherwise.

Plug wires connecting the program step 2 hubs to storage unit 2 RO, accumulator RI - (subtract) and column shift 1 cause 10y to be subtracted from x. To perform multiplication, one of the two factors

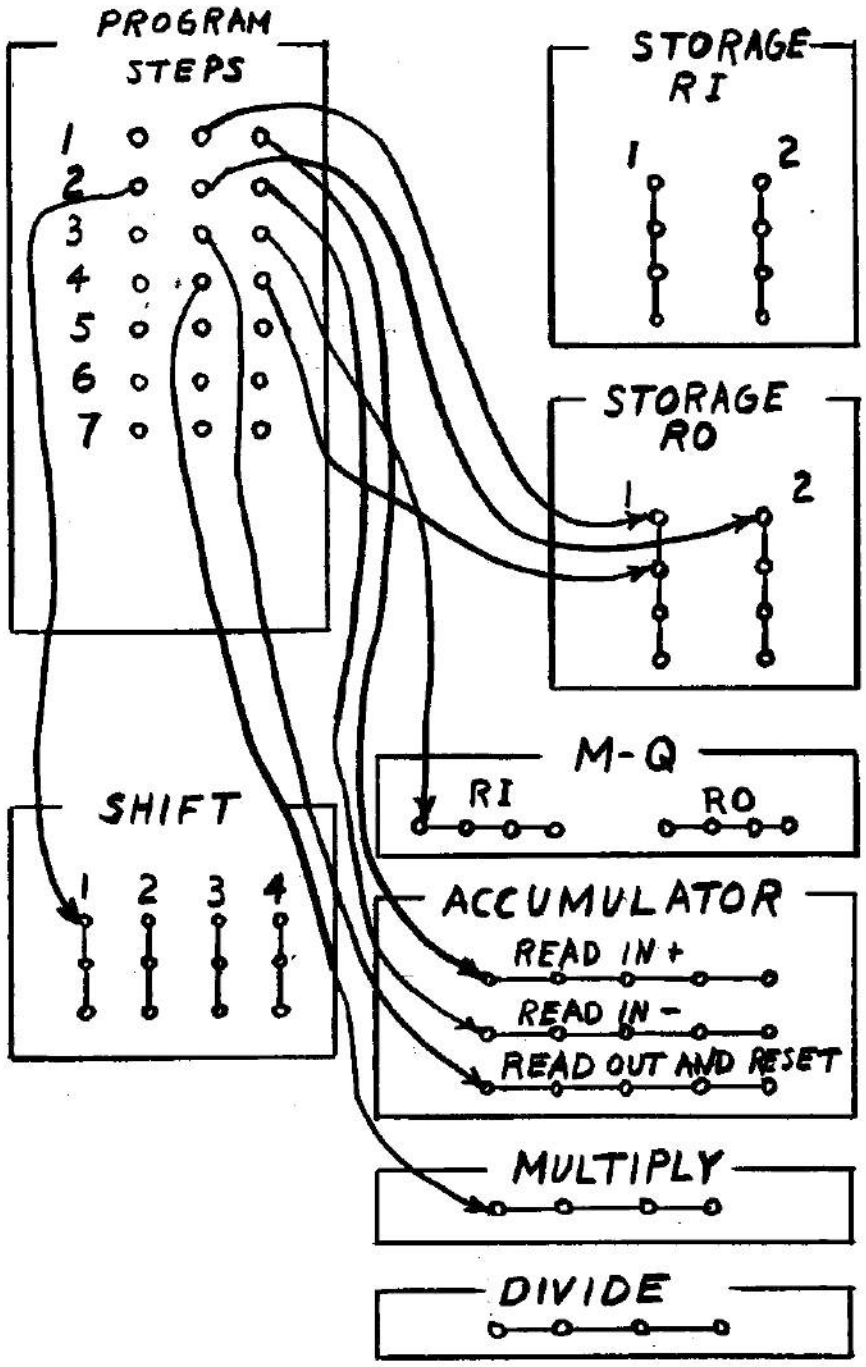


Fig. 11-2. Plug Board Wiring for Calculating x(x - 10y)

must be placed in the M-Q unit. This is accomplished in program step 3 by connecting plug wires to an M-Q RI hub and an accumulator RO and Reset hub as shown in Fig. 11-2. During the fourth program step the multiplication of x by x - 10y performed under the control of plug wires to another RO hub for storage unit 1 and to one of the multiply hubs. Internally, multiplication proceeds by an over-and-over addition process with the number in the M-Q unit as the multiplier and the number in the indicated storage unit as the multiplicand. During a multiplication the clock must execute as many complete cycles as there are additions to be performed so the pulse to the program ring from the clock is blocked until the multiplication is complete. During multiplication and division operations the column shift unit and the sensing of the multiplier digits one at a time are under the control of a ring counter which is advanced one step at the completion of the handling of each multiplier or quotient digit, as the case may be.

As might be expected, any practical plug board arrangement would contain many more features than the basic ones which have been described. Of these other features, the most important ones are the means for causing the calculator to proceed through different sequences of operations under the control of intermediate results and to repeat certain sequences as required in iterative processes. One system for accomplishing these results involves the suppression of certain program steps. Along with the three program exit hubs on the plug board corresponding to each program step there is a fourth hub called the program suppress hub. When a signal is applied to this hub through a plug wire from any other source, the program step will not be executed because signals will be prevented from appearing on the program exit hubs. In most cases the calculations can be arranged so that the determination of which sequence to be followed is dependent on the sign of a number in the accumulator. Therefore, the accumulator sign flip-flop can be used to supply accumulator-plus and accumulator-minus signals which may be used for suppressing unwanted program steps. Other sources of program suppress signals may, of course, be used if desired. To repeat a portion of the program using this system it is necessary to repeat the entire program while suppressing all program steps except the portion to be repeated. To facilitate the suppression of so many steps and for other purposes a "group suppression" feature has been devised. A binary storage device is used to control the suppression of any desired group of program steps through the use of appropriate hubs and wires on the plug board. The steps in the group are suppressed or not according to the status of the bistable device, which may be an electronic flip-flop or a relay, and which may be turned "on" or "off" by signals from any of a variety of sources in the calculator. By supplying a multiplicity of group suppression devices, considerable flexibility may be achieved in the altering and repeating of sequenced operations. This system is used on IBM's type 607 calculator and the "Card Programmed Calculator."

More flexible and higher speed methods of altering and repeating programs may be worked out, usually at the expense of additional equipment. One such method involves the elimination of the "ring" feature of the program counter. Instead of proceeding from step to step in a uniform sequence, each program step is provided with an "in" hub to which a signal must be applied to initiate the action called for by the plug wires of that step. Also, with each program step there is an "out" hub on which a signal appears when that particular step has completed its operation. A plug wire must then be used to connect the "out" hub of each step to the "in" hub of the step which is to follow. When the selection of the next step is to depend upon the status of some condition in the calculator (such as the plus-or-minus condition of the number in the accumulator) a group of pluggable "and" and "or" switches are needed. For example, assume that the program steps are executed in a uniform sequence through step number 15, but after step 15 the next step should be 16 if the number in the accumulator is plus or step 7 if it is minus. To accomplish this result, the outputs of the accumulator sign flip-flop are combined by means of plug wires in "and" switches with the "out" signal from program step 15. The outputs from the "and" switches are plugged to the "in" hubs of steps 7 and 16 respectively with the result that upon completion of step 15 a signal will be routed to step 7 or 16 in accordance with the status of the sign flip-flop. The "or" function is needed at the input of step 7 because this step may be initiated from step 6 or step 15. Since there are no restrictions on which step may be the "next" one, the altering and repeating of sequences are accomplished in the same way.

Storage Devices. Before proceeding to stored-program calculators a few points concerning the devices used in them for number storage should be understood. While the counters used for storage in the earlier externally-programmed and plugged-program calculators could be used in a stored program fashion, they are not nearly as practical as the storage devices which have been developed specially for the purpose. The organizations which have been worked out for stored-program machines depend in large measure upon the nature of the storage medium which is chosen.

Of the various storage devices which have been considered for calculator applications, five have emerged as being acceptable from engineering and commercial standpoints. They are: magnetic cores, electrostatic storage tubes, acoustic delay lines, magnetic drums and magnetic tapes. The differences in these storage devices are not only in the physical mechanism of storage, but also in the means used for gaining access to the individual storage bits. In fact, from the standpoint of calculator organization, the access properties have a far greater effect on design than the storage mechanism itself. Naturally, cost has an important bearing also.

Magnetic Cores. Of the five, magnetic cores are the most recent to become successful, and it is with them that the shortest access times are possible. The storage principle of magnetic cores is simple enough. A toroidal-shaped piece of ferro-magnetic material with a nearly square hysteresis loop is caused to be magnetized to saturation with the lines of flux passing in one direction or the other around the toroid. The remanent flux represents the storage of a binary 1 if it is in one direction and 0 if it is in the other direction. The problem of gaining access in a practical manner to a specific core in an array is more difficult. In general, the means which are employed make use of the fact that, with a square hysteresis loop, it is possible to apply a magnetic field of strength H_O which will have no effect where a field of $2H_O$ is sufficient to cause the flux to change completely from one state to the other. Then, in a rectangular array of cores, a current in a wire which passes through all cores in one column combined with a current in a wire which passes through all cores in one row can be used to selectively magnetize the core at the intersection of the row and column. To sense the status of the core it is necessary to magnetize the core by the same procedure but in the opposite direction and to use a third wire through the core, which acts like a secondary winding of a transformer. A pulse or no pulse appears in the secondary depending upon whether the flux was changed or not when sensing. Therefore, the problem of selecting a given core reduces to a problem of selecting two wires (one for row and one for column) to which pulses of current should be applied. When the number of cores is so great that one array is not practical, it is necessary to select one of a third group of wires to select the desired array. When each individual core is specified by a number which is stored in a set of flip-flops, it is possible to select the desired wires through the use of matrices as described in the chapter on switching networks.

Electrostatic Storage. With electrostatic storage the 1's and 0's are stored in small charged areas on the face of a cathode ray tube; if a given area is left charged positively or negatively a 1 or a 0, respectively, may be stored. Here, positive and negative are relative to each other and not to ground. There are several different ways by which the electron beam (cathode rays) can be used to create the desired charge pattern. The most successful, at least in terms of amount of use it has received, is known as the Williams tube system. With this system ordinary cathode-ray tubes developed for oscilloscopes and television sets can be used although better results are obtained when certain design parameters are changed.

The feature in common with most electrostatic storage systems is the use of deflection plates to deflect the electron beam to the desired area on the tube face for access purposes. To gain access to a given spot, a voltage which is one of a multiplicity of discrete values is applied to the horizontal deflection plates and another voltage which is one of a different set of discrete values is applied to the vertical deflection plates. If a number representing the location of the desired spot is stored in a register composed of a set of flip-flops, it is necessary to convert these binary voltages to a sort of "semi-analog" voltages to be applied to the deflection plates. This effect is usually accomplished through current summing circuits comprised of precision resistors with values appropriately chosen. The summed currents are passed through other resistors to develop the stepped voltages which are then amplified and applied to the deflection plates.

The time required for access with electrostatic storage is slightly greater than for magnetic cores. The reason is largely in the fact that it takes some time to develop the deflection voltages to the accuracy necessary for the beam to hit the desired spot on the tube face within the tolerance required for reliable operation. With either cores or electrostatic storage the sensing of information destroys it and it must be re-recorded if its continued storage is desired. In the case of Williams tubes, all stored information must be re-recorded periodically whether it is used or not because the charge pattern would otherwise gradually disappear because of ohmic leakage. As a result, with this type of storage in some applications, a certain amount of time is required for regenerating storage and is not available for calculating.

Acoustic Delay Lines. With acoustic delay lines the storage mechanism is the presence or absence of mechanical vibrations travelling along the length of some material, usually a column of mercury. The term, "acoustic" is somewhat of a misnomer in that the frequency of the vibrations is well above the range of hearing; in fact, the pulse repetition rate may be as high as 2 megacycles or more and the major frequency components of the individual pulses are even higher. To convert the electrical signals from the calculator to mechanical vibrations in the mercury, a quartz crystal is mounted at one end of the column with the vibrations being induced by virtue of the piezoelectric properties of the quartz. Another quartz crystal is used at the far end of the line to sense the pulses as they arrive and convert them back to electrical signals. Of course, the amplitude and shape of the pulses are deteriorated when passed along the delay line. Therefore, for continued storage of information they are not returned directly to be recirculated; instead, they are used to gate fresh pulses into the line. This procedure is possible because it is the time of the pulse in the line which is of consequence in identifying it with a particular binary digit. The practicality of acoustic delay lines lies in the fact that it is possible to store several hundred binary digits of information in one line.

The means used for gaining access to a given bit in a delay line is quite different from the means used for cores or electrostatic storage. Once a pulse is sent down the delay line it cannot be sensed until it reaches the other end. Even a few intermediate sensing points along the line are usually not considered practical. However, once the pulse does appear at the output of the delay line, it appears on one wire and no further selection is required. To specify the location of a given pulse it is necessary to divide time into cycles which repeat over and over, where the time for one cycle is equal to the time required for a pulse to travel the length of the line. Each cycle is sub-divided into individual pulse periods equal in number to the number of binary digits stored in the line. Each subdivision of time may then be assigned one of a series of numbers from zero to the storage capacity of the line. To gain access to any given binary digit the number representing its time of arrival at the output of the delay line (which is also the time of entry) is placed in a counter. At the start of one of the cycles, pulses are supplied to the counter to cause it to count towards zero in synchronism with the individual pulse period of the delay line. When the counter arrives at zero an "and" gate is opened to allow the pulse, if any, which is emerging from the delay line at this time to pass into other calculator circuits. A 1 or a 0 is represented by the presence or absence, respectively, of a pulse at this time. When more than one delay line is used, some of the digits of the number which represent the desired storage location are applied to a matrix to select the appropriate line in the same way that electrostatic tubes or arrays of cores would be selected.

Magnetic Drums. In the case of magnetic drums the binary digits are stored as small magnetized areas on the surface of a revolving cylinder. Access to a spot on any one "track" is accomplished by substantially the same procedure as is used for acoustic delay lines. The delay in the drum is obtained from the time required to physically transport the spot in its circular path. The drum may be used in two different ways. If separate magnetic heads are used for recording and sensing the binary digits, the digits may be recirculated in ~~substantially~~ the same manner as in the case of delay lines. A track operating in this way is sometimes called a "revolver." The other mode of operation involves the use of the same magnetic head for recording and sensing. With this scheme, the digits are not continually erased and rewritten, but instead they may be left on the drum surface indefinitely. However, it then becomes necessary to synchronize the drum position with the circuits used for access. A "timing track," which consists of a special track with a uniform series of magnetized spots permanently recorded, may be used for this purpose. The pulses obtained from the timing track can be used to control the drum speed or the drum may be allowed to run at its own speed and the pulses used to control the access circuitry.

Practical pulse repetition rates for drums are in the order of 100 kilocycles, which is less than for acoustic delay lines by a factor of, roughly, 20. Also, the access time for comparable storage capacities is proportionately greater. Both of these properties are strong disadvantages, but magnetic drums have nevertheless found wide application because of their relatively low cost.

Magnetic Tape. The storage principle used for magnetic tape is substantially the same as for magnetic drums although because of tape handling problems the maximum practical pulse repetition rate is much less. However, the access procedure is radically different. The amount of information that can be stored on tape is almost unlimited because pieces of tape can be spliced together to form very long lengths. The problem of physically handling the tape becomes acute, of course, when unduly long lengths are employed, but even with short lengths of tape the problem of locating a desired spot is of importance. As a general rule, no attempt is made to assign numbers to the storage locations on tape (although it can be done) as is done with all of the other storage devices which have been described. Instead, information is usually recorded along the length of the tape in sequential fashion as it passes under the magnetic head without regard to the exact position of any particular bit of recorded information. When sensing the information, the tape is passed by the magnetic head, and the circuits which respond to the pulses from the head must be prepared to accept pulses at any time because the time of their arrival is not known.

When, as is usually the case, the recording is not being done continuously, the tape drive mechanism must be stopped when not recording. Otherwise, large sections of unused tape would pass by the heads, and this procedure would not only waste tape but would make the access problem even more severe. Since the tape drive mechanism cannot stop or start in a time which is short compared with the pulse repetition rate of the recorded information, a certain amount of space on the tape must be allowed between each "block" of recorded information. Further, there must be at least some safety factor in the amount of space. Because of the general uncertainty of the location of any information stored on tape it is usually not practical to alter single bits of information. It is more common to erase an entire block and then record it again in its altered form. Care must be taken that sufficient space for tape starting and stopping is allowed at each end of the block.

Because of these characteristics of tape, calculators are usually designed so that access to the desired places on tape is gained by means of programming instead of by built-in equipment. For example, the program may be written so that the blocks of information recorded on the tape can be counted. For another example, if tables of functions are stored on the tape, the arguments and values can be stored in alternate blocks. Then through programming, the calculator can examine by a comparison operation all of the arguments until it comes to the right one, at which time it senses the next value of the function.

Consideration of Storage Devices Relative to Stored-Program Calculators. Of the various storage devices which have been described, magnetic cores and electrostatic storage tubes are the most attractive for stored-program calculators in that access to any storage location may be gained with equal ease at any time. The choice between the two in any given application would be based on several engineering and economic considerations, but from the standpoint of calculator organization either may be chosen.

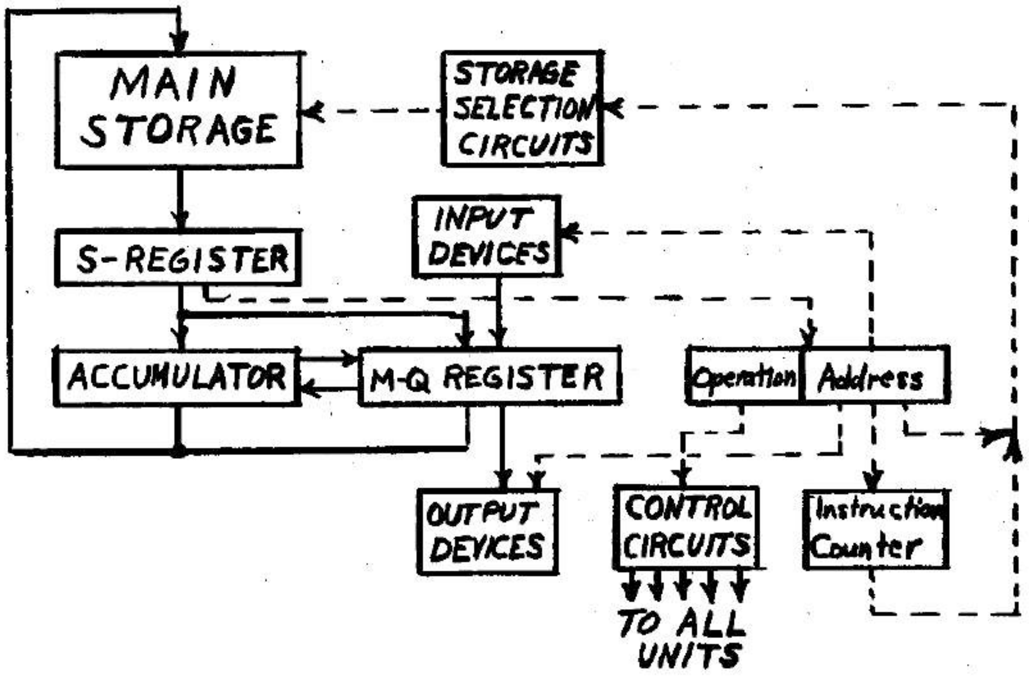
Acoustic delay lines and magnetic drums are less desirable because in the general case it is necessary to require the calculator to wait to place any information in storage or to recall it from storage. Nevertheless, these two forms of storage are used in many machines because they have other properties which are more advantageous. When using acoustic delay lines in a calculator employing the serial mode of operation, a reasonably fast and reasonably inexpensive machine can be built because of the very high pulse repetition rates which are possible. The waiting time can be minimized by writing the program so that at each step the desired storage location is the one approaching the end of the delay line at the time. The outstanding advantage of magnetic drums is the relatively low cost of a given amount of storage, and again, access time can be reduced by writing the program in an appropriate manner. However, the problem of "minimum access programming," as it is usually called, is by no means a simple one. In the case of magnetic tape, the access time is far too great to make tape practical as the principal storage medium in general purpose electronic calculators.

Also, either magnetic drums or tape, or both, are useful storage devices in calculators using some other medium for its main, fast-access, storage. Larger storage capacities can be made available at a more attractive cost. The access problem can be side-stepped in large measure by transferring large blocks of information between the fast-access storage and consecutive locations on the tapes or drums rather than by making reference to substantially random locations on them. With this arrangement, all arithmetic operations are performed with the aid of only the main storage. When the capacity of the main storage will be exceeded, a few storage locations are saved for instructions which will record appropriate blocks of intermediate results on the tapes or drums and then call in blocks of more program steps or data, as required.

Tape reels on a calculator can be changed readily by an operator. For this reason magnetic tape is frequently viewed as being an input or an output device instead of a storage device. Since the true source of information is seldom, if ever, found on magnetic tape and since invisible spots on a magnetic material are of little value as a final output, the role of magnetic tape as an input-output device could be questioned (compared with meter readings, keyboards, printers, and graph plotters, for example). Regardless of the semantics of the case, magnetic tape is frequently treated as an input and output device from the standpoint of calculator organization. Also, magnetic drums are frequently handled organizationally as input-output mechanisms when used in conjunction with magnetic cores or electrostatic storage tubes even though drums are usually not physically removable from the machine.

Organization of a Stored-Program Calculator. As with other forms of calculators, there are a great many variations in the ways in which a stored-program calculator can be organized. The organization which will be described here is patterned after IBM's Type 701 (a parallel, binary machine) although some modifications in details and nomenclature have been made to facilitate explanation of basic principles without confusion from information which is not pertinent. The 701 was chosen because the arrangement of this machine is about as straightforward as any and because all of the fundamental features of stored-program calculators are easily illustrated. Other important features found in the 701 and other machines of this category will be described in later sections.

As a general rule, the storage devices which are useful for the main storage in stored program calculators are not satisfactory for the temporary storage of numbers during the time that they are undergoing arithmetic operations or controlling the sequence of operations. For this reason, a set of miscellaneous storage registers, each specially adapted to a specific purpose, are employed in addition to the main storage. They may be comprised of any of a variety of storage elements. Conventional bistable flip-flops are frequently used, and for purposes of visualization, the use of flip-flops may be assumed. Three registers are used for storing numbers which enter into the calculations. One, known as the "storage register," or S-register (see Fig. 11-3), serves the primary function of storing the multiplicand during multiplication and the divisor during division so that it is not necessary to make repeated references to the main storage during these operations. With the electrostatic storage system, as used in the 701, it is thereby possible in some cases to do nearly all of the regenerating of storage during these two relatively lengthy operations. Numbers are transferred from the main storage to the S-register over a set of parallel wires, one for each binary digit.



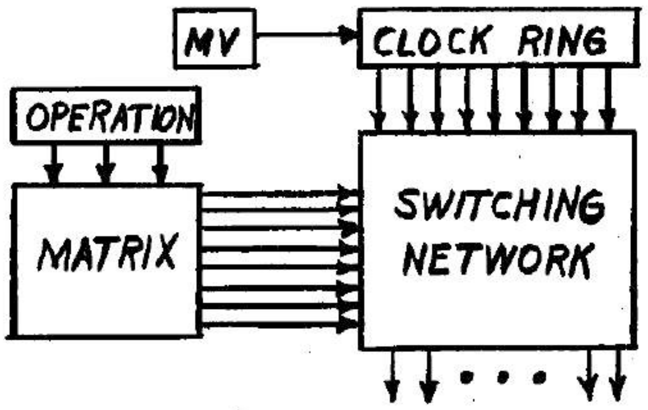


Fig. 11-3. One arrangement for a stored-program calculator.

The accumulator register can be of any one of the forms described in the chapter on binary addition and subtraction, or it can be an adder used in conjunction with a conventional register. The multiplier-quotient, or M-Q register, as the name implies, is for the purpose of storing the multiplier during multiplication and the quotient during division.

For addition and subtraction, numbers are taken from appropriate locations in the main storage and sent to the accumulator. They are sent through the S-register because this path is needed anyway for other purposes. For multiplication, it is necessary as a first step to cause the multiplier to be transmitted from the main storage to the M-Q register. The accumulator and M-Q register are both capable of shifting the numbers in them to right or left. Then at the start of the actual multiplication process the multiplicand is obtained from the main storage and placed in the S-register to be added in over-and-over fashion in the accumulator, which shifts to the right one step after each addition. The product as it is built up is shifted into M-Q register as has been described in an earlier chapter. At the conclusion of the multiplication process the double-length product is stored with its high-order digits in the accumulator and its low-order digits in the M-Q register. If it is desired to retain the entire product, two storage locations in the main storage are required and two program steps are used to transfer it from the two registers. Division is substantially the reverse of multiplication. The dividend is placed in the accumulator (or the accumulator and the M-Q register if a double-length dividend is required) and the divisor is stored in the S-register. As the division process proceeds, the digits in the accumulator are shifted to the left with the result that the final remainder appears in the accumulator and the quotient in the M-Q register. All paths used for the data and results are indicated in Fig. 11-3 by solid lines.

The problem is now to control the transmission of data between the main storage and the three registers in the arithmetic portion of the machine. This function is to be accomplished through the use of numbers representing program steps, and these numbers are to be stored in the main storage along with the numbers representing data. The instruction counter, the operation-address register, and the control circuits are the major units which are used for accomplishing this purpose.

The instruction counter ~~is a counter which~~ has two functions. First, it keeps track of the program step which the calculator is executing at any given time. Normally, a pulse is sent to the instruction counter at the conclusion of each arithmetic operation to step it up by one count, but for altering or repeating a program the contents of the address part of the operation-address register may be transferred to the instruction counter to replace the number there. The second purpose of the instruction counter is to control the storage selection circuits when a number representing a program step is being sensed in the main storage. The nature of this control would depend upon the choice of storage device used for the main storage, as described in previous sections. The paths to and from the instruction counter are indicated in Fig. 11-3 by dotted lines, as are all of the paths which transmit information pertaining to the control or programming of the calculator.

Before describing the function of the operation-address register, the meaning of the term, "address," must be explained. In its narrowest sense, an "address" is a number which represents a storage location in the main storage. Usually, each location is assigned one of a series of consecutive numbers from zero to the storage capacity of machine. Then, when an address is sent to the storage selection circuits, access is gained to the storage location represented by that address. By this definition the number in the instruction counter is more than an abstract number used for counting program steps; it is an address also, because it prescribes the storage location from which a number representing the program step is to be taken. As will be explained further later, an address can be used to designate other things. For example, it is used to specify the desired input or output mechanism when sending information to or from the calculator. Also, the address specifies the number of shifting steps that are to take place in a shift operation.

The operation-address register is used for storing the "instruction," which has been previously referred to as the number which represents the program step. An instruction consists of two parts, known as the operation part and the address part. The operation part specifies the operation to be performed, which may be an arithmetic operation such as add or multiply, or which may be any one of a long list of other operations such as the transfer of a number from one place to another or the causing of a magnetic tape unit to rewind. This part of the instruction causes the calculator to perform the indicated operation by means of control circuits, which are described more fully in a later section. As the name implies, the address part of the instruction specifies the addresses of the operands when the main storage is involved or the input-output device, the number of shifts, and so on, as the case may be in other types of operations. In the 701 as well as in several other calculators the address part of the instruction contains only one address, and machines of this type are called "single-address" calculators. Some calculators contain as many as four addresses per instruction. The nature of the individual operations are somewhat different in multi-address calculators from the ones found in single-address calculators, but the general organization of the two types of machines can be quite similar. Incidentally, the address part of the register is a counter as well as a register and is used for keeping track of the shifts in a shift instruction or during a multiplication or a division.

During operation, the calculator alternately comes under the control of the instruction counter and the operation address register. To visualize the sequencing of the calculator functions, assume that the program is initially stored in the main storage with at least the first few instructions in the lowest numbered addresses. The various items of data may be at any desired addresses. If the instruction counter is initially at zero, the control circuits first cause the instruction at address zero to be taken from the main storage and sent to the operation-address register. (The fact that the path is through the S-register is incidental.) Normally, the instruction is rewritten at address zero so that it may be used again. The calculator then performs the operation indicated by the digits in the operation part of the address register. Upon completion of the first operation, control is returned to the instruction counter, which has in the meantime been stepped from zero to one. The instruction at address one is now caused to be sent from the main storage to the operation-address register, after which the second instruction is executed under control of this register, and so on. In other words, each program step consists of two parts, which are (a) the securing of the instruction and (b) the execution of the instruction. Reference to the main storage may be made during each part; in (a) the storage selection circuits are under control of the instruction counter and in (b) they are under control of the address part of the operation-address register.

For a further understanding of a stored-program calculator it becomes necessary to consider the list of instructions which the calculator is able to perform. In any practical calculator this list would involve a mass of detail, the description of which would be well beyond present purposes. The chapter on programming describes a simplified list of instructions with sample programs illustrating its use. Here, the discussion will be limited to two outstanding features of instructions which can be built into calculators of the stored-program variety.

The basic problem of altering a program or repeating portions of it is solved in the stored-program calculator by a "jump" (sometimes given other terms such as "branch" or "transfer") instruction. The jump instruction causes the address part of the instruction, which is in the operation-address register, to be sent to the program counter to replace the number there. The result is that the uniform sequence of addresses from which instructions are obtained is terminated, and a jump is made to some other address. Then, because the program counter receives one pulse to be counted for each program step, the selection of instructions from sequentially-numbered addresses is resumed at the new address and is continued until another jump instruction is encountered. The great usefulness of jump instructions arises from the possibility of using some criteria in the calculator to control whether or not the jump is actually made. It is common practice to include at least two jump instructions, with one causing the jump to be made unconditionally and another causing the jump to be made or not under control of the sign of the number which is in the accumulator at the time. Other jump instructions may be included which are conditional upon different factors. It is not necessary that the factor be within the calculator; for example, in some applications the jump might well be made dependent upon the time of day.

A second important feature of the instructions in a stored-program calculator is that they are indistinguishable from the data. The programmer must keep track of which is which. Occasionally a certain amount of confusion results, but it is useful to be able to perform arithmetic operations on instructions. The addition or subtraction of a constant from the address part of an instruction is an operation which is performed frequently when using sub-programs. Another example of the usefulness of the feature is in the storage of tables when the arguments form a uniform sequence. To find the address of the value corresponding to any given argument, it is sufficient to perform a simple computation on the argument and then use the result as the address part of an appropriate instruction. A time-consuming searching process is thereby avoided. Arithmetic operations on the operation part of an instruction can be of considerable value too, but they usually fall in the category of tricks, any one of which can be used only on the machine for which it was devised. Also, as a result of the interchangeability of instructions and data, the program can be arranged so that extensions of the program can be entered into the calculator through the input devices under control of the program itself in the same manner that new data is entered.

The input and output devices are shown connected through the M-Q register in Fig 11-3. That the M-Q register is used in this way is incidental; it just happens to be convenient. However, some temporary storage of some sort is usually needed between the input and output devices and the main storage because the various units are not synchronized with one another. When an instruction calls for a number to be sent from main storage to an output device, for example, the output device may not at that particular instant be prepared to accept it. Then when the output device is ready to accept the number, the timing in the arithmetic part of the calculator may not be at the right point for transmission. Another factor, which is probably even more compelling, is the fact that the form of the number may be different in the two places. In particular, in the 701 numbers are transmitted 36 bits at a time to and from main storage, but only 6 bits at a time to and from magnetic tape. Both the timing and the change of form problems can be solved through the use of "buffer" storage, as it is sometimes called.

Control Circuits. The objective to be accomplished by the control circuits in a stored-program calculator is the causing of all the individual units of the calculator to perform in such a manner that the instructions in the main storage are sensed in the proper sequence and executed. In general, the units are controlled by sending pulses to them over a set of wires which may be called "command lines." Each command line is for a specific purpose such as transferring a number from one register to another, shifting the number in a register, resetting a flip-flop, or any one of a multitude of other functions. Usually it is necessary to send pulses, appropriately sequenced in time, over several different command lines to execute any one instruction. The circuit arrangement to be used in any given case for distributing the control pulses on the command lines depends in large measure on the organization of the calculator as a whole, and in existing machines great variations will be found when comparing one calculator with the next. Two general methods of assembling control circuits which can be used in a wide range of machine organizations will be outlined.

One possible organization for the block labelled "control circuits" in Fig. 11-3 is given in Fig. 11-4. The basic source of pulses is a continuously running multivibrator (MV) which drives a ring counter indicated in the figure as the "clock ring." Timed pulses are obtained from each stage of the clock ring, and one complete set of timed pulses is called a "cycle." At least two cycles are required to perform any arithmetic or other operation; one cycle, called the "instruction cycle," is used to transfer the instruction from the main storage to the operation-address register, and then at least one "execution cycle" is needed for the actual operation. The timed pulses from the ring are, therefore, sent through a switching network which distributes them to the various command lines as called for by the operation and the status of certain signals applied to the switching network as "miscellaneous inputs."

Among the more important miscellaneous input signals are signals from a flip-flop in the calculator which indicates whether an instruction cycle or an execution cycle is being performed at the time. An instruction is obtained from storage during an instruction cycle and executed during an execution cycle. During an instruction cycle the signals from the clock are sent to appropriate units for the transfer of the instruction from the main storage to the operation-address register. Recall that during an instruction cycle the storage selection circuits are under control of the instruction counter. One of the last pulses in the cycle is used to alter the instruction-execution flip-flop to cause the next cycle to be an execution cycle. In general, all instruction cycles are exactly alike, but there is a different type of execution cycle for each arithmetic or other type of operation that the calculator can perform. Further, some operations, notably multiplication and division, require many execution cycles, not all of which are alike. Also the execution cycles for conditional jump operations depend upon the conditions as supplied to the switching network through some of the miscellaneous input lines.

The major factor controlling the calculator during an execution cycle is the number which was placed in the operation part of the operation-address register by the immediately preceding instruction cycle. The digits of this number are applied to a matrix which has one output line for each operation the calculator is capable of performing. The output signal from this matrix together with certain signals from miscellaneous inputs then controls the distribution of the clock pulses to the command lines to cause the execution of the indicated operation. For all operations which refer to the main storage, the particular location selected will be the one indicated by the number in the address part of the operation-address register. For operations which require more than one execution cycle, a counter must be provided, and output signals from the counter are among the miscellaneous input signals to the switching network. A command line is used to pulse the counter at the completion of each cycle; when the counter indicates that the required number of cycles have been completed, a pulse on a different command line causes the instruction-execution flip-flop to call for the next instruction cycle. During the execution of the instruction one of the command pulses was used to advance the instruction counter by one step so that the instruction located in the next sequentially-numbered address will be sensed unless the operation was a jump, in which case the next address will be the one which was transferred into the instruction counter from the address part of the operation-address register.

While it is a relatively straightforward matter to assemble control circuits along this pattern which will make a calculator capable of performing any list of operations that might be desired, minimizing the number components is usually a complex puzzle. Boolean algebra is a useful tool, but it is not wholly adequate, not only because of the multiplicity of signals involved, but also because the timing of the signals is a parameter which the designer can vary and which falls outside the scope of Boolean notation. Much is left to the ingenuity of the designer.

A second system for assembling control circuits avoids the use of a clock. Instead, a series of delay units is used for generating the timed command pulses needed for transferring the instruction from the main storage to the operation-address register. Also, (in its most straightforward form) there is a set of delay devices to correspond to each operation the calculator is capable of performing. The basic concepts of this type of control will be explained with reference to Fig. 11-5, which shows an abbreviated set of circuits in block diagram form. Assume that a pulse is applied at the point marked X. This pulse will travel to the right in the figure, and as it passes through the delay units appropriately timed pulses may be taken from the junctions. The timed pulses are sent along command lines to the required units in the calculator for sensing the instruction in the main storage and transferring it to the operation-address register. This much of the functioning corresponds to an instruction cycle in the previous arrangement. The command lines are indicated in the figure as unmarked arrows pointing downward.

When the pulse emerges from the first set of delay devices it is applied to a set of "and" switches, one of which is opened by a signal from the matrix. The matrix, as before, is under control of the digits in the operation part of the operation-address register. The pulse will now travel along the set of delay devices corresponding to the particular operation which is to be performed. The path of the pulse from this point may now branch or loop in any of a variety of fashions under control of miscellaneous conditions in the calculator. For example, if the A_1 switch is the one which is open, the pulse will appear on command line C_1 or not according to the status of miscellaneous signal M_1. When the operation corresponding to A_2 is to be performed, the pulse will traverse a closed loop as long as no signal is present on M_2. Presumably one of the command lines, say C_2, leads to a counter which controls M_2, and when the loop has been traversed the desired number of times a signal will be applied to M_2 so that the pulse can leave the loop and continue to the right. Loops of this type are useful in multiplication, division, and shifting operations. In cases where the pulse passes through A_3, the path branches under control of M_3 so that two quite different sets of command lines can be pulsed with a minimum of switching. Regardless of which operation was executed, the pulse will eventually appear at one of the inputs to an "or" switch to be returned to point X. The next instruction will then be sensed and executed.

As with the other system of control, the problem of minimizing the number of delay devices and other components is a complex puzzle with no straightforward method of solution. It should be noted, however,

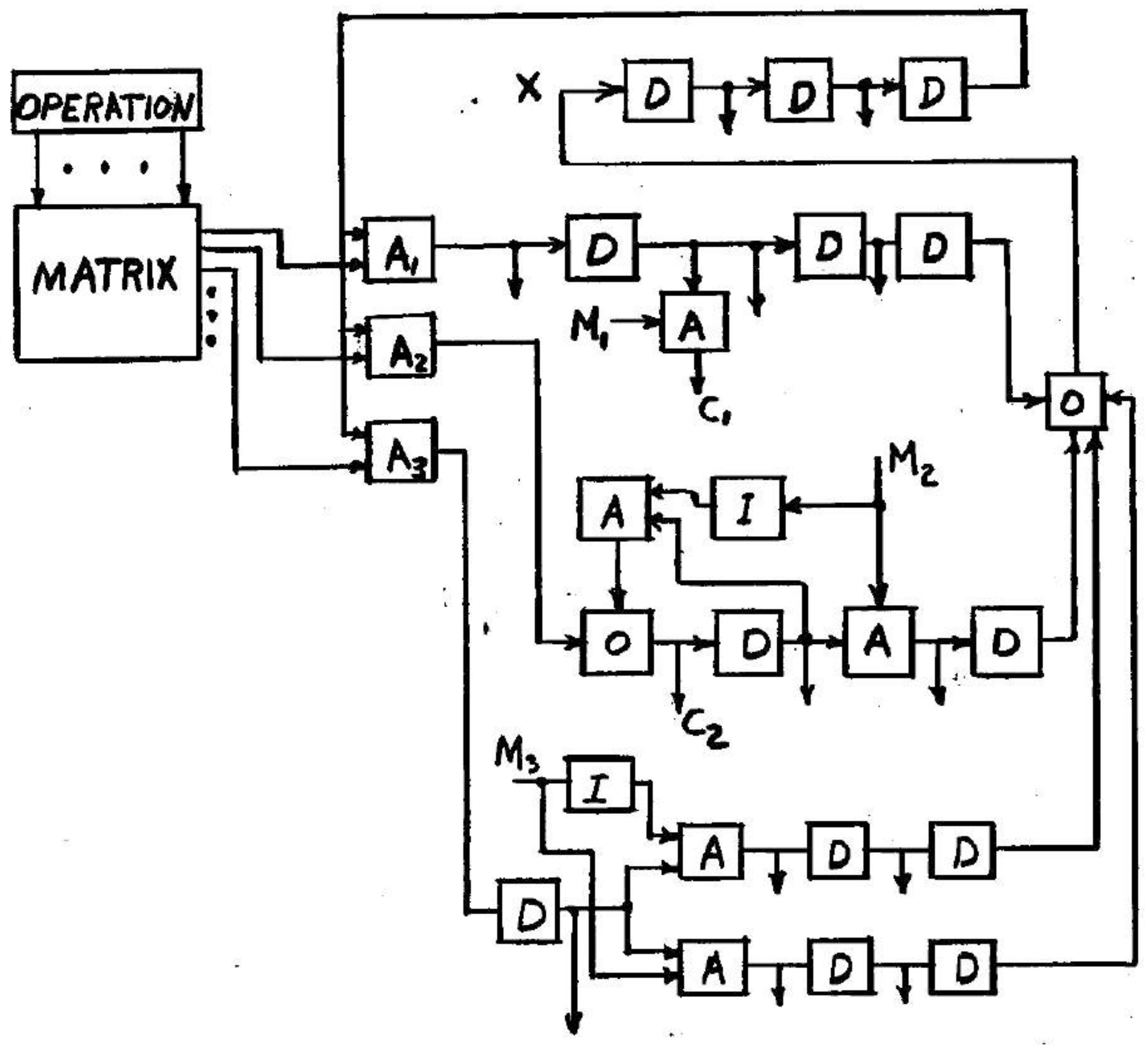


Fig. 11-4. Control Circuits.

that it is not necessary to have a different set of delay devices for each operation because in most calculators many of the operations are similar to one another. A single set of delay devices may then be used for all operations in a given category with signals from the matrix being used as miscellaneous inputs to gate the command pulses as required. In some calculators it may be necessary to operate the main storage or other units on a fixed cycle basis. In this case, synchronization problems may be encountered because it will not always be known which part of the calculator will finish first when certain operations, such as shift, are being performed. Synchronizing circuits similar to the ones described in the chapter on switching networks may be used to cause the unit finishing first to wait for the other.

Synchronous and Asynchronous Calculators. The two methods for designing control circuits which were described in the previous section are representative of two design philosophies, either one of which may be continued much further in the design of a calculator. In the first method, the timing of all operations is under control of the "clock," and, therefore, all operations take place in synchronism with the clock. Each operation requires an integral number of complete clock cycles. Because of the definite time relationship between the period of the clock and the execution of the operations, calculators functioning in this way are said to be "synchronous" calculators.

In an "asynchronous" calculator there is no fixed time reference for the execution of the operations. Instead, one operation is commenced as soon as the previous one is completed. To accomplish this purpose, the circuits must be arranged so that at the completion of each operation a signal is generated which may be used to initiate the next one. In Fig. 11-5 the delay devices and switching circuits were so arranged that the path of the pulse was continuous; when the pulse arrived at the end of one set of delay devices it was immediately entered into another set. The philosophy of asynchronous operation may be extended to apply to the individual parts of each operation. For example, the time required for handling the carries in an addition operation depends upon the number of carries that happen to occur. Rather than wait an amount of time required for the maximum number of carries, plus a safety factor, the asynchronous adder described in the chapter on binary addition and subtraction may be used. In this case the path of the control pulse would be through the carry circuits of the adder during this portion of the operation. The asynchronous multiplier which was described in the chapter on binary multiplication and division could be used in a similar manner. With this type of multiplier the loop for the control pulse would be in the multiplier itself instead of in the control circuits as shown in Fig. 11-5. The pulse would be circulated by passing it along the carry or the shift lines as dictated by the digits of the multiplier until a counter indicated that all multiplier digits had been sensed. Then, through a switch operated by the counter, the pulse would be returned to the control circuits.

It is not clear that either the synchronous or the asynchronous system has any outstanding advantages over the other. It would seem that the organization of a synchronous machine would tend to be more systematic and, therefore, perhaps easier to design, understand, and service. On the other hand, there is the possibility of making an asynchronous machine somewhat faster because it is not necessary to use a complete "cycle" of time for each operation. However, with either type of machine, the limiting factor with regard to speed is usually in the main storage unit because, in most cases, it has been possible to develop practical arithmetic circuits capable of operating on the numbers as fast as they can be taken from and sent to storage. An outstanding exception to the rule is in the operations of multiplication and division. For this reason it is sometimes desirable to execute multiplication and division is asynchronous fashion in a machine which is otherwise synchronous. Many other compromises in the two design philosophies are to be found, particularly in the input and output units, which cannot in a practical sense be synchronized with the very high electronic speeds of the arithmetic unit.

AC and DC Systems. Another criterion by which calculators may be classified is in the manner by which signals are transmitted through switching networks and from one part of the machine to another. In one type all of the signals are pulses. Even the storage registers are of the dynamic type, and the output signals from such devices are represented by the presence or absence of pulses. Machines in this category are known as AC (alternating current) calculators. The SEAC, which was built by the National Bureau of Standards, was the first calculator in which the AC technique was employed extensively.

At the other extreme is found a class of machines known as DC (direct current) calculators. In a DC calculator each signal is represented by a steady-state voltage, which is maintained at one of two levels according to whether the signal is a 1 or a 0. The voltage level is changed, of course, when the signal is changed, but during the time that the signal is being used it is held constant. All of the storage devices used in the registers are flip-flops or some other type of mechanism that produce steady-state output signals. All flip-flops are caused to transfer back and forth between their two stable states during calculations by means of "pull-over" circuits actuated by steady-state signals and not by pulses. An outstanding example of a DC calculator is the ORDVAC, which was built at the University of Illinois.

Not all machines can be classified as definitely one type or the other. For example, IBM's 604, 650, and 701 calculators each contain many circuits employing pulses as well as many circuits of the DC type.

It is not clear that either the AC or the DC technique is superior to the other. With the AC technique a serious design difficulty is encountered from the fact that all pulses applied to an "and" switch must arrive at the same time for an "and" function to exist. Since the passage of a pulse through almost any component introduces a delay, the design of an arrangement which will cause all pulses to arrive at the desired points at the desired times can be a very complex puzzle. Also, when using an inhibitor, it is necessary in practical circuits that the inhibiting pulse be applied slightly before the arrival of the pulse to be inhibited, and it must be maintained until a time which is slightly later than the pulse to be inhibited has disappeared. Further, the requirement of retiming and reshaping of signal pulses by means of clock pulses throughout the calculator is certainly an undesirable feature of the AC technique. On the other hand, the fact that pulses may be passed through condensers and transformers causes certain circuit isolation and voltage level setting problems to be much simpler with the AC technique than with the DC technique. Apparently no two machines, one AC and one DC, of sufficiently similar capabilities and history have been found for accurate comparison of such factors as cost (of the machine as a whole), reliability, and ease of servicing. These factors in the relative merits of the two design techniques seem to be matters of opinion.

Other Forms of Calculator Organization. The arrangements which were described in the sections on externally-programmed, plugged-program, and stored-program calculators do not exhaust the variations by which a calculator may be caused to proceed through the desired sequences of operations. For one thing, any two, or even all three of the sequencing means may be combined. As an example, IBM's Card Programmed Calculator (CPC) uses punched cards as an external programming means together with a plug board. Briefly, each card contains an instruction which causes the calculator to proceed through a certain set of program steps that are wired on the plug board. The instructions, as recorded on the cards, are, therefore, highly flexible. Combinations of stored program techniques and externally-programmed techniques may be found in almost any stored program calculator which includes an instruction for sensing an input device. Since new instructions as well as data may be entered through the input device, the program itself may be used to call in instructions from an external source for continuing the sequence. Some interesting possibilities can be visualized through the use of a plugged program together with a stored program, but this combination has not been exploited.

In the organization of a stored-program calculator it is not necessary that storage locations be used interchangeably for instructions and data. The Harvard MARK IV calculator, for example, is a stored-program machine which has separate storage locations and separate transmission paths for instructions and data. Arithmetic operations cannot be performed on instructions in the same sense that this function is possible in calculators with interchangeable instructions and data, but there is provision for altering or repeating sequences of operations under control of the computed results. There are advantages to keeping instructions and data separated, such as the relative simplicity of certain aspects of programming. However, the vast majority of stored-program calculators have been designed with interchangeable instructions and data because of the many useful programming procedures which are thereby made possible.

Another form of organization for a stored-program calculator can be obtained by using an idea found in the digital differential analyzer, which was explained in the preceding chapter. It might be described as a "no address" calculator in that each instruction specifies only an operation. The operation and the number or numbers to which it applies must be stored in the same location, that is, at the same address. The addresses are inspected sequentially, and at each address the indicated operation is performed on the data found there. Again, the possibilities are interesting, but so far as is known, no machine functioning in this way has ever been put to use.

Special-Purpose Calculators. For the most part, the calculator organizations which have been described are for "general-purpose" machines, where a general-purpose machine is one that is capable of solving a wide variety of problems the nature of which may not be known prior to the design of the machine. When a calculator is to be applied to a specific type of mathematical problem, special features may be incorporated to aid in its solution, and it then becomes known as a special-purpose calculator. These special features may range all the way from one or two special instructions in an otherwise general-purpose calculator to an entirely different organization throughout. An example of a machine with a radically different organization is the digital differential analyzer, which is intended primarily for the solution of differential equations. Another special-purpose calculator bearing little resemblance to any general-purpose machine is the USAF - Fairchild Specialized Digital Computer, which was designed specifically for solving simultaneous linear equations.

Word Length. The term, "word," is used to signify any group of digits that are handled as a unit in a calculator. Usually a word is one number when referring to data, or it is one instruction, although there are exceptions to this definition. Most calculators operate with a fixed number of digits in each word, and in the design of such machines a decision must be made concerning the number of digits in each word, or the word length, to be used. If the word length is made large enough to handle the largest numbers expected to be encountered, much storage space and other equipment will be wasted for those problems requiring less accuracy in their solution. If a short word length is chosen, two or more words may be used to store each number when long numbers are necessary, and arithmetic operations may be performed in parts by following a so-called "multiple-precision" technique. However, many extra program steps are required to execute multiple-precision operations with a resultant wast in storage space and also with a waste in time. Word lengths which have been used vary from 15 binary digits including sign (equivalent to about 4-1/2 decimal digits) in MIT's Whirlwind I to 23 decimal digits in the Harvard Mark I.

The choice of word length is most difficult for general-purpose calculators when it is not known what mathematical problems the machine is to solve. Some studies have indicated that the choice is not critical, but that there is an optimum word length for the "average" problem somewhere in the range of 10 to 12 decimal digits (or the binary equivalent).

The number of digits in an instruction, as well as the accuracy of calculations, enters into the choice of word length. The number of digits in an instruction depends upon the number of different operations the calculator is capable of performing, the number of addresses in each instruction, and the size of the addresses. In the interests of conserving storage space it is desirable to adjust the number of digits in an instruction or the number of digits in the word so that one is an integral multiple (usually one or two) of the other.

There is no inherent reason why the words have to be of fixed length. A calculator can, by means of appropriate instructions together with an appropriate organization, be caused to perform arithmetic and other operations on numbers of varying length. The increments in size may be as small or as large as desired. In a decimal machine, increments as small as one decimal digit are advantageous in which case each individual digit location in the main storage is addressable. An address in an instruction then refers to the first digit in a word, and the remaining digits are stored in successively higher number address positions. The number of digits to be used at any given time may be indicated by a special counter, by a special character in the accumulator, or by other means. Appropriate instructions must be included for altering the word length. Another scheme is to include the word length as a part of each instruction. Machines in this category are known as "variable word length" or "variable field length" calculators in contrast to "fixed word length" calculators.

Storage Capacity. Another important decision which must be made during the design of any calculator has to do with capacity of the storage medium. In order to solve very complex problems at high speed it is desirable to have a large storage capacity; yet, in the interests of high reliability and low cost, the storage capacity should be made as small as possible. Clearly, a compromise is required. However, in some cases other considerations arise that help in the determination of the proper storage capacity. For example, in a plugged-program calculator a rough upper limit to the number of storage locations that can be used profitably is set by the number of program steps available on the plug board.

In a stored-program calculator the upper limit to the number of storage locations in the main storage unit that can be used is set only by the problem to be solved. Nevertheless, beyond a certain point, a "law of diminishing returns" becomes apparent because of the possibility of introducing blocks of new data or instructions into storage at reasonably high speed. It should be noted that most calculations applied to sequenced calculators are highly repetitive in nature (otherwise, it would be as easy to solve them on a desk machine as to prepare the program). Because of this fact, the time required to complete the operations specified by one filling of the storage unit of a given size may be comparable with the time required for refilling the storage unit. When this condition exists, a larger size storage unit would increase the overall speed of the calculator by only a small factor. This line of reasoning applies best to calculators which use a random-access, high-speed, storage medium such as electrostatic tubes or magnetic cores for the main storage and which use magnetic tape or drums for auxiliary storage. For machines of this nature, the optimum main storage capacity appears to be in the range of 1000 to 4000 words.

Single-Address and Multiple-Address Calculators. As has been mentioned, a calculator may be organized with each instruction containing one, two, three, or four addresses. More than four would be possible, but the incremental advantages of each address beyond four diminishes rapidly.

For single-address calculators it is customary to execute the instructions as they are found in sequentially numbered addresses (except for jumps) under the control of an instruction counter in the manner described in the section on stored-program calculators. The address in the instruction may refer to the location of one of the operands entering into the calculations in the case of an arithmetic operation, or it may refer to the number of shifts, the identity of an input or output device, or something else in the case of a different kind of operation. With this arrangement a simple addition operation usually requires three instructions. The first instruction is used to cause one operand to be transferred from the specified address to the accumulator. The second instruction causes the other operand to be transferred from its specified address to the accumulator to be added to the first operand. Then the third instruction is used to transfer the sum from the accumulator to the desired address in storage.

The two addresses in a two-address calculator could be used to specify the locations of two operands, or a shift and the location of one operand, or any of a variety of other combinations. However, the second address is probably most commonly used to specify the location of the next instruction. With this arrangement, the instruction counter is not a counter; instead, it is only a storage register. A counting action is not needed because instructions are not necessarily taken from sequentially numbered storage locations; each instruction is essentially a jump instruction. For conditional jump operations, one address may be used to specify the location of the next instruction when the condition is satisfied (such as positive sign for the number in the accumulator at the time), and the other address may be used for the location of the next instruction when the condition is not satisfied. A two-address system with one address specifying the location of the next instruction is particularly useful in calculators employing magnetic drums (or some other storage medium which is not of a random-access type) for main storage. By judicious positioning of the instructions on the drum, the calculator can be caused to waste much less time waiting for the next instruction to appear at the sensing heads than when the instructions are positioned randomly. The finding of the proper addresses for the instructions is part of the procedure known as "minimum-access programming."

An example of the use of an instruction in a three-address calculator would be an arithmetic operation where the addresses of the two operands are specified as well as the address of the location where the result is to be stored. With this arrangement only one instruction would be required to perform the addition operation that required three instructions in the single-address machine. Several different variations are possible when one or more of the addresses are used to specify locations of subsequent instructions.

In a four-address calculator at least one of the addresses is almost always used for the location of the next instruction because there is little, if any, use for more than three addresses for other purposes.

As a general rule, as the number of addresses in each instruction is increased, the number of instructions required and the time required to solve any given problem are decreased because each instruction is able to accomplish more. On the other hand, a multiple-address calculator is usually somewhat more complex than a corresponding single-address machine. Also, more storage space is required for a multiple-address instruction than for a single-address instruction, and in many cases the extra space is largely wasted because the extra addresses are not always needed. There is some reason to believe that the preparation of a program can be made a simpler and more straightforward process for single-address machines than for multiple-address machines, but this point is subject to question. Successful machines of each type have been built.

Floating-point Calculators. The floating-point feature can be installed in any of the calculator organizations which have been described. In most cases it will probably be found desirable to use special counters for storing the exponent parts of the numbers entering into the calculations. For each arithmetic operation, the required shifts are made under control of the counters and under control of circuits for sensing non-zero digits in the shifting register. A pulse is sent to the exponent counter each time a shift is made. While a great many details in many parts of a calculator are affected by adoption of the floating-point system, no new principles are involved, and appropriate extensions of the control systems used for fixed-point calculators can be used.

Index Registers. Innumerable special features can be found in the various calculators which have been built. Among the more important features in some machines is a set of registers, called "index registers." In these machines, which are usually of the single-address type, each instruction specifies an index register as well as an operation and an address. For each operation, the number stored in the indicated index register is automatically added to the address, and the sum is then the actual address which is used. A "load index register" instruction is needed for transferring a number from storage into the indicated index register so that the numbers stored in the index registers may be changed during calculations. Index registers may be changed during calculations. Index registers facilitate the use of sub-programs as will be described in the next chapter. A calculator built at the University of Manchester in England was the first to employ index registers, although with this machine the term used to designate them (collectively) was "B tube."

Repeat Counter. Another useful feature is a "repeat counter" which can control the number of times an instruction is repeated. A "load repeat counter" instruction is used to transfer the desired number from storage to the repeat counter. Then, each time the instruction to be repeated is executed, a pulse is sent to the counter to count it down towards zero, and the repeating is continued until the counter reaches zero. Usually it is desirable to alter the address part of the instruction each time it is repeated. For this purpose, the register storing the address part of the instruction may be made capable of counting, and the pulses which are sent to the repeat counter may be sent to this counter also. Functions such as adding a long list of numbers stored at sequentially numbered addresses are very easily and rapidly accomplished with a repeat counter, where an iterative loop of some sort would be required otherwise. Also, the repeat counter is useful in transferring large blocks of information between the main storage and an input or output mechanism. In this case, the number of words to be transferred is placed in the repeat counter.

Input-Output. In some respects the subject of input and output mechanisms for a calculator can be as extensive as the subject of the arithmetic operations themselves. In particular, for calculators intended for accounting applications, the preparation of voluminous information for machine consumption and the subsequent preparation of records, reports, bills, checks, and other documents are problems that can easily overshadow the arithmetic problems. In many cases even the machine work is largely sorting, collating, and other non-mathematical operations that can be accomplished to some extent through appropriate control of multiple input and output mechanisms.

Relative to the arithmetic operations, the outstanding characteristic of practically all input and output devices is that they are slow in comparison with the speeds attainable with electronic components which may be used in the arithmetic and control portions of a calculator. There are some problems, particularly in the scientific field, that involve long sequences or operations on relatively small amounts of input and output data, and for these problems, input and output speeds are not important. However, in a surprisingly large number of cases even in the scientific field it is found that time required for input-output functions is an appreciable fraction of the time required for the calculations. For this reason, it is desirable that the calculator be organized in such a way that the calculations can proceed simultaneously with the input-output functions. One way of accomplishing this purpose is through the use of buffer storage. For example, if punched cards are the input medium, a card full of data may be transferred to buffer storage during the time that the calculator is processing the data corresponding to the previous card. At the completion of the operations, the data is transferred at high speed from the buffer storage to the main storage in the calculator. An alternate method is to arrange the functioning of the system so that arithmetic operations can be performed between individual operations of the input-output mechanism. In the card-input example, the few milliseconds of time between the sensing of individual holes may be sufficient to allow the calculator to execute a reasonably long sequence of arithmetic operations. Both of these schemes have been used successfully.

Error Detection and Correction. Because of the multiplicity of components in a large calculator, the detection and correction of errors is a constant battle. It would, of course, be desirable to develop components to a state of reliability such that a calculator would function for years without error. However, it is not essential that this objective be obtained; in fact, the success of calculating machinery is due in large measure to the fact that procedures have been worked out for making effective use of a calculator even when components are randomly and frequently deteriorating beyond the point of usefulness.

An error may be placed in one of the three categories according to whether it was caused by a complete failure of a component, the marginal operation of a component, or a random malfunctioning of a component. Actually, an engineer attempting to repair a defective calculator must consider other sources of errors. For example, the internal wiring may include some incorrect connections, particularly if the calculator is new or if modifications have been made recently. Also, in the course of searching for a defect it can happen that wires will be disconnected and then reconnected improperly. Further, the calculator may be in perfect condition with the defect occurring in the program. Even "tested" programs can fail when unforeseen and previously unencountered parameters are used. Here, it will be assumed that the calculator is designed properly and that the program is correct. For the most part, all errors regardless of source are detected by the same methods, but important differences arise in the means used for locating defective components.

One method of sensing errors is through the use of error-detection circuits, some of which have been discussed in previous chapters. The storage and transmission of information within a calculator is easily checked through the use of redundancy bits. Also, it is not difficult to incorporate checking circuits for adders and other arithmetic devices if appropriate codes are chosen. However, with many calculators it has not been found possible to devise practical checking facilities for the entire machine; the control circuits, in particular, can be very difficult to check completely. The other means of detecting the presence of errors is through programming. The nature of program checks depends upon the problem. For example, the solution to an algebraic equation may be checked by observing (through programming) whether or not the result fits the equation. For other types of problems it may be necessary to solve the problem by a second method, which is as different as possible from the first method, and compare the results.

After an error is detected either by the error-detection circuits, or by programming, the calculator may be automatically stopped for repairs. The source of an error is usually relatively easy to find when the error is caused by the complete failure of a component, for example, a filament burn-out in a tube. Test programs that require the functioning of all components in the machine or the suspected region of it are useful aids in locating components which have failed completely.

Probably a more common type of component failure is a gradual deterioration until a point is reached where the component causes intermittent failures. The operation of the component is then said to be "marginal." Test programs are helpful in locating marginal components although they are not when the marginal component creates errors only rarely. It is usually desirable to increase the frequency of the failure of the component by varying some parameter in the machine. For example, if the cathode emission of a tube in a flip-flop circuit is low, the operation of the circuit may be critically dependent upon the grid bias voltage. Therefore, by altering the grid bias the circuit may be caused to fail consistently so that it can be located easily. Since this method is useful for locating components about to fail as well as components which have already caused errors, the best way to treat errors caused by marginal components is to prevent them as much as possible by periodic machine inspection.

Sources of errors are most difficult to find when they are of a random nature. An example of a random error would be found in the case where a few loose flakes of cathode coating material in a tube were dropping off and causing temporary grid-to-cathode short circuits. Another example might be a condenser with a dielectric which occasionally breaks down with a very small arc and then heals. The search for sources of random errors is a puzzle with no fixed pattern for solution. Almost every calculator engineer seems to have had the experience of searching for many days or weeks for an elusive defect. Sometimes the frequency of the error can be increased somewhat, in which case the error would probably belong in the marginal category. In the example of the loose flakes of cathode material, a light tapping of the tube might cause the flakes to fall more frequently. On the other hand, after a few flakes have fallen the tube might, for a time, be better than before. In the case of the condenser the frequency of the arcing could probably be increased by increasing the voltage across it, but this step might be impractical because of the location of the condenser in the circuit. Defective connections in the wiring are another source of substantially random errors that are difficult to locate.

In general, error detection methods do not indicate the nature of the source of an error. Since an error may not be caused by the complete failure of a component, it may not recur for a long period of time. Therefore, it may be preferable to repeat the portion of the problem where the error occurred and then continue instead of stopping the machine. The repeating may be accomplished through the program, and the attention of an operator is not necessarily required. In some applications it is desirable to continue the calculations for other reasons. For example, if an error occurs in a calculator which is computing the direction of fire for an anti-aircraft gun, there is no point in correcting the mistake because the target will have moved. Yet, the next shot might be successful. In contrast to these examples, when searching for the source of an intermittent error, it is desirable to be able to stop the calculator on the very step where the error occurs. Then, by studying the status of information in the various registers it is sometimes possible to deduce the location of the defective component. For this purpose, error-detection circuits are necessary; program checks seldom detect an error until one or more steps later.

For very long programs, catastrophic failures such as loss of power are worrisome. To avoid having to restart a problem at the beginning when failure of this type occcurs, all intermediate results may be recorded at periodic intervals on punched cards, magnetic tape, or some other medium whereby they can be re-entered into the calculator. When this precaution is taken, calculations may be resumed at the point where intermediate results were last recorded. In this connection storage devices are often classified as "volatile" or "non-volatile" according to whether they lose or retain, respectively, information when power is removed. Electrostatic storage, for example, is volatile, and most magnetic storage devices are non-volatile. To make use of non-volatile storage in minimizing the effects of power failures, it would be necessary not only to "stop" the calculator before the voltages had collapsed an appreciable amount, but also it would be necessary to stop the calculator at a point where calculations could be resumed. The latter requirement might be difficult if a relatively slow-speed input-output device were in operation at the time.