Machine Maintenance

The problem of maintaining a large-scale electronic calculator is one worthy of extensive planning. It is felt, by the engineers associated with this project, that regularly-scheduled preventive maintenance is the best approach to the problem at this time.

To facilitate the location of incipient troubles, the 701 has the following features:

(a) Direct coupling is generally used throughout, thus permitting static checking of signals.
(b) All dc power supplies may be varied approximately ±10 per cent for the purpose of marginal checking.
(c) The basic pulse repetition frequency (normally 1 mc) may be varied from 600 kc to 1.2 mc.
(d) The Machine Cycle key switch located on the Operator's Panel permits the execution of an instruction, one machine cycle (12-microsecond period) at a time. For example, the instruction *Multiply* would require 38 depressions of the Machine Cycle key switch and in doing so would permit analysis of the partial product as it is being developed.
(e) Facilities are also available to permit the high-speed repetitive execution of any instruction. This type of operation is extremely important when a timing problem or an intermittent condition exists.

In addition to the above, a large number of diagnostic programs[11] have been prepared which may be used to test thoroughly the performance of selective portions of the 701 system. Errors that occur during these tests are recognized by means of the program and reported to the engineer by means of lights, punched cards or printed results. Sufficient data may be stored and printed at a high rate of speed to permit an easy analysis of the difficulty.

A complete set of manuals (16) have been prepared covering the installation, operation, programming and maintenance of the calculator.

[11] L. R. Walters, "Diagnostic programming techniques for the IBM type 701 electronic data-processing machine," *IRE Convention Record;* 1953.

Conclusions

The IBM 701 is a large-scale, high-speed, electronic digital computer that is adaptable to a wide variety of problems. It incorporates the latest development in three principal types of storage and provides sufficient storage capacity to accommodate the many complex problems existing today.

A versatile input-output system using magnetic tapes, magnetic drums, the proven punched card and a line-at-a-time Printer, provides rapid communication between the machine and the operator. These features make possible the efficient use of a high-speed computer on problems containing large quantities of input data and resulting in many pages of printed results.

Acknowledgments

The Type 701 was developed at the IBM Engineering Laboratory, Poughkeepsie, N. Y. by groups under the direction of N. Rochester and J. A. Haddad.

The author wishes to acknowledge the assistance rendered by his associates in the preparation of this paper.

The Arithmetic Element of the IBM Type 701 Computer*

HAROLD D. ROSS, Jr.†, ASSOCIATE MEMBER, IRE

Summary—**The arithmetic element of the IBM Type 701 Computer contains a number of innovations, both in circuits and in logic, which distinguish it from that used in other high-speed, parallel, binary computers. The most significant of these is the storage element used in the arithmetic registers, in place of the more usual flip-flop or trigger circuit. Use of this new element enables a number of simplifications to be introduced in the methods of executing instructions.**

Introduction

A NUMBER OF large-scale, parallel, binary digital computers are now in operation and the circuits and logic used in their arithmetic units include a variety of techniques. Most of these machines make use of the flip-flop or trigger circuit as the basic storage component in arithmetic registers and at least one uses a re-circulating pulse type of storage device.

A still different type of storage is employed in the arithmetic element of the IBM Type 701 Electronic Data Processing Machines and Associated Equipment, the large scale, high-speed, electronic computer installations now in production by the International Business Machines Corporation.[1,2] This dynamic type circuit has the property that its output level during each one-microsecond interval is the same as the input level during

* Decimal classification: 621.375.2. Original manuscript received by the Institute, June 23, 1953; revised manuscript received July 23, 1953.

† International Business Machines Corp., Engineering Lab., Poughkeepsie, N. Y.

[1] C. E. Frizzell, "Engineering description of the IBM type 701 computer," Proc. I.R.E., pp. 1275–1287, this issue.
[2] W. Buchholz, "System design features of the IBM type 701 computer," Proc. I.R.E., pp. 1262–1275, this issue.

the previous one-microsecond period; that is, there is a one-microsecond delay through the unit. The output bears a marked resemblance to that of a flip-flop in that it is essentially a dc signal at one of two different levels. This circuit is used in conjunction with dc-coupled diode switching in order to achieve a number of simplifications in arithmetic processes.

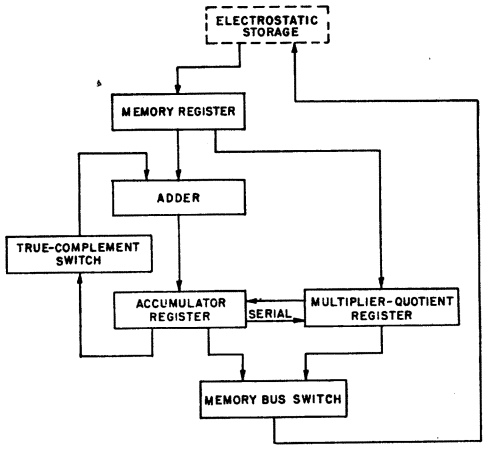


Fig. 1—Type 701 arithmetic element.

Arithmetic Element Block Diagram

A simplified block diagram of the arithmetic element appears in Fig. 1. In this diagram the major components are shown, with the paths of information flow during execution of an operation. The characteristics of the several units which make up the arithmetic element are:

1. Memory register. In executing an instruction requiring a word from memory, the word first appears in this register. It is not capable of shifting.
2. Multiplier-quotient register. This register may be entered from the memory register directly, or from the accumulator register by shifting. On input-output operation it acts as a buffer register. It is capable of shifting to the left or right. It holds the multiplier during multiplication and receives the quotient during division.
3. Accumulator register. In such operations as addition, this register holds one of the two operands and is connected to the adder by true-complement switching. It is capable of shifting to the right or left, either alone or in conjunction with the multiplier-quotient register.
4. Adder. This unit is mainly a switching array in which the outputs of the accumulator and memory registers are added together and the sum fed back to the accumulator register. It is described more fully in a later section.
5. Memory bus switching. This unit contains the switches necessary to connect the output of either the accumulator or multiplier-quotient registers to the electrostatic storage unit in order to store a word. Either the entire contents or only the more significant half of each register may be stored.
6. True-complement switching. This consists of an inverter and switching for each column of the accumulator register to permit either the true or inverted accumulator register output, or neither, to be connected to the adder unit.

Microsecond Delay Unit

The three arithmetic registers make use of a new storage unit called the microsecond delay unit. This circuit stores binary information in a dynamic fashion but its output appears as substantially a dc level of either +10 or −30 v, depending upon the binary digit stored. When suitable input switching is provided, information may be entered from one of several sources, such as another register or from the adjacent columns of the same register. When new information is not being entered, the status of the unit is maintained by an external feedback connection from its output to its input. As used in the 701, the output does not assume the new level until one microsecond after the input signal is applied.

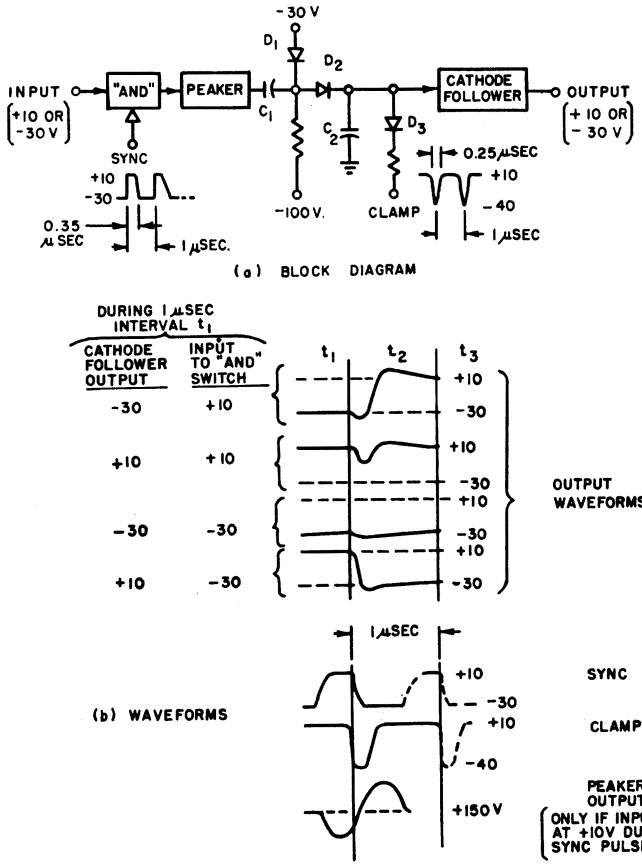


Fig. 2—Microsecond delay unit.

A simplified diagram illustrating the operation of this circuit is shown in Fig. 2a (a schematic diagram of the delay unit is given in Fig. 12). To facilitate understanding the operation of the circuit it is divided into four parts which are (1) an "and" switch in which the level of the input signal, either +10 or −30 v, is sampled by a sync pulse of about 0.35 μsec duration; (2) a vacuum tube peaker circuit driven by the "and" switch; (3) a

capacitor (and associated circuits) which stores the pulse energy from the peaker circuit for one μsec; (4) an output cathode follower driven by the capacitor.

Circuit Operation

In describing the operation of this circuit, there are four possible conditions to consider. These possibilities are listed in Fig. 2b. It should be pointed out here that the output continues unchanged until one μsec after the new input is applied, hence the table of Fig. 2b shows the various combinations of input and output that may be present simultaneously. Considering the first of these conditions, the input signal during t_1 is at +10 v and is applied to the diode "and" switch together with the sync pulse. Under these conditions, the sync pulse is gated through to the peaker tube which will conduct for the duration of the pulse. At this time a negative pulse will be coupled over to the junction of C_1, D_1 and D_2 but will go no farther because D_1 will conduct, holding this point to −30 volts. After the peaker tube is cut off, a positive pulse will be produced, of about 0.5 μsec duration, which will be coupled to the storage capacitor C_2 via C_1, D_1, and D_2. Since the output of the unit was assumed to be −30 v, this was the charge on C_2 (consideration of how this is done will be postponed momentarily). The positive pulse from the peaker charges C_2 to +10 v. The upper level is limited by conduction of D_3 when the normal +10 v upper level of the clamp signal line is exceeded. The voltage across C_2 is directly coupled to the output cathode follower stage. Fig. 2b shows the waveforms mentioned above.

The process just described began with the output of the unit being a −30 v level; consider now the process when the output is initially +10 v. An input level of +10 v will again gate the sync pulse through to the peaker tube, causing it to conduct and generate a negative pulse followed by a positive one. Once again the latter pulse proceeds via C_1, D_1 and D_2 to charge C_2 to +10 v. In the present case C_2 was already charged to +10 (except for leakage through the back resistance of D_2). The +10 v charge is renewed and the output simply continues at +10 v with this reservation: occurrence of the clamp pulse causes partial discharge of C_2, coming as it does during the initial portion of the positive peaker output pulse. Since the peaker output is of longer duration than the clamp, in the end it prevails and charges C_2 to +10 v as shown in Fig. 2b.

Both of the above descriptions assumed an input signal level of +10 v at the time the sync pulse occurred. Consider now the action if the input level is −30 v; coincidence will not occur in the "and" switch and the peaker tube will not conduct, consequently no charging pulse will be generated. If the charge on C_2 had previously been −30 v, it might have leaked off to some extent, through the back resistance of D_3 to the +10 upper level of the "clamp" signal. Fall of the clamp to −40 v will restore the charge. If, on the other hand, C_2 had previously been charged to +10 v by the process described, and no positive charging pulse is generated by the peaker, the clamp pulse will return the charge on C_2 to −30 v. These processes are illustrated by the last two output wave forms of Fig. 2b.

The delay feature is thus due partly to the action of the peaker circuit and partly to the storage capacitor, C_2. The usefulness of this delay is that the 0.6 μsec period while the sync pulse is at −30 v may be used to switch the input of the unit between several sources, one of which is ordinarily the output of the delay unit itself. When the unit is so connected, it maintains its dc signal level at either +10 or −30 v until this connection is interrupted and a new source connected to the unit. This connection is made for one μsec, following which the delay unit input is connected back to its output in order to retain the new information. The only requirement in switching the delay unit input is that the new input signal must have reached its final dc level by the time the sync pulse appears.

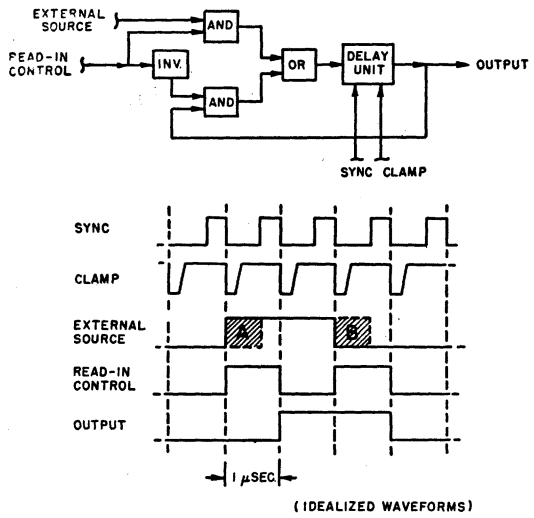


Fig. 3—Delay unit input switching.

Input Switching Associated with the Delay Unit

Fig. 3 shows a simple block diagram of a delay unit with the switching required to maintain continuously a +10 or −30 v level or to enter new information from an external source. Normally the "read-in" control line is at −30 v. When it is desired to enter new information into the delay unit this line is raised to +10 v for one μsec. Since the read-in line is normally at −30 v, the inverter output is normally +10 v, the lower "and" switch is conditioned and the delay unit output is connected to its input. Pulsing the read-in line for one μsec conditions the upper "and" switch and de-conditions the lower, effectively breaking the connection from the delay unit output. As the result of this one-microsecond "look" at the external source, the unit assumes the status of that source and at the end of the read-in signal, goes back to reading its own output. The occurrence of the read-in pulse is suitably timed with respect to the sync and clamp pulses as shown in Fig. 3. In this figure, the delay unit initially stored a −30 v level

(corresponding to a binary zero). The first read-in pulse caused the unit to store a +10 v level (corresponding to a one). Two μsec later a second read-in pulse occurred. At this time the external source presented a −30 v level and hence a zero was stored by the delay unit.

Another function performed by the action of the delay unit is that of re-timing; this is indicated by the shaded areas of the external source waveforms in Fig. 3. The shaded area *A* indicates that the rise of the waveform may occur anywhere within this interval and still give the output change at the time shown. Similarly, area *B* indicates that the fall of the waveform may take place anywhere within this interval and yet give correct operation. The re-timing is due to the fact that the input waveform is sampled by the sync pulse and effectively compensates for the effect of small accumulated delays.

Delay Unit Shifting Registers

If the "external source" for the circuit shown in Fig. 3 was the output of the adjacent delay unit of a register, a shifting register would result. A simple shifting register or ring, having no input switching, is shown in Fig. 4 with actual waveforms at several points. A pattern of two ones and a zero is shown, with the displacement of the pattern at successive stages by one microsecond, being evident.

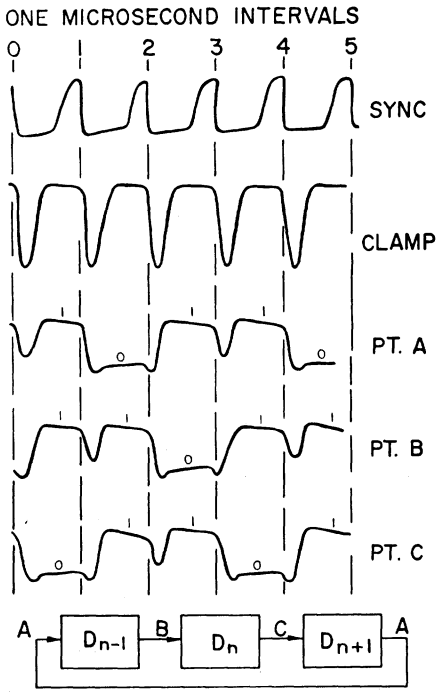


Fig. 4—Waveforms in a delay unit ring.

Fig. 5 shows three columns of a register, capable of shifting to the right or left as well as holding information statically. Each column consists of a delay unit with a three-way switch enabling it to read the output of the unit to the left, read its own output or that of the unit to the right. The function to be performed depends on the condition of the "shift right" and "shift left" control lines at the bottom. Normally these lines are at −30 v, de-conditioning the left and right "and" switch in each column. However, the center "and" switch is conditioned by way of the two-input "or" switch and inverter at the lower left. The output of this circuit is called the "hold" control line, since it conditions the center "and" switches which cause each delay unit to "hold" their status for an indefinite period. Shifting to the left by one column occurs if the "shift left" control line is pulsed to +10 v for one μsec at the proper time with respect to the sync and clamp pulses, in the same manner as the read-in control in Fig. 3. Pulsing either shift control line will cause the "hold" control line to go negative for the same period. A two μsec pulse will cause a shift of two places, and so on.

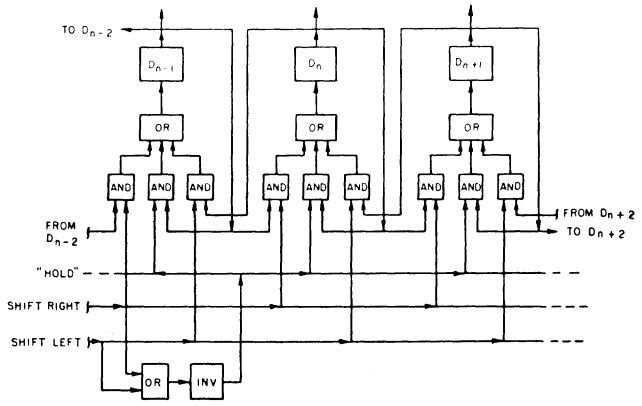


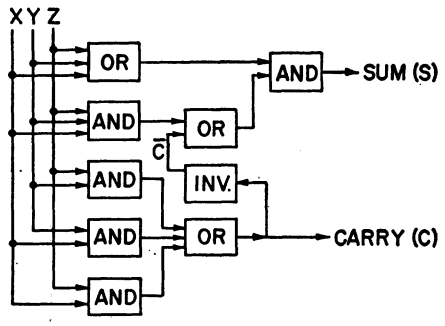
Fig. 5—Shifting register switching.

The circuit shown in Fig. 5, with up to six different inputs per column, forms the basis of the arithmetic registers used in the 701. It is felt that a high order of reliability, logical flexibility and freedom from close timing conditions has resulted from its use. Some of these points will be discussed further in the sections to follow.

Parallel Binary Adder

The means for performing addition in the 701 makes use of logical adders which are passive diode switching circuits of the type shown in Fig. 6. Also shown there is the "truth table" for this circuit. In both the circuit and the table *X*, *Y*, and *Z* are the three inputs, representing one digit each of the addend and augend, together with the carry to the next more significant column. The outputs are of course the sum and the carry from the next less significant column. From examination of the truth table it is evident that a carry is generated whenever there are signals present on two or more inputs and this fact is set forth in Boolean notation by the first logical equation. Further examination shows that a sum of one will be produced under two conditions: if there are signals present at all three inputs and if there is a signal present at one and only one input. This second condition may be re-phrased as fol-

lows: a sum of one will occur if there is a signal present at at least one of the inputs provided there is no carry output generated. This will also be evident from examination of the table. This last condition is exemplified by second Boolean equation in Fig. 6. Having these expressions for sum and carry, circuit follows at once.



(a) LOGICAL ADDER

X	Y	Z	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
1	0	0	1	0
0	1	1	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$C = XY + YZ + XZ$

$S = (X+Y+Z)(XYZ+\bar{C})$

(b) "TRUTH TABLE"

Fig. 6—Three-input binary adder.

It should be mentioned that the circuit as shown in Fig. 6 contains logical elements only; cathode followers required to drive the diode circuits have been omitted. As a result of including the necessary cathode followers a loss in signal amplitude is encountered resulting in a rise in the lower level of the carry signal of about three volts per follower. It thereby becomes necessary to introduce means for restoration of signal amplitudes and these means, which involve over-driven stages, introduce delays into the circuit operation.

Inasmuch as the 701 performs addition in parallel upon 37 columns (35 bits plus two accumulator overflow positions) the delay mentioned above accumulates so that for a full length carry about 3.5 μsec are required. Actually about five μsec are allowed for this process. Since facilities for addition, but not subtraction, are built into the computer, provision must be made to complement one of the operands in order to carry out the latter operation. Complementing is performed on but one of the two factors, namely the one from the accumulator register and a simple diode switch and inverter are required in each column to connect the adder to the accumulator register contents either as a true or a complement number.

Logical Aspects of the Operation "ADD"

In the 701 all numbers are represented as magnitudes with signs. Furthermore, means are provided to obtain the complement of only the accumulator register contents in subtraction, it having been determined that the occurrence of zeros of either sign was not only acceptable but desirable. Fig. 7 contains examples to illustrate the rules by which the form of the result of an arithmetic subtraction is determined. In case 1, the ones' complement of a positive number in the accumulator register is added to a larger negative number in the memory register. An end-around carry results, indicating that the result is in true form and that the sign of the accumulator register must change. In case 2, the magnitudes are reversed and the absence of an end carry indicates a result in ones' complement form and that the accumulator register sign will not change. In order to put the result into true form, a re-complementing process is undergone. Cases 3 and 4 are similar except that, since the magnitudes of the factors are alike, the result will be zero in each case. Here, the sign of the zero result is the initial sign of the accumulator register.

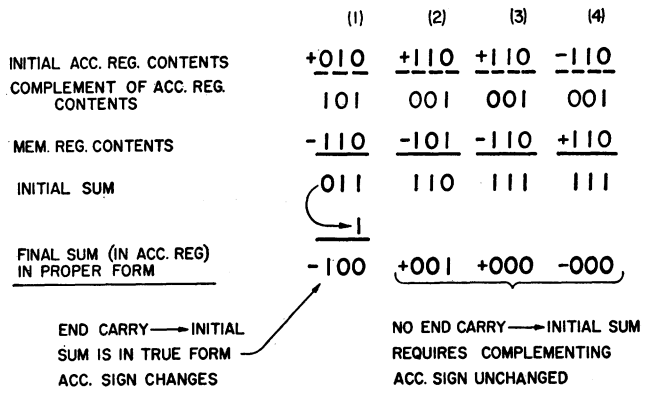


Fig. 7—Examples of ones' complement arithmetic when only accumulator operand can be complemented.

The ones' complement system, while causing a certain inconvenience in having to deal with end-around carries, makes the re-complementing process less painful, in that no carries are produced. The twos' complement system would have required allotting time for possible lengthy carries in re-complementing, as illustrated in Fig. 8. In other words, re-complementing in the ones'

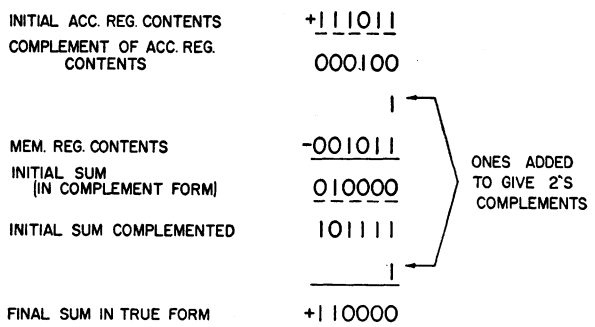


Fig. 8—Example showing occurrence of long carry in changing a twos' complement sum to a true number.

complement system consists simply of inversion, a process requiring little time, whereas in the twos' complement system a complete addition is involved. The table in Fig. 9 summarizes the situations that may arise in adding two numbers of varying sign and magnitude.

SIGN OF FACTOR IN ACCUMULATOR	SIGN OF FACTOR IN MEMORY	RELATIVE MAGNITUDES	FORM OF INITIAL SUM	IS RECOMPLEMENTING REQUIRED?	SIGN OF FINAL, TRUE SUM
+	+	——	TRUE	——	+
−	−	——	TRUE	——	−
−	+	$\|M\| > \|A\|$	TRUE	NO	+
−	+	$\|M\| < \|A\|$	COMPL.	YES	−
−	+	$\|M\| = \|A\|$	COMPL.	YES	−
+	−	$\|M\| > \|A\|$	TRUE	NO	−
+	−	$\|M\| < \|A\|$	COMPL.	YES	+
+	−	$\|M\| = \|A\|$	COMPL.	YES	+

Fig. 9—Logic of sum generation and addition.

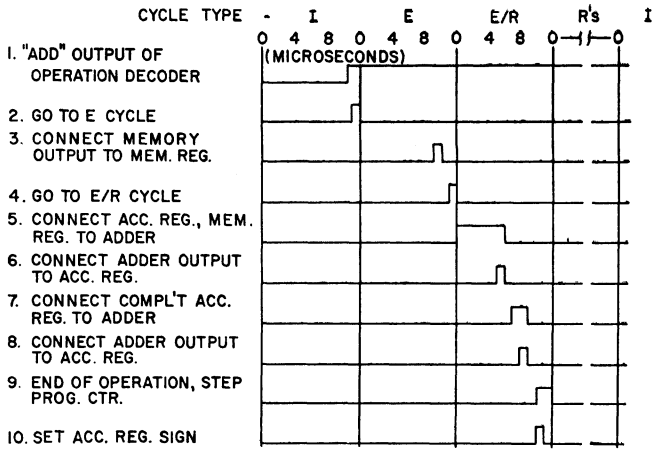


Fig. 10—Timing for the operation "ADD."

Control and Timing of the Operation "ADD"

The timing of the various commands necessary for the execution of the operation ADD is illustrated in Fig. 10. Execution of any operation begins when the output of the operation decoder assumes its new value, near the end of an "*I*" (Instruction) cycle. In this particular case, the output was the one signifying the operation ADD, hence an "*E*" (Execution) cycle is desired next. An *E* cycle is required whenever a reference to memory is needed, either to store a number or to withdraw one from memory. After nine microseconds, the word from memory is available to be entered into the memory register. There being insufficient time remaining in the cycle for the commands yet to come, an "*E/R*" (Execution/Regeneration) cycle is next called for. In this cycle, two full words of the electrostatic memory are regenerated while the actual addition is done. Addition begins by connecting the memory register to the adder and the accumulator register to the adder in true or complement form as required. After five μsec, the output of the adder is sampled and set into the accumulator register in place of the operand originally there. Depending on the relative magnitudes and signs of the operands, the result at this point may be in complement form as described in the preceding section. If so, the complement of the new accumulator register contents is connected to the adder for two μsec; the other adder input is left disconnected and represents a row of 35 zeros. After one μsec the new adder output is sampled and the result, now in true form, returned to the accumulator register. Finally, the sign of the accumulator register is set, according to the table in Fig. 9, and a signal sent out indicating the execution of the operation is complete. This last pulse also advanced the program counter one step in preparation for obtaining the next instruction. Following the *E/R* cycle, two "*R*" (regeneration only) cycles may be called for to satisfy the regeneration requirements of electrostatic memory.

For different operations, the sequence of cycles will vary, depending upon the need for a reference to memory, and upon the amount of time needed for manipulation of words by adding, shifting, and so on. The arithmetic speed is limited by timing requirements of electrostatic storage rather than by arithmetic circuits themselves. Advantage was taken of this where possible to avoid using faster, more elaborate circuits.

The Control of Shifting

The number of places by which the contents of the accumulator register or the combined accumulator and multiplier quotient registers are to be shifted, is controlled by a counter into which the address part of the shift instruction is entered. As shifting proceeds, pulses are sent to this counter, causing it to count down in synchronism with the shifting process. When the contents of the counter reach zero, this fact is sensed by an "and" switch and the shift gate to the registers involved is terminated. Shifting takes place at a one megacycle rate and up to eight places may be traversed within a 12 μsec cycle. A variable number of cycles is required for the execution of a shift depending upon the length of the shift. The maximum length of a shift is 255 places; such a shift, while resulting in a string of zeros in the registers affected, facilitates the programming of floating-point computations.

Multiplication

Multiplication in the 701 consists of repeated addition and shifting, the addition being under control of the currently least significant multiplier digit. Thirty-six successive 12-μsec *E/R* cycles are required for the 35×35 multiplication, following the *I* and *E* cycles used to obtain the instruction itself and the multiplicand. During each *E/R* cycle one addition of the multiplicand may occur together with a shift of the accumulator and *MQ* registers to the right. These functions require but little more than half the 12 μsec allotted; the actual time taken for multiplication might have been reduced by packing the successive additions closer together but in practice as the additional time is used to regenerate the electrostatic storage, the time expended is recouped in that succeeding instructions may be executed in less time. By using the 12-μsec ADD-type *E/R* cycle, control of multiplication is simplified.

Division

One of the consequences of the use of dc coupled diode adders and delay unit registers is that division becomes somewhat simpler than would otherwise be the case. A system which is superior to either the restoring or nonrestoring scheme is used; it is made possible by the ability to examine the result of a subtraction before discarding the original number. Thus, if the result of a subtraction is an overdraw, this result may be ignored (that is, not put into the accumulator register) and the previous remainder retained and shifted left, whereupon another subtraction may be tried. If the subtraction resulted in a remainder of unaltered sign, the result will be in complement form and must be complemented prior to shifting left. In the former case a quotient digit of zero is inserted into the lowest order multiplier-quotient register column prior to shifting and in the latter case a quotient digit of one is inserted. This is repeated in successive cycles, so that the quotient builds up in the *MQ* register on the right as the less significant part of the dividend is shifted out into the accumulator register on the left.

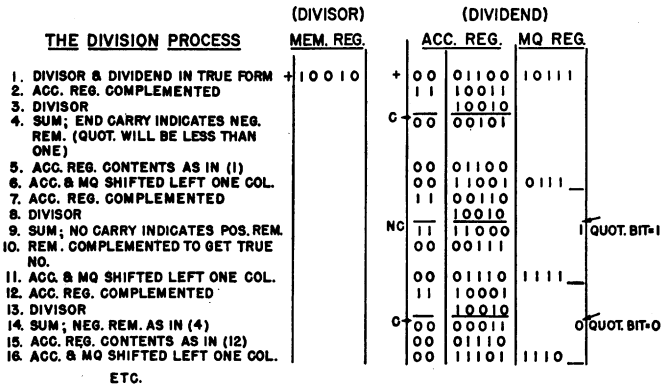


Fig. 11—The division process.

The process just described is illustrated in Fig. 11. Here a typical division is shown using abbreviated binary numbers of five-bit length (the signs are handled separately). The repeated subtraction cycle is shown for two different cases in steps 6–10 and 11–15. In the first case the result of subtraction was a positive remainder; this fact is indicated by the absence of an end carry which therefore causes the adder output to be inserted back into the accumulator register. In the second case a carry appeared and so the adder output was ignored, the previous remainder shifted to the left, and a new subtraction started, thus it is seen that no restoration is necessary. A further point is illustrated by the example: namely, the method of testing for a quotient less than one. The first four lines illustrate this; the divisor is subtracted from the dividend with no preliminary shift. An end carry in this process indicates an overdraw and consequently a quotient of less than one. If no carry were obtained, an alarm would be generated and calculation stopped. In making this test the end carry is the only result desired, the sum is not put in the accumulator register. Actual division then begins with a shift to the left of the original accumulator register contents together with the *MQ* register.

The complementing cycle mentioned above, which is required when subtraction did not cause a sign change, could have been avoided by use of additional switching but there was little to be gained by the increase in speed, as ADD-type cycles of the sort described previously are made use of here without change in timing. Thus 36 cycles are required for the complete operation, resulting in a surplus of regeneration for the electrostatic memory. This surplus usually enables some regeneration to be dispensed with in the operations immediately following so that a portion of the time spent in dividing is usually made up elsewhere.

Testing for Zero in the Accumulator

The incorporation of the operation "Transfer on Zero" requires a method for detecting whether the accumulator register contains zero. The contents of the accumulator register are to be unchanged as a result of this test and this requirement is easily complied with by the scheme described below. Once again, the use of delay units in the accumulator register simplifies the process.

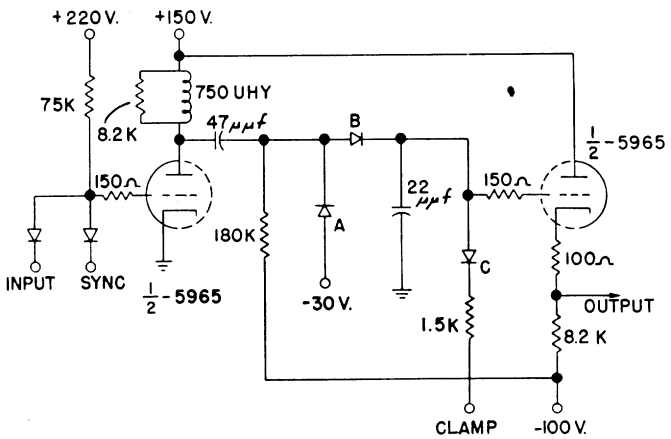


Fig. 12—The delay unit circuit.

The process consists of adding the complement of the accumulator register contents, to zero (obtained by leaving the memory register disconnected from the adder) and adding one in the lowest order. If the contents were initially zero, a full length carry will result, giving a carry from the highest order. This carry is used to cause the next instruction to be taken from the address specified in the Transfer on Zero instruction. The adder output is not set into the accumulator and hence the contents are undisturbed. This ability to examine the adder output without putting it into the accumulator register is facilitated by the one-microsecond delay in the delay units and the use of dc coupled diode adders.

Conclusion

The arithmetic element of the IBM Type 701 Computer makes use of a new electronic storage circuit, the microsecond delay unit, whose output may be either of two dc levels, yet uses dynamic pulse storage techniques. Through the use of this device, in conjunction with dc coupled diode adding circuits, important simplifications are realized in shifting registers, and in the execution of division and testing for zero in the accumulator.

Acknowledgment

The microsecond delay unit described herein was developed by B. L. Havens of the Watson Scientific Computing Laboratory. His support, encouragement, and assistance contributed greatly to the success of the 701 project and his kind permission to describe this unit is greatly appreciated by the author. The author also acknowledges the valuable assistance of other members of the project in the preparation of this paper.

The SWAC-Design Features and Operating Experience*

H. D. HUSKEY†‡, ASSOCIATE, IRE, R. THORENSEN†, ASSOCIATE, IRE, B. F. AMBROSIO†, ASSOCIATE, IRE, AND E. C. YOWELL†

Summary—**The SWAC is an ultra-high-speed digital computer utilizing a Williams tube memory, an auxiliary magnetic drum memory and a punched card input-output system. A general description of the functional organization of the computer is given together with a brief discussion of the various commands and how they are executed.**

Some of the special engineering features of the computer are described, in particular those relating to the electrostatic and magnetic drum memories.

Finally a short survey of the types of problems solved by the computer during the last year is presented.

Introduction

THE NATIONAL BUREAU OF STANDARDS Western Automatic Computer (SWAC), located in Los Angeles at the Institute for Numerical Analysis of the National Bureau of Standards, is a digital computer utilizing a Williams tube memory,[1] an auxiliary memory in the form of a magnetic drum,[2,3] and a punched card input-output system. A primary feature of the SWAC is its high speed, which, for handling the binary equivalent of eleven decimal digit numbers, is greater than that of any other computer now in operation.

The SWAC stores in its high-speed memory the numbers involved in the computation and also all the instructions necessary to perform the calculation. This makes it possible for the calculator to utilize fully the speed of the Williams tube memory, doing complete arithmetic operations in a few microseconds. Instead of handling numbers as a train of pulses, there are parallel circuits in the SWAC which transfer numbers almost instantly (actually the transfer takes place in the width of a pulse which in some cases is one-tenth μsec). This transferring of numbers in parallel makes it possible to do computations at many times the speed of serial computers (for example, acoustic delay-line machines) without having excessively high pulse rates on the lines of the computer.

The SWAC is the first of the Williams tube computers to be completed in this country. At present it is producing useful results during seventy per cent of the time that its power is turned on.

Brief Description of the SWAC

The Williams tube memory in the SWAC stores 256 words (numbers or instructions). The various digits of a particular number are stored in corresponding positions on each of 37 cathode-ray tubes. A word in SWAC is 37 binary digits long, which is the equivalent of about 11 decimal digits. A Williams tube memory requires regeneration. To accomplish this, time is divided into 8-μsec intervals. During one of these intervals a particular number in the memory may be regenerated, read out into the arithmetic unit, or replaced with a new number. The regeneration takes place during alternate 8-μsec intervals with the whole memory being regenerated once each 4,096 μsec. The other 8-μsec periods are called action intervals, and it is during these periods that numbers may be transferred between the memory and the arithmetic unit.

The SWAC uses eight basic commands: add, subtract, multiply (rounded-off answer), product (two word answer), compare, extract, input, and output. The compare command has a variation which compares absolute values. There is a special form of input used for initially

* Decimal classification: 621.375.2. Original manuscript received by the Institute, June 8, 1953; The preparation of this paper was sponsored (in part) by the Office of Scientific Research, U. S. Air Force.

† National Bureau of Standards, Los Angeles.
‡ Wayne University, Detroit, Michigan.

[1] F. C. Williams, and T. Kilburn, "A storage system for use with binary digital computing machines," *Proc. IEE*, Part III, pp. 81–100; March, 1949.

[2] R. Thorensen, "Design Features of a Magnetic Drum Memory for the National Bureau of Standards Western Automatic Computer (SWAC)," *Proc. Electronic Computer Symposium;* April 30, May 1–2, 1952, Los Angeles, Calif.

[3] A. A. Cohen, "Magnetic Drum Storage for Digital Information Processing Systems," Mathematical Tables and Other Aids to Computation, vol. IV, No. 29, pp. 31–39; January, 1950.