CONFIDENTIAL

K. Olsen

6889
Engineering Note E-487

FOR INTERNAL DISTRIBUTION ONLY

Digital Computer Laboratory
Massachusetts Institute of Technology
Cambridge, Massachusetts

Subject: NOTES ON THE LOGICAL DESIGN OF THE IBM 701 COMPUTER

To: M. M. Astrahan and N. H. Taylor

From: R. P. Mayer

Date: October 8, 1952

Abstract: The IBM 701 computer has logical characteristics similar to WWI except for terminology, the use of half-words, the use of a slightly different central control system, the use of diode logical circuits for handling both voltage "levels" and pulses, etc.

IMPORTANT NOTE:

Please consider the contents of this report COMMERCIALLY CONFIDENTIAL. This paper has been prepared to furnish information on the progress of new developments in the engineering laboratories of IBM. Since the material contained herein is of recent date, it is requested that the recipients confine its use to IBM personnel and MIT Project Lincoln personnel who need to have this information.

1.* Introduction.**

The general physical outline of the 701 computer is shown in the floor plan of Figure 1. Actually, only the control unit is designated by the number 701, while the other units have other numbers. The over-all system has no number, but is called an Electronic Data Processing Machine (EDPM). The title "Defense Calculator" is obsolete.

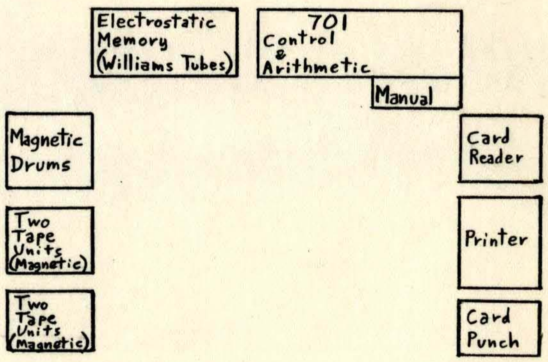


Fig. 1. Physical Outline (Floor Plan)

* Appendix F is a table of contents.

** This paper has not been fully checked by IBM and may contain some errors.

CONFIDENTIAL

1.1. Drawings

The system drawings for the "701 and associated equipment" are bound in two volumes. These are indexed by block outlines which show a system number for each block, and these numbers correspond roughly to the page numbers in the books. These system drawings are called block diagrams, but they are almost exactly equivalent to our block schematics. The 701 system has no drawings like our block diagrams.

1.2. Characteristics.

The general logical characteristics of the 701 system are tabulated in Appendix A, which should be studied at this point. Details on the circuit of the "Havens delay unit" may be obtained from Dick Best.

1.2.1. Word Length.

Each half-word is 18 bits long and has its own address. All instructions, and any numbers, are stored as half-words. In this sense, the half-word is almost exactly like the WWI "word".

The arithmetic registers normally work with words 36 bits long. Such a word is called a "full-length word", or just "word". A half-word is always treated as the left half of a full word whose right half is all zeros. Most of the 701 system handles each number (not instruction) as a single, full-length, parallel word.

1.2.2. Addresses.

The address of each instruction refers to the location of a half-word. If the sign digit of the instruction is positive, then the addressed half-word is referred to. If the sign digit of the instruction is negative, then the full-length word referred to is the one made up of the two half-words whose addresses are obtained by making the units digit of the specified address first zero and then one.

1.2.3. Storage Tube Layout vs. Addresses.

The Williams tubes are laid out much like the MIT Electrostatic storage tubes. Each shielded box is called a "drawer", and contains two ES tubes (one from each bank) and associated logical circuitry. Each drawer represents one digit of a full-length word, so there are 36 drawers. (See Figure 2.) Each tube holds 512 spots (dots or dashes). There are two banks, so memory contains 1024 full-length words, each one with an even-numbered address. There are consequently 2048 half-length words, numbered consecutively. An eleven-bit address is necessary, but a twelve-bit

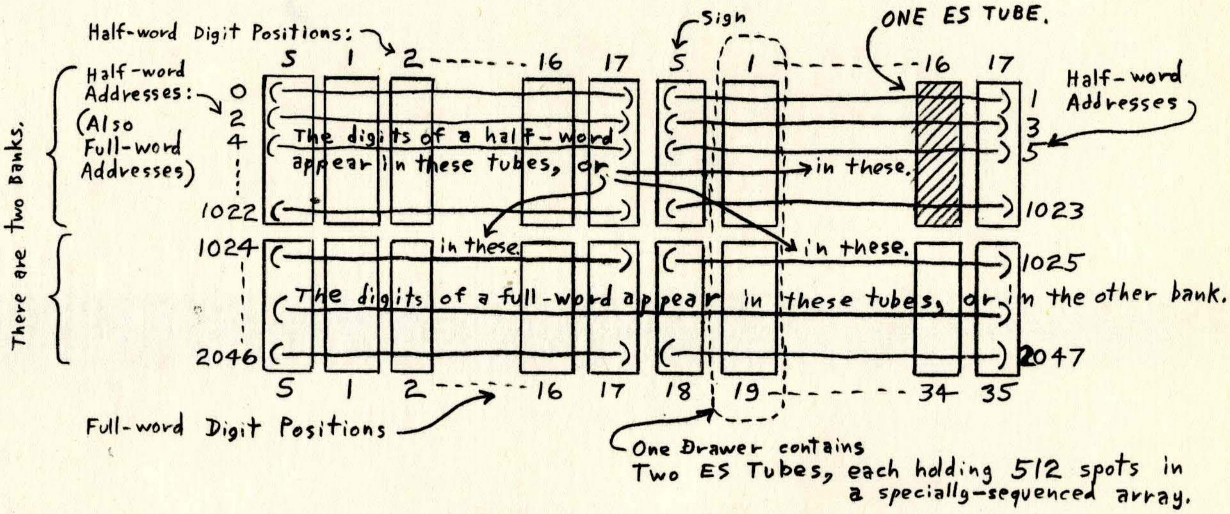


Figure 2. ES Addresses.

address is provided so that an additional storage block can be plugged in.

1.2.4. Signs and Negative Numbers.

The sign digit of a number does not enter into arithmetic operations, but is manipulated independently, as discussed in sections 3.2.3 and 3.2.7. The physical location of the sign digit with respect to other digits has no significance. Thus a full-length number is said to contain 35 bits plus sign; a half-word, 17 plus sign.

1.2.4.1. Interpretation.

The numerical digits are always of positive magnitude, and the sign digit indicates whether the whole number should be negative or positive. Thus, the blank part of a half-word is represented by zeros regardless of whether the number is positive or negative.

1.2.4.2. Location of Sign Digit.

Although the sign position is not associated with any digit position of a number, it must be stored in some position in storage and other registers. This position is, as in WWI, at the left end of the number. Also, as mentioned above, the sign digit does not enter into the numerical part of the calculation. If it is desired to manipulate on the sign digit numerically, it can be stored in an odd-numbered half-register, as shown in Figure 2. It can then be brought into ACC as the middle digit of a full-length number.

1.3. Terminology.

The terminology for the 701 computer is tabulated, and compared with WWI terminology, in Appendix B. An attempt has been made to use 701 terminology throughout most of this note. Where both the 701 and WWI terms are used, the 701 term is written in ALL CAPITALS (which is standard for much of the 701 literature), and the WWI term is placed in [brackets]. Particularly troublesome terms are mentioned below.

AC - The 701 AC is either arithmetic-control or else the ADDRESS COUNTER (which is sometimes called the INSTRUCTION COUNTER or the PROGRAM COUNTER).

The 701 accumulator is called ACC.

Block Diagram - The 701 BLOCK DIAGRAM is like the [block schematic].

The 701 has no [block diagram].

Delay Unit - The 701 DELAY UNIT is also called the HAVENS UNIT, and is described in section 1.4.

The 701 has no [delay element].

Memory Register - The 701 MEMORY REGISTER is like the [PR] and [AR] because it acts as a buffer out of storage (but not into storage) and adds directly into ACC.

The 701 [memory register] is called a MEMORY LOCATION.

Reset - The 701 RESET is like [clear], but sometimes is like [set].

The 701 [reset] (to some number, α) is called SET.

Switch - The 701 SWITCH is like a set of /read in gates/.

The 701 /switch/ is called a MATRIX, or DECODER.

Transfer - The 701 TRANSFER is like /sp/ or /cp/.

The 701 /transfer/ is called STORE, or COPY. (The COPY instruction is like /rd or rc/.)

Trigger - The 701 TRIGGER is like /FF/, and is abbreviated "T".

The 701 /trigger/ is also called "trigger" (verb) (in lower case).

1.4. Symbols.

The symbols used in the 701 BLOCK DIAGRAMS are shown (and compared with WWI /block diagram/ symbols) in Appendix C. There is little neeed to follow these symbols when drawing /block diagrams/ of the 701 circuits, and so WWI /block diagram/ symbols will be used in 701 /block diagrams/ except where 701 BLOCK DIAGRAM symbols will give a clearer picture (as in multiple-input AND /gate/ circuits, or negative AND circuits, etc.). Note that the major differences that will result from this technique are: replacing OR circuits with /arrowhead mixers/, and using a /FF/ whose inputs and outputs might be mixed up with respect to those of the corresponding "T".

1.4.1. TRIGGER.

Notice (Appendix C) that the TRIGGER circuit is like the /FF/, but that the lines to the box are considered to behave just as if a circuit schematic were inside the box. Thus, a positive pulse on the righthand side, or a negative pulse on the lefthand side, will cause the righthand tube to conduct and make the righthand outputs "low". The upper output comes directly from the plate, while the side output comes from a voltage divider connected to the plate. The TRIGGER is not labeled with a 0 or a 1, but a convention has been established: if the righthand outputs are "up", the TRIGGER is said to be "on" and is said to contain "1", and vice versa. Notice that the "TWEAKER" is simply a convenient terminal for manually changing a T, and has no logical significance.

1.4.1.1. Read in to a TRIGGER.

A number is usually read into a TRIGGER register by first RESETTING /clearing/ it and then /pulsing the opposite sides/. An exception is the input to the MDR (MEMORY DEFLECTION REGISTER) /ESD/.

One digit of this is shown in Figure 3. The operation of this circuit can be deduced from the symbols of Appendix C. The T is pulsed either negative or positive, depending on whether the input digit is high or low.

1.4.2. DELAY UNIT (HAVENS DELAY).

Notice (Appendix C) that the DELAY UNIT is the only kind of delay unit used in the 701, and that its output is a "level" which changes to the value applied at the input only at the time that the clamp signal appears. This time coincides with the end of the sync gate, which has admitted the input level. Thus the output of such a unit does not change until just as the input becomes no longer significant.

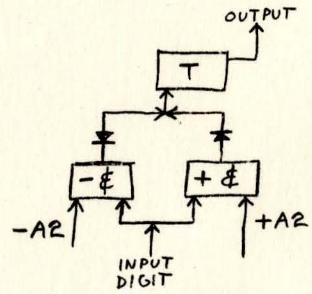


Figure 3. Input to MDR.

1.4.2.1. Read in to a DELAY UNIT.

A typical DELAY UNIT SWITCHING circuit is shown in Figure 4. Notice that the unit "looks at its own tail" when no control signal is applied, and therefore acts like a [dynamic FF]. If a control signal makes it "look at" some other unit, then it is prevented from looking at its own tail. It then assumes the value of the specified input number, whenever a clamp signal (not shown) appears.

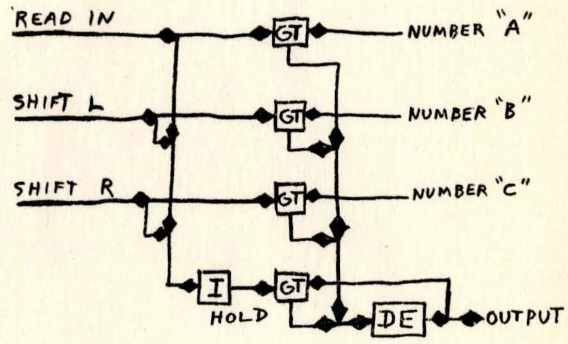


Figure 4. Input to DU.

2. General Logical Design.

Appendix D lists the 701 order code and compares each operation with the equivalent WWI operation, with only a short note to explain the difference. It is hoped that anyone familiar with WWI operation will get a reasonably accurate picture of 701 operations by looking at Appendix D. It may be necessary to refer to sections 2.1 and 2.2 (on Block Outline, and Timing) in order to understand the operations.

2.1 Block Outline.

The over-all logical organization of the 701, shown in figure 5, is very much like WWI. The major differences are as follows:

Stored spots are not [held] all at once, but a group of spots is REGENERATED whenever possible. The REG. COUNTER keeps track of which spots must be regenerated next. Each regeneration regenerates a spot in every tube (i.e., four addresses at once). Thus, it is necessary to include (in the memory drawers) a TRIGGER for each storage tube. These T's form two registers, each much like our [PR]. But, as they play no part in the logic of the machine, they have no special name.

The MEM. REG. acts like [PR] when reading out of storage, except that addresses also go through INSTR. REG. before going to MEM. DEFL. The MEM. REG. also acts like [AR] when doing arithmetic. After a half-length number is placed in MEM. REG. (see section 1.2.1., and note on figure 5) it is handled as if it were a full-length number.

MEM. REG., ACC, and MQ use DELAY UNITS. A "D" cannot be complemented directly, so ACC is complemented by running its contents through the complement part of the TRUE/COMPLEMENT (T/C) circuit and the adder (adding in Zero). (See section 3.2 for arithmetic details).

The MQ (Multiply/Quotient) is most like [IOR], but is also used for [BR]. While tape is using [IOR], care must be taken to prevent any other use of MQ.

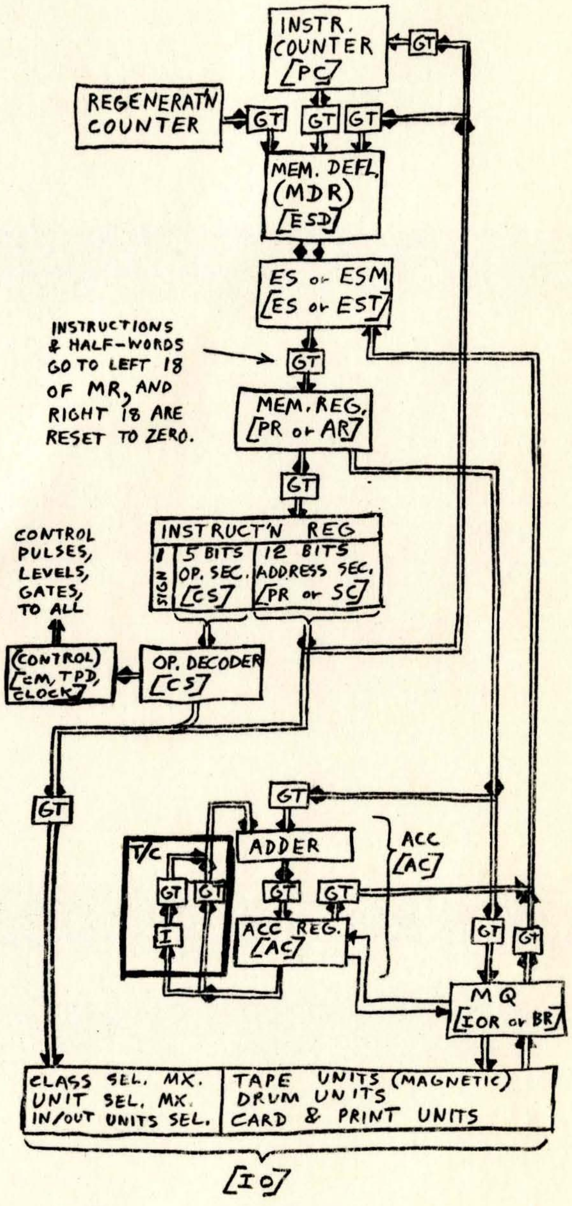


Figure 5.

701 BLOCK OUTLINE

2.2 Timing Outline.

A general outline of the timing for ADD is shown in figure 6. The general outline for other instructions may have more cycles or less. There are two kinds of signals sent throughout the computer to indicate the point in the timing of a single instruction: time pulses (0 through 11) and CYCLES (I, E, E/R, R). Each cycle refers to a complete processing of one word of storage. (Each R or E/R refers to four addresses. See section 2.1.) Notice that any part of [operation timing] which does not need to use storage allows storage to regenerate, and so is made an E/R cycle. The R cycles do nothing but regenerate storage, and can be omitted if storage is in good shape, as discussed below. If they can not be omitted, an ADD takes 60μs, as shown. (Regeneration is "safe", and the programmer never needs to think about it.)

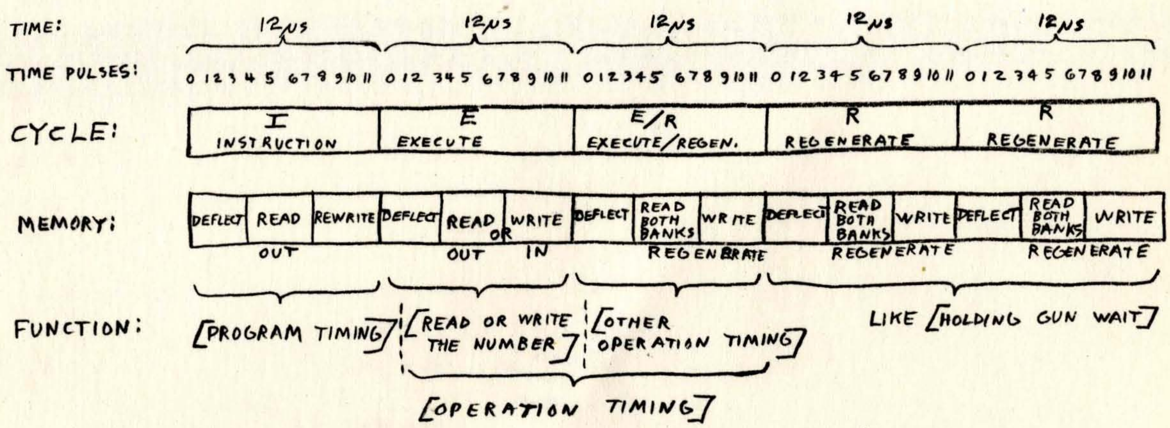


FIGURE 6. Outline of a typical ADD instruction.

2.2.1 "Free games" technique.

One E/R cycle is used for each step of multiply or divide. Thus, one of these operations regenerates a great deal of storage, and it is safe to omit the R cycles on the next twelve instructions of a program. Whenever a MULTIPLY, MULTIPLY-ROUND, or DIVIDE is performed, a counter starts counting instructions as they are performed and allows twelve to occur without any R cycles. This counter is popularly called the "free games" counter. A single multiply or divide takes quite a long time, but if it is followed by twelve ADD's (for instance) then the apparent multiply or divide time is about as short as a single ADD time.

2.2.2. Traffic Outline.

Figure 7 shows a rough outline of the flow of information from register to register during an ADD instruction. The vertical axis shows the registers laid out as in Figure 5. The horizontal axis is "time", as in Figure 6. A dotted line shows the influence of MEM. DEFL. on ES. At the "end of operation", the I cycle line comes on, but is suppressed until R cycles are completed. The two inputs to the adder are actually applied throughout the transient period and until the result is read to ACC.

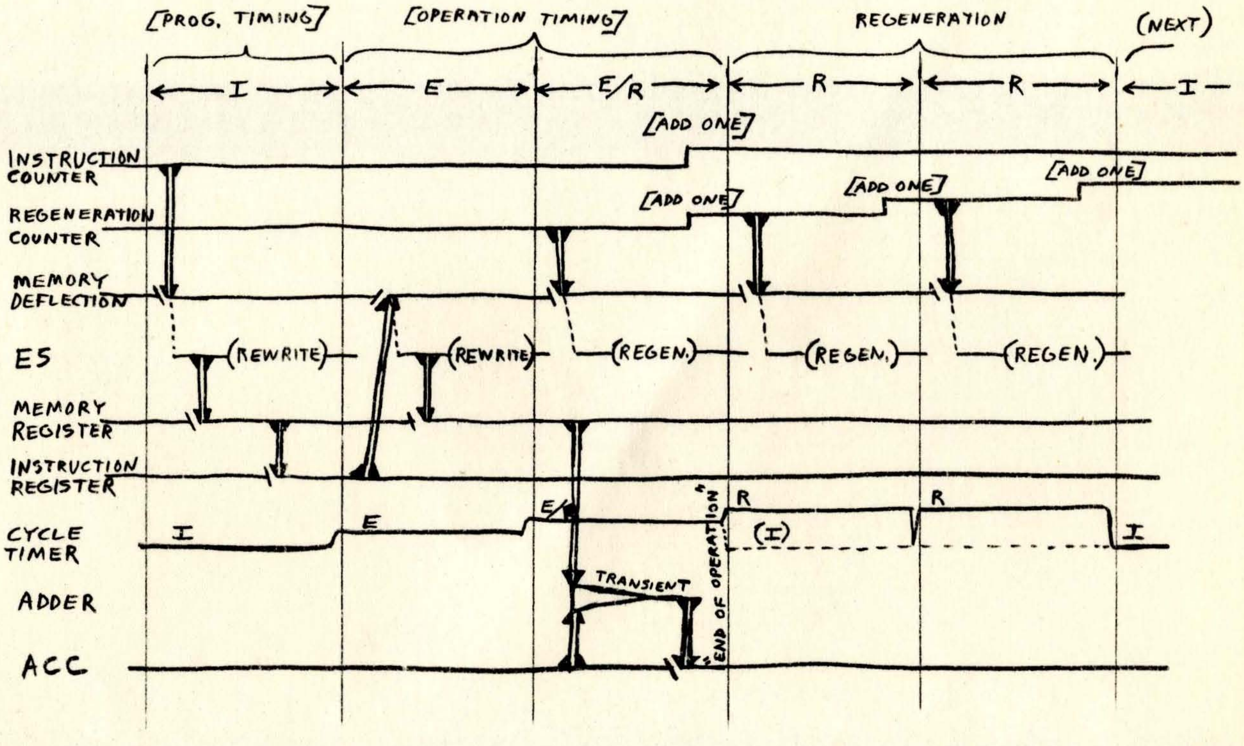


Figure 7. Traffic Outline for a Typical ADD.

3. Some Details.

The following sections provide a little more detail on some aspects of the 701 system.

3.1. Control.

The general outline of [central control] is shown in Figure 8. A clock distributes 12 time pulses. A cycle timer determines the kind of cycle, with the regeneration control telling whether an R is needed or not. The INSTRUCTION REGISTER [CS] tells what operation to perform. These three kinds of signals are combined with other conditions within the machine to provide control signals.

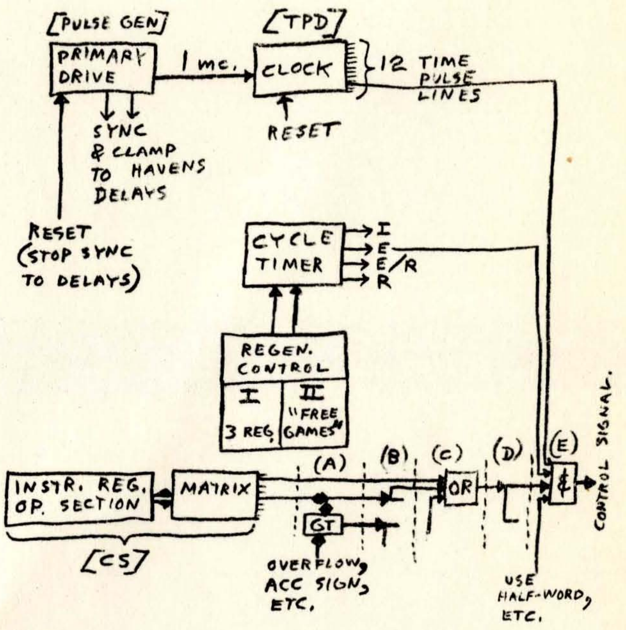


Figure 8. Outline of Control.

3.1.1. Primary Drive and CLOCK.

The primary drive [pulse generator] supplies 1mc.pulses to the CLOCK [TPD]. It also supplies sync and clamp pulses to the delay units, which can be [cleared] RESET by stopping the sync pulse.

The CLOCK never stops, because regeneration must take place if nothing else. It contains 12 T's, only one of which is ON at a time. When it goes OFF it forces the next one ON. The odds and evens are alternately pulsed OFF by a 13th T. The ON consequently progresses around the ring. The CLOCK is "stopped" if more or less than one T is ON. It is "reset" by holding the number zero T ON until all the others are OFF, and then letting it go.

3.1.2. Regeneration Control.

The regeneration control is sketched in the bottom half of Figure 9. The counter on the left side counts to make sure that each instruction has three regenerations, and if so, "R-completed" line is ON, allowing the cycle timer to go to the I cycle. If the R cycles are not completed, the "R-required" line is ON, keeping the cycle timer in the R cycle. The counter on the right is the "free games" counter, which is RESET [cleared] at the beginning of operations like "multiply". It counts once on every instruction until 13 instructions have been started, and during this time the "R-completed" line is turned ON, indicating that no R is required at the ends of the twelve intervening instructions.

3.1.3. Cycle Timer.

Part of the cycle timer is sketched in the top half of Figure 9. It has four T's. At the end of an operation (see also Figure 7) the I TRIG is turned ON. But if an R is required, the R TRIG is also turned ON, and this suppresses the I output line. As soon as enough R cycles have been completed, the R TRIG is turned OFF, allowing the I line to come ON. This same sort of arrangement is used (not shown) when the computer is operated in a manual fashion, so that the R line is on when you have manually stopped the computer, and yet the other TRIGGERS remember what cycle the instruction is in when you want more pulses in the computer.

A negative AND gate (not shown) in each of the top three TRIG inputs allows a negative pulse to turn OFF the TRIGGERS which are not being turned on. Thus, the setting of one TRIG /clears/ the others (but the R TRIG is special, as described above).

3.1.4. Control Signals.

The general method of obtaining control signals was mentioned in section 3.1, and sketched in Figure 8. Notice that the output lines from the MATRIX /cs/ do not go to a /control matrix/ but to a system of AND and OR gates, called EXECUTION CONTROLS and EXECUTION CONTROL MIXERS. Roughly five sections to these circuits can be recognized: (A) a MATRIX line is combined, if necessary, with other conditions in the computer to find out what kind of instruction is called for. (B) The resultant signal is sent to the various OR circuits which control the kinds of /commands/ required.

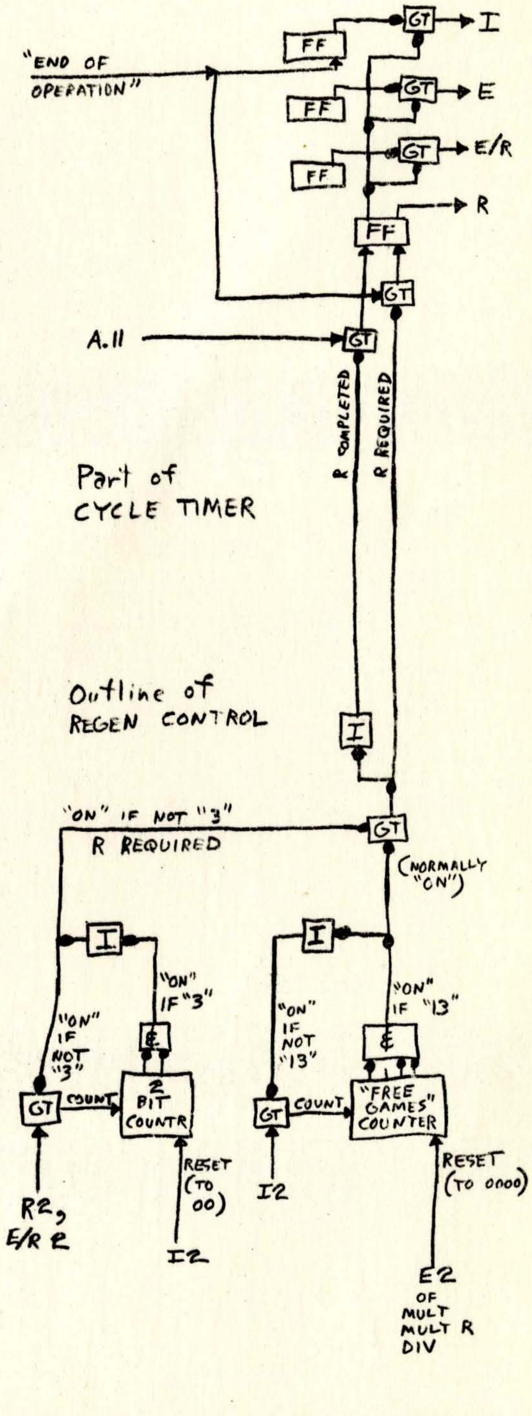


Figure 9. Simplified Sketches of CYCLE TIMER & REGEN. CONTROL.

(C) The OR circuits activate these [commands] regardless of the instruction which calls for them (Sections B and C are roughly equivalent to the inputs and outputs of the [control matrix]).
(D) The signal is then split up, if necessary, into several AND circuits in order to combine with other conditions in the computer.
(E) A control signal appears if its AND circuit finds that conditions, cycle, and time pulse are correct (this section is roughly equivalent to the [cpo units]).

3.2. Arithmetic.

Arithmetic in the 701 is much like that in WWI except for the method of handling overflow and signs, and except for the adder circuits.

3.2.1. Digits in ACC.

The ACC has two digits which never appear anywhere else. (See Figure 10). They are called P and Q (pints and quarts) and represent the digits 2^0 and 2^{+1}. These digits are added and shifted along with the remaining digits of the number so that double-length numbers can be added simply by letting the overflow go into the P and Q digits and then later shifting these digits to the least significant end of the ACC and adding in the most significant part of the double-length number.

The binary point of the 701, as with WWI, can be considered anywhere for addition and either at the extreme right or at the left (between P and 1) of single-length multiplicands and their double-length product. An attempt is made not to think of any particular location for the binary point of the 701 system, but when confusion must be avoided the same convention as WWI is usually used.

TITLE:	S	Q	P	1	2	··· 35
VALUE:	(SIGN)	2^1	2^0	2^{-1}	2^{-2}	2^{-35}
		IN ACC ONLY				

Figure 10. Digits of ACC.

3.2.2. Overflow.

Whenever an overflow occurs into the P digit (when adding, rounding, or shifting left) an overflow T is turned ON. This T remains ON (lighting a light on the operator's panel, but not influencing the operation of the computer) until a TRANSFER OVERFLOW instruction is performed, which senses the overflow, TRANSFERS CONTROL if necessary, and RESETS the overflow T.

3.2.3. Sign Control for Add.

Because of the method of representing negative numbers (see section 1.2.4.1), a subtract instruction is the same as an add except that the sign digit (but not the number digits) is considered to be complemented. The adding and sign-handling circuits are (with the above slight exception) not concerned with whether a positive number is being subtracted or a negative one is being added. Thus, the analysis of ADD describes SUBTRACT as well.

3.2.3.1. The Simple Cases.

If the two numbers being added have the same sign, we wish to let one magnitude increase the other, and we know the sign of the result must be the same as that of the original numbers. Thus the two numbers are simply added and the ACC sign is not touched.

3.2.3.2. The Other Cases.*

If the two numbers being added have unlike signs, we wish to let one magnitude decrease the other. We must then decide what the resultant sign should be. The result will have the same sign as the original ACC if the magnitude of ACC is large enough to nullify the magnitude of MR.[AR] (See Figure 11.)

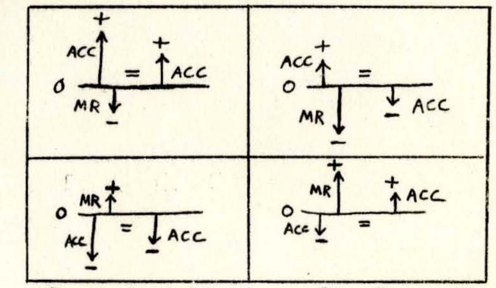


Figure 11. ACC Sign after "subtract".

Since a magnitude smaller than zero can not be represented directly, the decreasing of magnitudes can not be done unless one of the numbers is complemented. As in WWI, the 9's complement is used, thus letting "111111---111" represent "0" [minus zero] (or [-0]). (Notice, however, that ACC sign is not involved in this complementing.) Therefore, the complement of the magnitude of ACC is added (by way of the T/C circuit of Figure 5) to the magnitude of MR. This accomplishes the decreasing of one magnitude by the other with respect to [-0], as shown in Figure 12. (The brackets show the relationship between the ACC magnitude and its negative-magnitude-with-respect-to [-0], and show also the resultant magnitude-with-respect-to [-0].) The magnitude

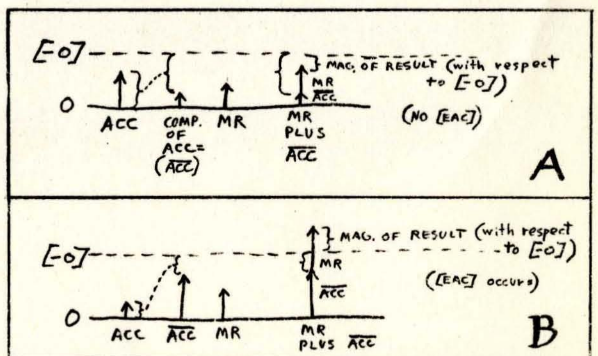


Figure 12. Magnitudes with respect to [-0]

*An explanation not used by IBM, but presumably giving the correct results.

must then be returned to its correct form. In Figure 12B, the magnitude is above [-0], so an [EAC] (end around carry) (as in the WWI system) has occurred, and the magnitude is in correct form. In Figure 12A, however, no [EAC] has occurred, so the ACC must be complemented again (by sending it alone through the T/C and adder of Figure 5) to arrive at the correct magnitude.

So far, our addition of unlike signs has produced the correct magnitude. (An overflow can not arithmetically occur, so it is not sensed.) All that remains is to find the correct ACC sign. As in Figure 11, the sign digit of ACC must not be changed if the magnitude of ACC is large enough to nullify the size of MR. This occurs in the case of Figure 12A. Thus, if no end around carry occurs, the ACC sign digit is not changed, and vice versa as in Figure 12B.

As in WWI, a result of "zero" produces no end around carry. But unlike WWI, the above rule says that in this case ACC sign is not changed, but the magnitude is complemented. So the sign of a "zero" remainder is always the same as the original sign of ACC, and the number digits are all zero.

3.2.3.3. Rule for Sign Control on ADD.

Signs same: add magnitudes; leave sign alone; set overflow T if necessary.

Signs different: add complement of ACC magnitude to magnitude of MR; forget overflow.

Then: - - if no [EAC]: ($|ACC| \geq |MR|$; Figure 12A); complement ACC, but not ACC sign.

- - if [EAC]: ($|ACC| < |MR|$; Figure 12B); complement ACC sign, but not ACC.

3.2.4. Adder.

The circuit for the passive adder is shown in Figure 13. Only one digit is shown. A "sum" output appears if all three of the inputs are ON, or if any of them is ON and there is no "carry out". The "carry out" is turned ON if any two or more of the three inputs are ON. Thus, when the input gates are turned ON for 6μsec. the circuit outputs eventually stabilize. It takes the carry levels about 3.5 μsec. to become stabilized from one end of the complete adder to the other. Thus a safety margin is included, since the pulse "sum to ACC" occurs 5 μsec. after the start of the read-in gate. The sum is then fed to the ACC DELAY UNITS,

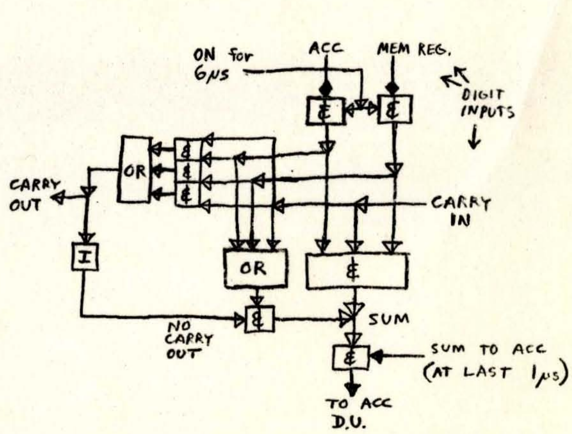


Figure 13. ADDER DIGIT

and they eventually become set to the correct sum.

3.2.5. Shift Counter.

The ADDRESS SECTION OF THE INSTRUCTION REGISTER receives the address section of all instructions and is therefore used as the [step counter] for shifting and also for multiply and divide. A shift of up to 255 can be programmed, and any shift over 72 will clear both ACC and M/Q. The reason for allowing such large numbers of shifts is that a calculated scale factor can be inserted as the address of a shift instruction with some assurance that large numbers of shifts will clear the register instead of simply producing small numbers of shifts (i. e., a shift of 128 would result in a shift of 0 if this feature were not provided).

3.2.6. Shifting.

Shifting is accomplished by making each DELAY UNIT of ACC (including P and Q) and M/Q "look at" its neighbor instead of "its own tail" (see section 1.4.2.1.). A shift of one place per μsec. occurs as long as they look at their neighbors, so the shift line is merely turned ON for the length of time determined by the [SC] Actually, no more than 8 shifts occur on any E/R cycle. See Appendix D for long and short shifts, and sign control for shifts.

3.2.7. Sign Control for Multiply and Divide.

The sign control for multiply and divide is similar to that of WWI except that the numbers themselves are already in positive magnitude form, and except for the fact that both ACC and MQ have sign digits which must be correct. (The "divide" leaves a quotient in MQ with its sign, and the remainder in ACC with its sign. See section 3.2.8.)

3.2.8. Dividing.

In the "divide" operation, the adder is sensed to discover whether a subtraction was too much, and if so, the sum is simply not copied into ACC. Thus, ACC always contains a regular remainder, rather than the plus-or-minus remainder found in WWI, and no corrective add is required.

3.3 In-Out.

As with WWI, the in-out system has so many variables that it is not as easy to describe as the rest of the system. The following notes, therefore, are even less complete than the preceding notes.

To operate an in-out unit, the general procedure is to use an instruction READ or WRITE, like [si], followed by any number of COPY instructions. The [si] used selects the type, mode, and unit to use, while each COPY actually copies a word into or out of ES depending on whether READ or WRITE was used as [si]. If a COPY is not programmed in time, the in-out unit in use at the moment usually simply disconnects itself from the computer and eventually stops. If a COPY occurs too early in the program, the computer simply waits. See Appendix D.

3.3.1. Card Characteristics.

Standard IBM cards can be punched in binary fashion under a program of control from the computer. When punched in this way, each 80-bit row of the card will contain two full-length words, or 72 bits. There are 24 full-length words, or 48 instructions, on each card. These cards read into ES under the control of a program in much the same way that paper tape reads into WWI. Standard alpha-numeric cards can also be read in by way of a special program.

3.3.1.1. Card Read-in Program.

A special loading card can be read in by means of three simple instructions programmed by pushing the LOAD BUTTON. One of these instructions is a COPY, which brings in the first word (2 instructions) from the card. One of these new instructions brings in another word, and so the card pulls itself into ES by its own boot straps. The remainder of this loading card has on it a program much like our read-in programs, and allows reading-in the remainder of the cards in a deck. The remaining cards of the deck each have a check number, a starting address, and a number indicating how many registers should be loaded from the card. The last card in the deck tells where to start the program that has just been read in, and is called the TRANSFER card. The cards between the load and the TRANSFER card can be in any sequence, since each has its own starting address.

3.3.1.2. Reader, Punch, and Printer.

The reader can read cards at a maximum of 150 cards/minute.
The punch can punch cards at a maximum of 100 cards/minute.
The printer can follow cards at a maximum of 150 cards/minute, and prints one line for each card. (It cannot follow binary-punched cards.)

In order to operate the punch and printer, these devices must receive information as if it were coming from a standard IBM card. This usually means that the information has to be calculated beforehand, and placed in storage as a "card image". This card image can then be copied half-row by half-row into the punch or printer.

3.3.2. Drum.

The following paragraph is to be considered HIGHLY COMMERCIALLY CONFIDENTIAL.

The drum circuit involves the use of a single counter, and the address of the desired drum register is placed in this counter by a special instruction. This counter does not count until the specified drum passes the zero mark. The counter then counts down until it is empty, at which time the drum is at the proper address. Thus it takes on the average half a revolution to start counting and half a revolution to reach the specified address, so that the average time is one complete drum revolution to find an address. The average time will be only half a revolution if addresses near the zero mark are specified, and these addresses are multiples of 32. There are four "drums" contained on two separate cylinders.

3.3.3. Tape (magnetic).

The magnetic tape contains blocks of information, of any length, with blank spaces between them. A single COPY instruction pulls in a full word length, which involves six lines, each six bits long. When reading, the tape will stop only at blank spaces. The program has full control of what is recorded where, and so must manufacture its own addressing system if one is desired. The tape moves at 75 inches/second, and contains 100 bits/linear inch.

4. Further Information.

Some further information can be obtained from publications in the Whittemore Building Library: Library number 2077, "Defense Calculator Memos", contains memos on the 701, only some of which are concerned with logical design (see list in Appendix E). Library number 2076, "701 Operator's Reference Manual", is a preliminary manual which is largely obsolete. However, the descriptions of operations (beginning in page 54) should be up to date except for the description of the "read backward" (and of COPY after "read backward"). Pages 47 to 51 might also be of interest, though possibly out of date.

Appendix A

Logical Characteristics of 701 System (see section 1.2.)

		(Footnotes)
Type	General-purpose, high-speed.	
Design	Electronic, digital.	
Number system	Binary.	1.
Register length (basic, or full word) (basic instruction, or $\frac{1}{2}$-word)	36 Binary digits. 18 Binary digits.	2.
Method of handling numbers	Parallel digital transmission, addition, and storage.	
Type of internal storage	Williams electrostatic memory tubes. (2 banks, 512 spots per tube.)	3.
Capacity of internal storage	2048 half-words, numbered consecutively.	4.
Access time to internal storage	12 μ sec. read-rewrite.	5.
Basic functional design	Pulses (usually one μ sec. long) and outputs from basic memory devices are switched via crystal circuits to determine new states for the memory devices.	6.
Basic memory devices	TRIGGER (like WWI FF), and HAVENS DELAY unit.	6.
Crystal switching circuits	AND circuits (like gates), and OR circuits (like mixers).	6.

1. See section 1.2.4.1.
2. See section 1.2.1.
3. See section 1.2.3.
4. See section 1.2.2.
5. See section 2.2.
6. See section 1.4.

Appendix B

Terminology (see section 1.3.)

Note: 701 NAMES ARE IN CAPITALS, and [WWI names are in brackets].
Abbreviations are listed at the end.
(Some names are popular, but not official.)

ADDRESS COUNTER = INSTRUCTION COUNTER = [PC]
BLOCK DIAGRAM = [block schematic] = SYSTEM DRAWING = LOGICAL BLOCK DIAGRAM
[block diagram] = 701 HAS NONE
DELAY UNIT = HAVENS UNIT = [see sections 1.4. and 1.4.2.]
[delay element] = 701 HAS NONE
GLITCH = [a botherless negative spike on an ON line]
MEMORY REGISTER = [PR or AR] (see sections 1.3. and 2.1.)
[memory register] = MEMORY LOCATION
MULTIPLY/QUOTIENT = [IOR or BR] (see section 2.1.)
REGENERATE = [hold spots in PS, one spot in each tube at a time]
RESET = [clear] or sometimes [set]
[reset] = SET (to some number, x)
SLIVER = [a bothersome spike]
SPIKE = [spike: botherless positive noise on an OFF line]
[step counter] = 701 USES PART OF ADDRESS REGISTER [PR].
SWITCH = [rather like read-in gates] (see section 1.4.2.1.), or [diode circuit]
[switch] = MATRIX or DECODER
TRANSFER = [cp] or [sp]
[transfer] = STORE or COPY
TRIGGER = [FF] or [trigger] (verb)
TRUE NUMBER = [a number in "magnitude-with-sign" form]
TWEAKER = [a convenient terminal for "screwdriver" setting of a FF]
= MANUAL "OFF" (or "ON") PIN CONNECTION.
UNIT RECORD = CARD, or A BLOCK ON TAPE

Some abbreviations:

A = ALL CYCLES, or AMPLIFIER
AC = ARITHMETIC & CONTROL UNIT, or ADDRESS COUNTER
ACC = ACCUMULATOR
D = DELAY UNIT, or DURATION OF SIGNAL (as in "A2(D1)" the duration is 1 μ sec.)
E = "EXECUTE" CYCLE
E/R = "EXECUTE/REGENERATE" CYCLE
I = INVERTER, or "INSTRUCTION" CYCLE
MQ = M/Q = MULTIPLIER/QUOTIENT REGISTER [IOR or BR]
MR = MEMORY REGISTER [PR or AR]
OR = OR circuit [mixer]
R = "REGENERATE" CYCLE
T = TRIGGER
T/C = TRUE/COMPLEMENT CIRCUIT

¢ = & = AND circuit [gate]

Appendix C

Symbols (see section 1.4.)

Time notation:

Example: + E 3 (D 2)
- (D 2) → Signal has duration of 2 μs.
- 3 → It starts with time pulse 3
- E → CYCLE (I, E, E/R, R. A = ALL CYCLES.)
- + → Polarity at time shown (nothing = plus).

(An input to a T which is not labelled with the time is usually just a TWEAKER, unless it comes from some other circuit.)

Other notations on lines are pin numbers, [System numbers] PAGE NUMBERS, etc.

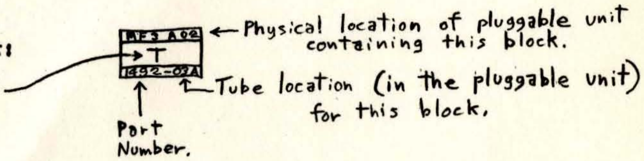
An attempt is made to have information flow from left to right.
So counters count from left to right.

Signal voltages are standard:

+10 (clamped) = ON = UP = HOT = PULLED

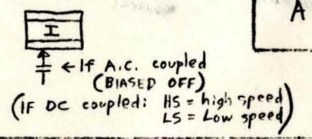
from -30 to -15 = OFF = DOWN

Each standard circuit is shown by a box:



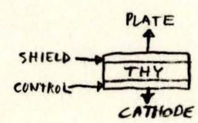
Abbreviations for kinds of circuits:

I = Inverters (fast)
- I_s = slow
- I_{ps} = pullover
- Pkr = Peaker
- SI = sync
- CI = clamp

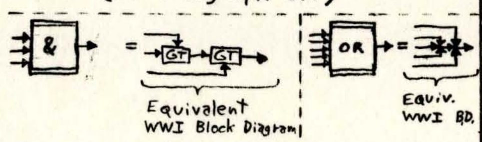


CF = Cathode Follower
PCF = Power CF
A = grounded grid Amplifier

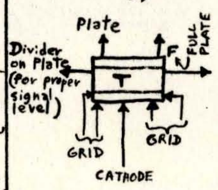
THY = Thyratron



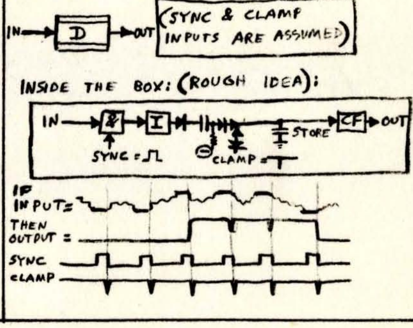
₵ = & = AND circuit [Gate]
OR = OR circuit [Mixer]
-₵ = -& = OR circuit operated with negative signals (Output OFF if all inputs OFF)
-OR = AND circuit with neg signals (OFF if any input OFF)



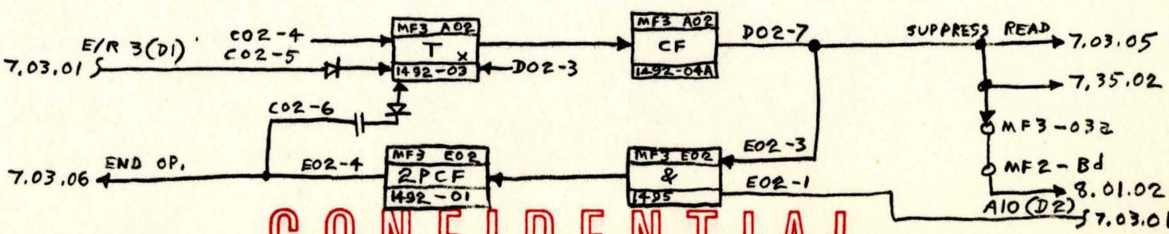
T = TRIGGER [FF]
T_s = Slow T
K_T = Key TRIGGER (controlled by key)



D = DELAY (HAVENS UNIT)



SAMPLE LOGICAL BLOCK DIAGRAM:



Appendix D - Order Code (see section 2.)

Deci, #	Short Name	Symbol	Closest WWI Equivalent	Explanatory notes (see section 1.2.2.)
0	STOP & TR	STOP	sp/rs, stop	(Old & NEW [PC] address is displayed while stopped).
1	TRANSFER	TR	sp	(old [PC] address is not remembered in [AR]).
2	TR OVERFLOW	TR OV	(cp on overflow)	(This is the only way to sense or to [clear] the overflow TRIGGER.).
3	TR ON PLUS	TR +	(cp +)	
4	TR ON ZERO	TR O	(cp O)	
5	SUBTRACT	SUB	su	ZERO RESULT HAS SAME SIGN AS PREVIOUSLY IN ACC. (5–11)
6	RESET & SUB	R SUB	cs	(RESETS [clears] P & Q DIGITS OF ACC).
7	SUB ABSOLUTE	SUB AB	sm (su magnitude)	
8	NO OPERATION	NO OP	rs, no stop	OVERFLOW SIMPLY GOES INTO P & Q AS IF THEY WERE REGULAR DIGITS, BUT ALSO SETS OVERFLOW TRIGGER, WHICH can be cleared only by TR OV. (5–11)
9	ADD	ADD	ad	
10	RESET & ADD	R ADD	ca	(RESETS P & Q).
11	ADD ABSOLUTE	ADD AB	am	(ad magnitude).
12	STORE	STORE	ts	
13	STORE ADDRESS	STORE A	td	(12 digits).
14	STORE MQ	STORE MQ	(ts from BR)	
15	LOAD MQ	LOAD MQ	(ca direct to BR)	(No change in ACC).
16	MULTIPLY	MPY	mh	MUST HAVE NUMBER IN MQ (by LOAD MQ), NOT IN ACC, which is lost. PRODUCT SIGN GOES TO BOTH ACC SIGN & MQ SIGN. (16–17)
17	MPY ROUND	MPY R	mr	(NO [clear BR]).
18	DIVIDE	DIV	dv	SIGN OF MQ IGNORED, ASSUMED SAME AS ACC SIGN. COMBINED, DOUBLE-LENGTH ACC→MQ divided by MR. QUOTIENT IN MQ; REMAINDER IN ACC with previous ACC sign. (see section 3.2.8.)
19	ROUND	ROUND	srr O	THIS DOES NOTHING BUT ROUND-OFF. IT DOES NOT CLEAR MQ. SIGN & ADDRESS OF ROUND INSTR. CAN BE ANYTHING.
20	LONG S L	L LEFT	slh	ACC SIGN IS MADE SAME AS MQ SIGN. P, Q, ACC, MQ ARE SHIFTED. (20–21)
21	LONG S R	L RIGHT	srh	MQ " " " " " ACC ". OV TRIG IS SET IF "1" GOES FROM DIGIT 1 TO DIGIT P. (20–23)
22	ACC S L	A LEFT	slr	ACC, P, & Q ARE SHIFTED, BUT NOT MQ. LIKE LONG SHIFTS EXCEPT MQ NOT TOUCHED, AND SIGNS NOT TOUCHED. (i.e. OV TRIG) (22–23)
23	ACC S R	A RIGHT	srr	(NO ROUND-OFF)(NO MQ [CLEAR]).
24	PREP TO RD	READ	(si read)	USE WITH PRINTER FOR ECHO [FEEDBACK] CHECK, AND ALL INPUTS EXCEPT READ B. (SEE SECTION 3.3). SETS IO INTERLOCK, WHICH IS [CLEARED] ONLY WHEN A UNIT DISCONNECTS. IF INTERLOCK IS SET, IT WAITS FOR DISCONNECT. (24–28)
25	PREP RD BKWD	READ B	(si rd. backw.)	
26	PREP TO WRT	WRITE	(si write)	
27	WRT END FILE	WRITE EF	(si blank tape)	CLEARS SECTION OF TAPE IF UNIT IS NOT IN READ MODE.
28	REWIND	REWIND	(si rewind)	USE WRITE EF FIRST.
29	SET DRUM ADR	SET DR	(si drum address)	USE AFTER READ (DRUM) OR WRITE (DRUM). ADDRESS = DRUM STARTING ADDRESS. ZERO, IF SET DR NOT PROGRAMMED.
30	SKIP, CONTROL	SENSE	(si light, or cp on manual)	ADDRESS = LIGHT TO TURN ON, (OR TURN ALL OFF), KEY TO SENSE: ON = SKIP ONE INSTRUCTION OF PROGRAM. (ONE SENSE ADDRESS = WAIT FOR IO INTERLOCK TO GO OFF)
31	COPY & SKIP	COPY	(rd or rc)	SKIP ONE INSTR. IF NO MORE CARDS, ETC. (END OF FILE). SKIP TWO " IF NO MORE WORDS ON CARD, ETC. (END OF RECORD).

APPENDIX E (see section 4.).

List of IBM Memos

Memo Number	Description
*116	Some Symbols
117	Regeneration
(120	Checking)
125	Clock, etc.
131	Regeneration Control
*132	NOMENCLATURE
(138	ACC includes adder and T/C circuit)
(140	"Add-to-Memory" instruction does not exist)
150	Symbols (especially page 6). (See also 157).
*151	Drawing block diagrams. (See also 152 and 158)
(152	See 151)
155	Tape nomenclature
157	(See 150)
*158	Title boxes; labels. (See also 151)
(162	Neon bulbs)
(163	Labels and diode circuits)
183	Order "+ copy"
(184	Octal-decimal manual table)

* - of basic importance.

( ) - of lesser importance.

Appendix F

Index

APPENDICES

Signed Rollin P. Mayer
Rollin P. Mayer

Approved NHT
Norman H. Taylor

RPM/bs