JOHNNIAC PROGRAMMERS' MANUAL

by Wesley S. Melahn

DRAFT April 4, 1955

JOHNNIAC PROGRAMMERS MANUAL

DESCRIPTION OF JOHNNIAC

1.1 Introductory Description

The JOHNNIAC is an electronic digital computer. By means of an internally stored program the computer can perform specified sequences of operations on data stored internally and can control the operation of auxiliary storage and input-output equipment.

The internal storage consists of a magnetic core storage device containing 4096 places where an array of forty binary digits, called a word, can be stored. Each of these places at which a word can be stored is assigned a number which can be used to distinguish it from the other places or locations in the Store. This number is sometimes called the address of the word occupying the specified location. In addition to the internal storage which can be referred to directly by specifying one of the assigned addresses, the JOHNNIAC has 24,576 (initially 12,288) words of auxiliary magnetic drum storage. Words stored in the auxiliary storage cannot be referred to in the address part of an instruction; however the computer can make random access to information stored on the drum and can transfer blocks of data of length 1 word to 2048 (initially 1024) words back and forth between auxiliary storage and internal storage.

The input to the JOHNNIAC is an 80-column card reader, IBM collator, which feeds cards in either of two feeds at 240 cards per minute. Small amounts of information can also be entered from keys and switches on the console. The output is an 80-column, IBM 523, card punch and a 40-column ANelex printer. The punch feeds 100 cards per

minute. In addition to punching cards it can read the previously punched card and transmit this information to internal registers of the computer where it can be used to verify the punching. The printer can print 40 columns of decimal information at approximately 900 lines per minute. No special characters, such as signs or alphabetic symbols, are available on this printer.

The console contains buttons and switches for manual control of the computer, a display unit which can be used to monitor the contents of a selected computer register, and a set of keys for entering numbers into the various registers.

The major parts of the computer and some of the important connections are shown in Figure 1. The JOHNNIAC contains four registers; the Accumulator, the Multiplier-Quotient Register, the Number Register, and the Instruction Register. The Accumulator and Multiplier-Quotient Registers play a vital role in the arithmetic operations of the computer. The Number Register serves as a buffer between the Store and the Control. It holds an instruction while it is executed. The machine's arithmetic unit contains an Adder. All arithmetic operations are accomplished by proper manipulation of clears, complementations, shifts and additions. The functions of the various parts of the machine are explained in more detail in the description of JOHNNIAC operations.

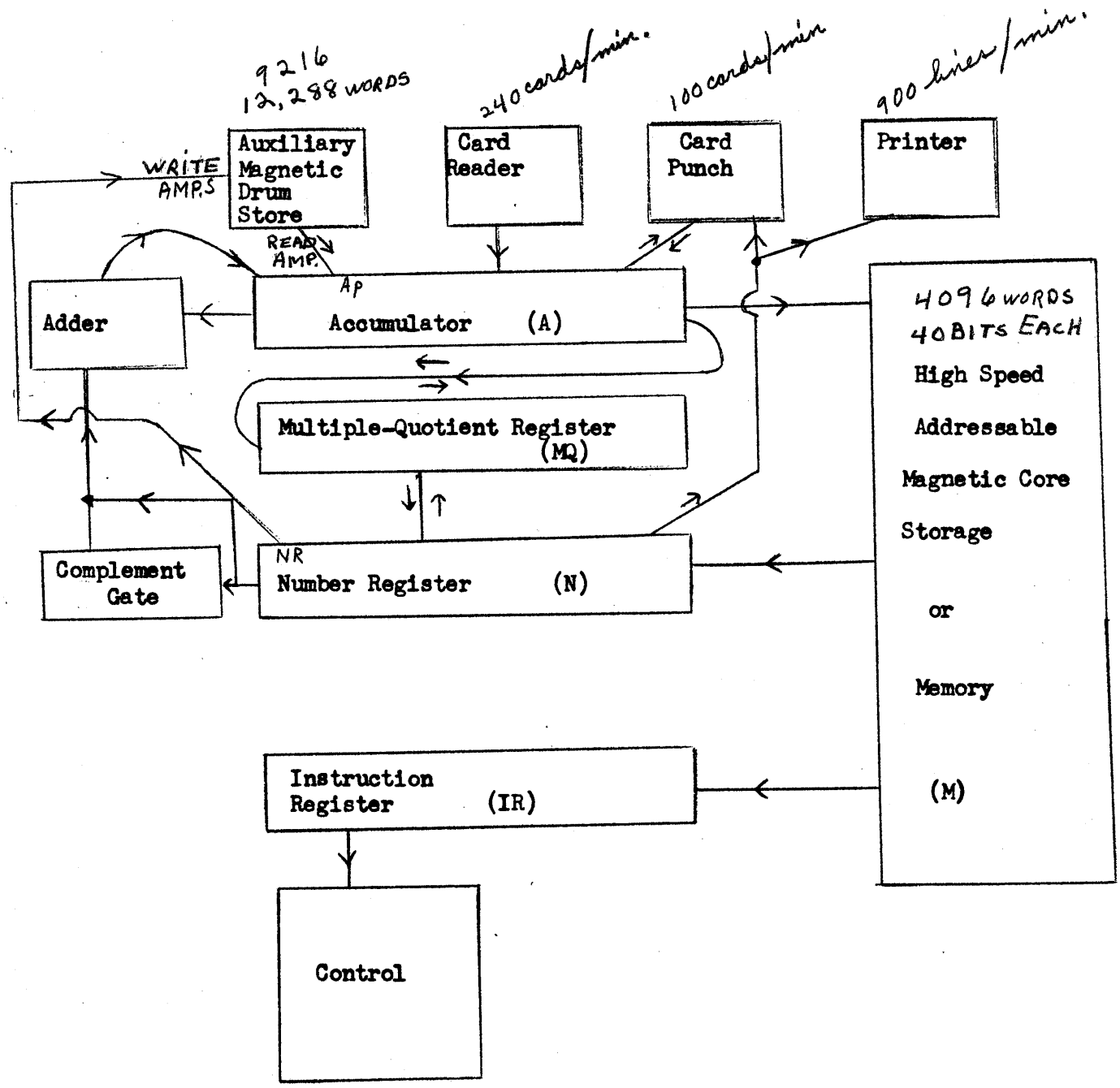


Figure 1. Schematic diagram showing parts of the JOHNNIAC and important connections.

1.2 Interpretations of a Word

Two sets of rules have to be described in order to explain the operation of an internally stored program machine like JOHNNIAC. One set of rules specifies how the control of the machine interprets a word as an instruction. The other set of rules describes how the words are treated as numbers and operated on by the arithmetic unit.

A JOHNNIAC word which is transferred into the Instruction Register for interpretation as an instruction is considered to contain four parts: left ORDER, left address, right ORDER and right address. Most JOHNNIAC ORDERS require one address to clarify the meaning of the ORDER. The left address is associated with the left ORDER, and the right address with the right ORDER. In general, the address specifies a location in the MEMORY which is to supply a word to the arithmetic unit or to the control unit, or which is to receive a word, or part of a word, from the arithmetic unit. The address parts that do not contain numbers identifying a location in the MEMORY usually contain some other information relative to the ORDER; for instance, the identification of an input or output unit or the number of places to shift.

Seven bits are used to specify an ORDER. Of the 128 possible ORDERS, 82 are now assigned. See the list of ORDERS. Twelve bits are assigned to an address part. This implies that the 4096 words in the MEMORY can be uniquely addressed, but that there are no spares. Further, some less direct way than a number in an address part of an instruction must be used to refer to information stored in the auxiliary magnetic drum store. The wiring of the JOHNNIAC Instruction Register assumes that the thirty-eight bits re-

quired by an instruction are located in the forty bit word as shown in the diagram of Figure 2.

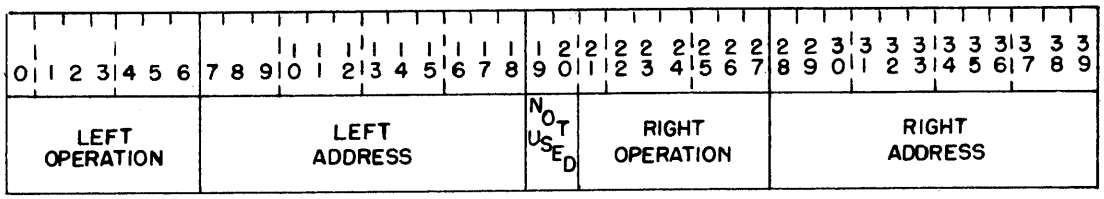


FIGURE 2. JOHNNIAC INSTRUCTION WORD LAYOUT

In the diagram the forty positions are numbered 0 to 39 from left to right. The two unused bits in positions 19 and 20 are ignored in the instruction register.

The interpretation of a JOHNNIAC word as a number is not as convenient to describe as its interpretation as an instruction because many interpretations can be made. One goal in building a machine like JOHNNIAC is to provide ORDERS which are convenient for doing a wide variety of problems. In view of this, it is not surprising to find that the ORDER list contains ORDERS which cannot all be described conveniently with the same interpretation of a word as a number. Or to find that an ORDER can be described equally well with more than one interpretation of the words involved. Perhaps the most common interpretations are those which consider the word to represent a fraction or an integer. In both cases representation for negative as well as positive quantities is desirable.

When the JOHNNIAC word is considered to represent a positive or a negative fraction, the representation in the machine is two plus the fraction, truncated to 39 binals, modulo 2. The binary point is

(CUT OFF SHORT)

considered to be between the first and second bit.

Examples:

$+ 7/8 = (2+7/8)_{\text{Mod } 2} = 7/8 = 0.111\ *$

$- 5/8 = (2-5/8)_{\text{Mod } 2} = 1+3/8 = 1.011*$

* The least significant 36 zeros have been omitted from these words. (This is one of the conveniences obtained by fractional interpretation.)

The fractions which can be represented are the truncated (CUT OFF SHORT) versions of the fractions in the range, plus $(2^{39}-1)/2^{39}$ to minus one. With this interpretation in mind, the Adder which forms the forty bit sum of two forty bit operands is described as a modulo 2 adder. Positive fractions and zero represented this way will have a zero high order digit; negative fractions and minus one will have a one as high order digit. Sometimes, during multiplication for instance, it is convenient to regard the high order digit as a sign digit which is treated separately from the other 39 digits. Then the sign digit is considered to have value zero or minus one. The 39 bits on the right of the point always represent a positive fraction. Positive fractions are represented as zero plus the fraction truncated to 39 binals. Negative fractions are represented by minus one plus the quantity one minus the magnitude of the negative fraction truncated to 39 binals.

Examples:

$+ 7/8 = 0 + 7/8 = 0.111\ *$

$- 5/8 = -1+(1-5/8) = 1.011\ *$

* 36 Zeros omitted.

The numbers handled by the JOHNNIAC can also be considered to be integers with the least significant digit on the right of the 40 bit word. In order to have a convenient representation for negative numbers, the convention is adapted that positive numbers will be restricted to the range zero to $2^{39}-1 = 549, 755, 813, 887$, and negative numbers will be represented by complements with respect to 2^{40}. This provides a representation for negative numbers from minus one to minus 2^{39}. These restrictions on range make it possible to recognize a complement by the one in the high order position of the 40 bit word.

Examples:

$-2 = 2^{40} - 2 = (1, 099, 511, 627, 774)_{10} = (1,111\cdots111,110)_2$.

$+7 = \quad +7 = (\qquad\qquad\qquad\qquad 7)_{10} = (0,000\cdots000,111)_2$.

Just as in the case of fractions, it is often convenient to treat the high order binary digit separately. This high order, or sign digit is given the value zero, or minus 2^{39}. The number on the right of the sign digit is always positive. That is, positive integers are represented as zero plus the integer; and negative integers as minus 2^{39} plus the positive number 2^{39} minus the magnitude of the negative integer represented.

Examples:

The machine representation for -2

equals $\quad -2^{39} + (2^{39}-2)$

equals $\quad -549, 755, 813, 888 + 549, 755, 813, 886$

equals $\quad (1.7,777,777,777, 776.)_8$

$+7 = 0+7 = (0.0,000,000,000,007)_8$

1.3 Description of JOHNNIAC ORDERS

The ORDERS which the JOHNNIAC can perform automatically are limited to those which have been planned for in building the control of the machine. Currently, 83 ORDERS are possible. More complicated ORDERS must be constructed by doing sequences of these basic ORDERS. This has been provided for by the construction of an automatic sequencing device as part of the control.

Some of the ORDERS are concerned with control of the sequencing device. These are called the transfer of control or jump ORDERS, and may be conditional or unconditional ORDERS. If executed, the jump or transfer instruction indicates which word in the MEMORY is to be placed in the Instruction Register and whether the left or right ORDER is to be executed first. If the left ORDER is executed first, the sequencing device executes the left ORDER, the right ORDER, and then fetches the word from the next higher numbered location in the MEMORY and proceeds to execute the left ORDER. This sequence left, right, fetch continues until a halt or a transfer instruction interrupts it. In case the cycle is started by the execution of a right ORDER the sequence goes right, fetch, left, right, fetch, etc. This sequencing of the control includes a memory device which stores the address of the next instruction to fetch. The abbreviation NIA, next instruction address, is used to refer to this counter. The other ORDERS in the list have to do with control of the input-output devices, with the transmission of information between the arithmetic unit and the MEMORY, or with the processing of data in the arithmetic unit. Any of the ORDERS in the list can occur in either the left ORDER part or the right

[redacted] part of an instruction. The left address is associated with the left [redacted]: and the right address with the right [redacted]. The control will hang up; that is, the machine will stop if a number which is not one of the assigned [redacted] turns up in the instruction register.

The 82 [redacted] can be grouped in twelve groups or classes. The functions of the various [redacted] in a group are in general similar. The numbers which designate the [redacted] are chosen so that the high order four bits designate the class and the low order three bits designate the variations. Not all groups contain eight variations, and in a few cases unrelated [redacted] are added to an incomplete group. The description which follows will be made by group. Octal digits will be used to indicate the [redacted] codes. A mnemonic [redacted] code symbol that has been used in the assembly program is also included in the margin opposite the description headings. The binary digit positions in a word will be considered to be numbered 0 to 39 from left to right for purposes of reference.

00X CONDITIONS TRANSFER AND OTHER ORDERS

This is one of the groups which contain orders that are not all closely related.

000 NO OPERATION - PROCEED TO THE NEXT ORDER BLANK

This ORDER causes the computer to go ahead to the next step in the sequencing cycle left, right, fetch, but does no operation on the registers or the MEMORY of the computers. This is a gap filler order and was purposely chosen to have zero for operation code. The address part is ignored.

004 LOAD MULTIPLIER - QUOTIENT REGISTER LM

The word from the MEMORY designated by the address associated with this ORDER is read into the Number Register and transferred to the Multiplier-Quotient register. The Accumulator is undisturbed.

The remaining six ORDERS in the 00 group are three pairs of conditional transfer of control or jump operations. The sequencer, depending on whether the condition is met or not, either starts a new sequence of operations or continues on with the old sequence. If the condition is satisfied and the jump is to be executed, the Next Instruction Address is set to the address specified by the transfer ORDER, a fetch is executed and the control is set to start with the left or right ORDER. The sequencer continues from this point. When the condition is not satisfied and the jump is not ex-

ecuted, the sequencer goes ahead to the next step in its cycle. That is, from left ORDER to right ORDER, or from right ORDER to fetch the next instruction.

001 (005) TRANSFER NEGATIVE TO THE LEFT (RIGHT) TNL TNR

A jump to the left (right) ORDER of the instruction specified in the address occurs if the bit in position 0 of the Accumulator is one at the time 001 (005) is executed.

002 (006) TRANSFER PLUS TO THE LEFT (RIGHT) TPL TPR

A jump to the left (right) ORDER of the instruction specified in the address occurs if the bit in position 0 of the Accumulator is zero at the time 002 (006) is executed.

003 (007) TRANSFER OVERFLOW TO LEFT (RIGHT) TFL TFR

A jump to the left (right) operation of the instruction specified in the address occurs if the Overflow Toggle is on. Execution of one of these operations turns the Overflow Toggle off. The Overflow Toggle is turned on by execution of an add type order, 02 or 06 group, which gives a result that exceeds the permitted range of numbers. The overflow is detected by observing three signs; the Accumulator sign, the Number register sign (complemented if subtracting) and the sign of the sum. If two factors to be added have like signs, the sum must have the same sign or an overflow has occurred. The Overflow Toggle is turned on when an overflow occurs and remains on until one of the operations 003 or 007 turns it off. The computer will halt if an overflow occurs and the next operation is not

Transfer Overflow unless the switch on the console is set to Ignore Overflows.

01X TRANSFER OPERATIONS

010 (014) TRANSFER TO THE LEFT (RIGHT) TRL TRR

These operations cause an unconditional transfer of control to the left (right) operation of the instruction specified in the Address part.

011 (015) TRANSFER TO THE LEFT (RIGHT) IF SWITCH T1 IS ON. T1L T1R

A jump occurs if Switch T1 on the console is on. No jump occurs if the switch is off.

012 (016) TRANSFER TO THE LEFT (RIGHT) IF SWITCH T2 IS ON. T2L T2R

013 (017) TRANSFER TO THE LEFT (RIGHT) IF SWITCH T3 IS ON. T3L T3R

02X ADD OPERATIONS

The add ORDERS all bring the word specified by the address associated with the ORDER from the Store to the Number register and produce a result in the Accumulator. The Multiplier-Quotient register is undisturbed. The eight variations of the add ORDER in this group are obtained by combining the ORDERS: clear or don't clear the Accumulator, plus or minus, and absolute value or not absolute value. The Overflow Toggle is turned on by these operations if the result exceeds the conventional range of representable numbers. In all cases, including those which cause the Overflow Toggle to be set, the Adder functions as a forty column binary adder which forms the sum of a number residing in the Accumulator and a number coming from the Number register, complemented in some cases. The result is put back in the Accumulator. A carry that is produced in the high order position is lost.

Subtractions are performed by complementing and adding. Complements are formed by inverting the digits of a number, changing 0's to 1's, and 1's to 0's, and adding a corrective one in the least significant position, that is, in position 39. Overflow is detected by observing the sign bits of the two factors and the result. If the sign bit of the number residing in the Accumulator and the sign bit of the number coming from the Number register are alike, the sign bit of the sum will be the same if no overflow has occurred, and will be opposite if overflow has occurred. If the sign bits of the two factors are not alike, overflow cannot occur. The sign bit of the number coming from the Number register is taken to be the bit

which came from the Store unless a complement is required; in that case, the inverted bit is taken as the sign bit.

020 RESET ADD. RA

This operation replaces the contents of the Accumulator with a word from the Internal Store. The Overflow Toggle cannot be turned on by 020.

The steps in the operation are:

Clear the Accumulator to zero.

Read the specified word from the Store to the Number register.

Add and put the result in the accumulator

Set O.F. toggle if overflow occurred.

021 RESET SUBTRACT RS

This operation puts the complement (2's complement if the number is regarded as a fraction) of the word from the Store in the Accumulator. The 021 operation can set the Overflow Toggle if the word specified by the address is -1. That is, 1 in the high order position followed by 39 zeros. In this case, the computer control circuitry examines the sign bit coming from the complement gate and finds it to be zero which agrees with the sign bit of the Accumulator that was just cleared to zero. When the corrective one is added, the carry propagates to the high order position. The result has a one in the sign position so the Overflow Toggle is turned on.

The steps in the operation are:

Clear the Accumulator to zero.

Read the specified word from the Store to the Memory register.

Complement and Add.

Put the result in the Accumulator.

Set O.F. toggle if overflow occurred.

022 RESET ADD ABSOLUTE VALUE RAV

The 022 operation is conditionally an 020 or an 021 operation. The choice of 020 or 021 is made by examining the sign bit of the number that is brought from the Store to the Number register and selecting the operation that is expected to give a positive result. The number, -1, is an exception. The 022 operation applied to this number sets the Overflow Toggle and yields a result which has a one in the sign bit. (See 021). This operation followed by an appropriate conditional transfer can be used to test for -1.

The steps in the operation are:

Clear the Accumulator to zero.

Read the specified word from the store to the Number register.

Examine the sign bit of the Number register and choose (a) or (b).

(a) Add if the sign bit is zero.

(b) Complement and Add if the sign bit is one.

Put the result in the Accumulator.

Set O.F. toggle if overflow occurred.

023 RESET SUBTRACT ABSOLUTE VALUE RSV

The 023 operation is conditionally an 020 or an 021 operation. The operation is selected which is expected to yield a negative result in the Accumulator. Zero is the exception; it gives a result

zero which has a zero in the sign bit. The 023 operation followed by a Transfer Plus can be used to test for zero. The steps in the operation are:

Clear the Accumulator to zero.

Read the specified word from the Store to the Number register.

Examine the sign bit of the Number register and choose (a) or (b).

(a) Complement and Add if the sign bit is zero.

(b) Add if the sign bit is one.

Put the result in the Accumulator.

Set O.F. toggle if overflow occurred.

024 ADD A

The word specified by the address is added to the number residing in the Accumulator and the result is put back in the Accumulator. The Overflow Toggle may be set by this operation.

The steps in the operation are:

Read the specified word from the Store to the Number register.

Add.

Put the result in the Accumulator.

Set O.F. toggle if overflow occurred.

025 SUBTRACT S

The steps in the operation are:

Read the specified word from the store to the Number register.

Complement and add.

Put the result in the Accumulator.

Set O.F. toggle if overflow occurred.

026 ADD ABSOLUTE VALUE AV

The steps in the operation are:

Read the specified word from the Store to the Memory register.

Examine the sign bit of the Memory register and choose (a) or (b).

(a) If the sign bit is zero, add.

(b) If the sign bit is one, complement and add.

Put the result in the Accumulator.

Set O. F. toggle if overflow occurred.

027 SUBTRACT ABSOLUTE VALUE SV

The steps in the operation are:

Read the specified word from the Store to the Number register.

Examine the sign bit of the Number register and choose (a) or (b).

(a) If the sign bit is zero, complement and add.

(b) If the sign bit is one, add.

Put the result in the Accumulator.

Set O.F. if overflow occurred.

03X MULTIPLICATION

The basic multiplication process is the same for all variations of the multiply ORDER. Initially the multiplier is in the Multiplier Quotient register MQ. It can be put there by the Load MQ ORDER, 004, or it can be left in MQ by some other ORDER. The address associated with the multiply ORDER specifies the multiplicand. The result is a double length product in A and MQ. The Accumulator contains the sign bit and the high order 39 bits of the product; MQ contains zero in its sign bit and the 39 low order bits of product.

The multiplicand is read from the Store and put in the Number register at the beginning of the ORDER and remains there throughout the succeeding steps. The complement gates are set to complement if the ORDER is negative multiply. The first 39 steps are conditional add and shift right ORDERS. At each step the decision to add or not is made by examining the bit in position 39 of MQ. The add is called for when the multiplier bit is a 1 and is omitted when it is zero. The right shift always occurs. It serves a dual purpose. The partial product in A and MQ is divided by two and the multiplier is moved to the right to bring the next bit into position 39 where it can be examined. The low order bits of the product are shifted into the high order positions of MQ as they are vacated by the multiplier. After 39 steps of this kind, the multiplier sign bit is in MQ position 39 and the partial product is in the other positions of A and MQ. The final step is a conditional corrective subtraction (i.e. the complement gate is reversed if the multiplier

sign bit is one); followed by a shift of MQ only which brings zero into the MQ sign bit and puts the low order 39 bits of product in standard word position in MQ. The eight variations of the multiply ORDER are obtained by combinations of clear or don't clear the Accumulator at the start of the ORDER, insert a one in position one of A or don't, and Add or Add the complement of the multiplicand. The multiply ORDERS cannot turn on the Overflow Toggle -- no circuitry is provided for testing for overflows.

030 MULTIPLY ROUND - MULTIPLY AND ADD PLUS "ONE-HALF" MR

The Accumulator is cleared to zero and a one is inserted in position 1 at the start of the multiply operation. Thereafter the multiply is carried out as described above. By the time the multiplication is completed, the inserted bit has been shifted 39 times so that one has been added to the high order position of the MQ part of the double length product. If the result is a positive fraction, this is the conventional half adjustment of the product. The rounded product is produced in the Accumulator. Caution: Adding plus one-half $\cdot 2^{-39}$ to the complement is not equivalent to the conventional rounding of negative fractions.

031 MULTIPLY NEGATIVELY AND ROUND MNR

The Accumulator is cleared to zero and plus one-half, a bit in position 1, is inserted. The complement of the multiplicand is added to the partial products as the multiplication progresses, so the result is the negative product plus one in the 40^{th} position. This gives the usual rounded fraction in A only if the result is positive.

032 MULTIPLY M

The Accumulator is cleared to zero at the start of the operation. The double length product of the multiplier in MQ and the multiplicand from the Store is formed in A and MQ. The sign bit and the 39 high order bits of product are put in A; the low order 39 bits of product are in MQ. The sign bit of MQ is zero.

033 MULTIPLY NEGATIVELY MN

The Accumulator is cleared to zero at the start of the operation. The negative of the double length product of the multiplier in MQ and the multiplicand from the Store is formed in A and MQ.

036 MULTIPLY AND ACCUMULATE MA

The contents of the Accumulator is retained and added to the product as it is developed. Since the word which resided in the Accumulator at the start of the operation shifted as the multiplication progresses, it is added to the low order positions of the product.

037 MULTIPLY NEGATIVELY AND ACCUMULATE MNA

The contents of the Accumulator is retained and added as in 036. The complement gates are set so that the negative product is developed.

034, 035 MULTIPLY - BOTH ROUND AND ACCUMULATE MB MNB

In these operations the Accumulator is <u>not</u> cleared, but one is inserted in position 1; i.e., it stays a 1 if it was 1 and is made 1

if it was zero.

The positive product, 034, or negative product, 035, is developed and added to the number in the Accumulator.

EXAMPLE: MULTIPLICATION BY A NEGATIVE MULTIPLIER

7/8	7/8	7/8	0.111
- 5/8	-1+(1-5/8)	-1+3/8	1.011
-35/64	-1+(1-35/64)	-1+29/64	1.011,101

STEPS IN MACHINE MULTIPLICATION (4-BIT MACHINE)

	A	MQ	M
	0.000	1.011	0.111
Add	0.111		
	0.111		
Shift			
	0.011	1.101	
Add	0.111		
	1.010		
Shift			
	0.101	0.110	Examine Last Multiplier Bit
No Add			
	0.101		
Shift			
	0.010	1.011	
Comp. and Add	1.001		
Position MQ			
	1.011	0.101	

Note that some intermediate sums can exceed the normal range of representable numbers. This is handled by having one extra toggle in the Accumulator temporary register and reconstructing the proper bit in it before the right shift.

EXAMPLE: MULTIPLICATION WITH NEGATIVE MULTIPLICAND

- 5/8	-1+(1-5/8)	-1+3/8	1.011
7/8	7/8	7/8	0.111
- 35/64	-1+(1-35/64)	-1+29/64	1.011,101

STEPS IN MACHINE MULTIPLICATION (4-BIT MACHINE)

	A	MQ	N
	0.000	0.111	1.011
Add	1.011		
	1.011		
Shift			
	1.101	1.011	Examine
Add	1.011		
	11.000		least
Shift			
	1.100	0.101	multiplier
Add	1.011		
	10.111		bit
Shift			
	1.011	1.010	
No Add			
Shift MQ	1.011	0.101	

Note: The extra high order position shown in the example is not part of the adder; it is reconstructed in the extra Accumulator Temporary toggle.

04X DIVISION

The basic division process is the same for all variations of the divide operation. The dividend resides in A and MQ at the start of the operation. The divisor is specified by the address associated with the operation. The quotient which is generated by repeated trial subtractions, is put in MQ and the Remainder is left in the Accumulator.

At the start of the division process the Divisor is read from the Store and put in the Number register. If the operation is negative divide, the complement of the number in the Number register is treated as the "Divisor". The "Divisor" sign bit and the Dividend sign bit, i.e. the sign bit of A, are examined to determine the sign of the Quotient. If the Quotient is negative, one is put in the sign bit of MQ, and the "Divisor" is added to the word in the Accumulator. If the Quotient is positive, zero is put in the sign bit of MQ; no corrective addition is required. The number in A and MQ is shifted left one place, skipping the sign bit of MQ register which contains the sign of the Quotient. A trial subtraction of the "Divisor" is made, and a one or a zero is selected to be put in position 39 of MQ to indicate that the "Divisor" goes, or does not go. Go or no go is selected by examining the sign bits of the "Divisor" and the result of the trial subtraction. If the sign bits agree, one is written in the quotient, and the result of the subtraction is taken. If the sign bits disagree, zero is written in the Quotient and the result of the subtraction is rejected. The left

shift which follows the generation of a Quotient digit does double duty; it moves the quotient left, bringing the newly formed digit into position 39 and also positions the intermediate remainder for the next trial subtraction. The trial subtractions are continued until all the 39 digits of the Quotient have been generated. The final step consists of shifting the remainder right one place so that it will occupy standard word position in the Accumulator. The factors in the division operation always satisfy the following relationship:

Quotient times Divisor plus Remainder equals Dividend.

This is true for intermediate results as well as final results, provided allowance is made for the changing positions of the factors. For example, at the start of the operation the quotient is zero, and the remainder is equal to the dividend. If the quotient is negative, minus one is put in the quotient, and the divisor is added to the remainder. The equality still holds. All other digits are positive, and each one that is inserted is accompanied by a subtraction.

The rule which is used to recognize "go" and "no go" fails to give the expected result in one situation. If the divisor is negative and a zero remainder turns up, the signs disagree and "no go" is selected instead of "go". The final remainder will be equal to the divisor and the quotient will be too small by one in the least significant position. Note, however, that the above algebraic relationship is still satisfied.

Division can give results which are outside the range of representable numbers. No safeguards or alarms are built in the machine. There is no simple rule for reconstructing the correct quotient and remainder from the results of an improper division.

040 DIVIDE WITH SHORT DIVIDEND DS

Clear MQ and divide the number in the Accumulator by the number specified in the Store. Put the Quotient in MQ and the Remainder $\cdot 2^{+39}$ in the Accumulator.

041 DIVIDE NEGATIVELY WITH SHORT DIVIDEND DNS

Clear MQ and divide the number in the Accumulator by the negative of the number specified in the store. Put the Quotient in MQ and the Remainder $\cdot 2^{+39}$ in the Accumulator.

044 DIVIDE D

Divide the double precision dividend in A and MQ by the single precision divisor specified by the address. Put the Quotient in the MQ register and the Remainder $\cdot 2^{+39}$ in the Accumulator. The contents of the MQ sign bit at the start of the division is irrelevant.

045 DIVIDE NEGATIVELY DN

Divide the double precision dividend in A and MQ by the negative of the single precision divisor specified by the address. Put the Quotient in the MQ register and the Remainder $\cdot 2^{+39}$ in the Accumulator. The contents of the MQ sign bit at the start of the division is irrelevant.

EXAMPLE: DIVISION WITH QUOTIENT NEGATIVE

$$-35/64 \div 7/8 = -5/8$$

$$(-1 + 29/64) \div 7/8 = -1 + 3/8$$

$$1.011{,}101 \div 0.111 = 1.011$$

STEPS IN MACHINE DIVISION (4-BIT MACHINE)

	A	MQ	N
	1.011	0.101	0.111
Examine signs	↑		↑
Choose Neg. Quotient		1	
Add	0.111	1.101	
	0.010	1.101	
			0 Insert
Shift Left			
	0.101	1.010	
Trial Sub.	1.001		
No Go	1.110		0 Insert
Shift Left			
	1.010	1.100	
Trial Sub.	1.001		
Go	0.011		1 Insert
Shift Left			
	0.111	1.001	
Trial Sub.	1.001		
Go	0.000		1 Insert
Shift Left			
	0.000	1.011	
Position Rem. A Rt.	0.000	1.011	

EXAMPLE: DIVISION WITH NEGATIVE DIVISOR

$-36/64 \div -5/8 = +7/8$ with remainder $-1/64$

$(-1+28/64) - (-1+3/8) = 7/8$ with remainder $(-1+7/8)\cdot 2^{-3}$

STEPS IN MACHINE DIVISION (4-BIT MACHINE)

	A	MQ	M
	1.011	0.100	1.011
Examine Signs	↑		↑
Set Quotient Plus		0.	0 Insert
Shift Left	0.111	0.000	
Trial Sub.	0.101		
	1.100		
Go			1 Insert
Shift Left	1.000	0.001	
Trial Sub.	0.101		
Go	1.101		1 Insert
Shift Left	1.010	0.011	
Trial Sub.	0.101		
	1.111		
Go			1 Insert
Shift Left	1.110	0.111	
Power Shift Aright	1.111	0.111	

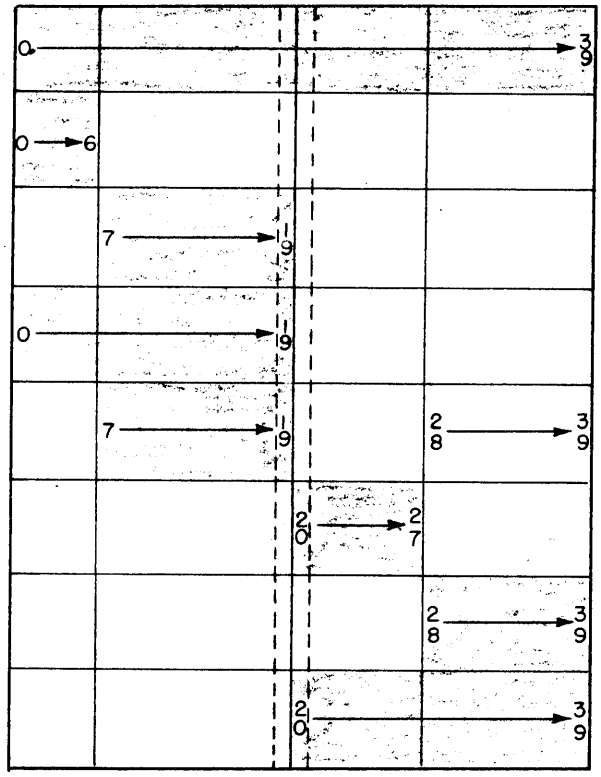
05X STORE AND SUBSTITUTE OPERATIONS

The 05 group of operations cause information to be transferred from the Accumulator to the STORE. The operation part determines which of the forty positions of the Accumulator are to be put in the storage location specified by the address part. Information is always transferred from the Accumulator to corresponding positions of the word in the Store. Those parts of the word in the Store that are not replaced by information from the Accumulator are undisturbed. The contents of all registers remain unchanged during this operation.

The forty channel bus is divided in four parts which can be manipulated separately or in groups to give various operations. The following parts can be substituted:

Positions Substituted are Shaded

050	Full Word	ST
051	Left Operation	SOL
052	Left "Address"	SAL
053	Left Half Word	SHL
054	Both Addresses	SAB
055	Right "Operation"	SOR
056	Right Address	SAR
057	Right Half Word	SHR



The partial substitution operations make it convenient to substitute parts of an instruction without disturbing the rest of the word. Note, however, that the group labeled left address includes one of the unused bits in the instruction, and that the group labeled right operation includes the other unused bit.

06X MULTIPLIER-QUOTIENT TO ACCUMULATOR AND MEMORY OPERATIONS

The 06 group of operations provide a way for putting the contents of the MQ register in the Store. The path over which this information travels is MQ to Number register, Number register to the Accumulator via the Adder, and from the Accumulator to the Store. Since the information goes via the Adder, all the eight options of the Add operation are provided. The result is put in the Accumulator and in the specified location in the Store. The word in the MQ register is not changed. The operations can be described as follows (see the description of the 02 group for details of the operations):

060	Reset Add the word in MQ into A and store the result in the specified location.	STQ
061	Reset Subtract the word in MQ into A and store the result in the specified location.	SNQ
062	Reset Add the Absolute Value of the word in MQ into A and store the result in the specified location.	SVQ
063	Reset Subtract the Absolute Value of the word in MQ into A and store the result in the specified location.	SNV
064	Add the word in MQ to the word in A and store the result in the specified location.	AQS
065	Subtract the word in MQ from the word in A and store the result in the specified location.	SQS

066 Add the Absolute value of the word in MQ to the word in A and store the result in the specified location. AVS

067 Subtract the absolute value of the word in MQ from the word in A and store the result in the specified location. SVS

07X SHIFT OPERATIONS

The shift operations can be divided into two groups which differ only by the clearing or not clearing of the Multiplier Quotient Register before the start of the actual shifting operation. The number of places to be shifted is specified by an integer in the address part associated with the shift operation. Shifts are limited to a maximum of 127 places.

070 (074) ACCUMULATOR TO THE RIGHT - SHORT RIGHT SRC SRH

The operation 070 clears the MQ before the start of the shift; 074 does not. The shift moves the bits in the Accumulator to the right the specified number of positions. Zeros are brought into the zero position of A and bits are dropped from position 39 of A as the shift progresses. MQ is not shifted. This shift is not equivalent to dividing the word in the Accumulator by a power of two if the word is negative.

071 (075) LONG CIRCULAR SHIFT TO THE LEFT CLC CLH

The operation 071 clears MQ before the actual shifting; 075 does not. The Accumulator and MQ are connected into an eighty position ring by connecting position 0 of MQ to position 39 of A and position 0 of A to position 39 of MQ. The shift moves the information the specified number of places to the left in the ring. A shift of 40 places exchanges the words in A and MQ. A shift of 80 places returns the words to their original locations.

072 (076) LONG RIGHT POWER SHIFT

The operation 072 clears MQ before the actual shifting occurs; 076 does not. The words in A and MQ are treated as a double length representation of a number. The shift is equivalent to dividing the double length number by a power of two, if the sign bit of MQ is ignored. Bits that are shifted out of position 39 of A skip the sign position and go into position 1 of MQ. The sign bit of the Accumulator remains the same and is copied into position 1 of A each time A is shifted one place. This maintains the correct representation of either positive or negative numbers as they are shifted right. Recall that lead zeros of a positive number turn out to be lead ones when the number is complemented. Since one of the anticipated uses of this operation is to set up a product as a multiplier with one operation, the sign bit of A is copied into the sign bit of MQ.

073 (076) LONG LEFT POWER SHIFT

LLC
LLH

The operation 073 clears MQ before the actual shifting; 077 does not. The words in A and MQ are treated as a double length number. The shift is equivalent to multiplying the double length number by a power of two. Position 1 of MQ is coupled to position 39 of A. Bits shifted out of Accumulator position 0 are lost. No provision for recording overflows is provided. Zeros are brought into the MQ positions vacated by shifting. The MQ sign bit is not changed.

10X INPUT OUTPUT OPERATIONS

10.0 SELECT INPUT OR OUTPUT DEVICE SEL

The operation 10.0 causes the JOHNNIAC to select an input or an output device designated by an integer in the address part. Currently all digits of the address except the least octal are irrelevant. The least octal digit designates selections as follows:

0 Selects the primary feed of the card reader

1 Selects the secondary feed of the card reader

2 Selects the feed mechanism of the card punch

3 Selects the feed and echo mechanisms of the card punch

4 Selects the ANelex printer and spaces the paper one space

5 Selects the ANelex printer without spacing the paper

10.1 COPY OPERATION C

The interpretation of the copy operation is a function of the input-output device selected.

10.1 COPY FROM THE CARD READER C

Cards are read face down nine edge first, one row at a time. Each row is read as two forty-bit words. Holes are interpreted as ones; blanks as zeros. The standard board is wired 80-80. The binary word read from card columns 1 through 40 is put in the Store at the location specified by the address; the binary word read from card columns 41 through 80 is put in the Accumulator. Twelve copies are required to read a full card. The computer is released to go about its business between copies; however, interlocks are provided

to signal late copy and give a copy check if too much calculation is interposed. A copy check also occurs if more than twelve copies are given; but fewer than twelve, including none, can be given without causing a copy check.

10.1 COPY TO THE CARD PUNCH C

When the card punch is selected one card is fed past the punching station nine edge first with face down. During this feeding cycle twelve copy instructions can be given to cause twenty-four binary words to be punched in the twelve rows of the card. Ones cause holes to be punched, zeros do not. Assuming the punch board is wired 80-80, at each copy the word in the Accumulator is punched in card columns 41 through 80 and the word from the Store specified by the address is put in the Number register and is punched in card columns 1 through 40. Copy checks can result from giving a copy to the punch too late or from giving too many copy instructions. Giving fewer than twelve copies does not cause a copy check.

If the card punch is selected to punch, then the above description of the operation is complete; but if echos are requested, the following reading also occurs. After each row is punched, the corresponding row of the <u>preceding</u> card is read. The binary word from card columns 41 through 80 is put in the Accumulator, and the binary word from card, columns 1 through column 40 is put in the MQ register. The same copy checks as above can occur.

A copy check which occurs during card reading or card punching or card punching and echoing, causes the computer to halt immediately. It must be restarted manually. The card machinery will com-

plete the card cycle during which the copy check occurred.

10.1 COPY TO THE ANelex PRINTER C

Each copy given after the selection of the printer causes forty bits to be brought from the specified location in the Store, put in the Number register and presented to the printer. Ones in the first word presented cause zeros to be printed in the appropriate columns. The next copy causes ones to be printed, etc. Ten copies are required to print the ten decimal characters on a line. A copy check can result from a late copy or from too many copies. Giving fewer than ten copies does not cause a copy check. The contents of A and MQ are undisturbed by the print copy operation.

10.4 DISPLAY DIS

This operation causes the word in the register selected by a button on the console to be displayed on the display unit of the console unless the display unit is locked out by the display delay mechanism on the console.

10.5 HOOT HUT

This operation causes a pulse to be emitted to the audio amplifier and speaker attached to the JOHNNIAC.

10.6 EJECT THE PAGE OF THE ANELEX PRINTER EJ

11X DRUM OPERATIONS

11.0 READ A BLOCK OF DATA FROM THE DRUM TO THE STORE RD

11.1 WRITE A BLOCK OF DATA FROM THE STORE ON THE DRUM WD

The following description applies to both the read and write operations provided the word "transfer" is understood to mean the transfer of words either to or from the drum. Also, provision is made for four drums. One is installed.

The description which follows is written for the initial drum layout which has 1024 words in each band. The capacity of the drum is 12 x 1024 = 12,288 in this case. Soon after the drum operation has been proved the engineers expect to double the capacity of the drum by writing at twice the pulse density. Then each band will contain 2048 words, and the description below will need to be modified accordingly.

The 11.X operations assume that the MQ register contains the following information pertinent to the drum operation; a selection of a drum, a position for the movable head mechanism, a band under the heads, and a first and last drum address within the band. The first address is put in the left address positions of the word in MQ. The drum, position, and band selections are put in the three octal digits of the right operation part. The last address is put in the right address part. The left operation part as well as the high order bits of the addresses are irrelevant. The address part associated with the 11.X operation specifies the first address in the ~~Store.~~ MEMORY.

The contents of MQ is ignored by the drum circuits until the 11.0 order is executed.

The d, p and b specified in the right operation digits of the word in MQ select the drum, the position, and the band. The d selects the drum, p selects one of three positions at which there are 4 bands, and b selects one of the 4 bands. The d selection takes about 3 ms. The p selection takes about 33 ms. The b selection takes about 3 ms. The selection time is determined by the d, p or b digit that is changed from one drum operation to the next. If more than one digit is changed, the selection time is determined by the slowest selection. If no digits are changed, the selection time is essentially zero. When writing on the drum, a gap is left so that another band may be selected in the interval between addresses $(1777)_8$ and $(0000)_8$.

The minimum access time to the first word is the selection time. The maximum access time to the first word is the selection time plus the drum revolution time, 33 ms. The time for a complete drum order is the access time plus the time for the word transfer. The time for transferring a complete band of 1024 words is two drum revolutions. The rate of word transfer is approximately 33/512=65μs/word.

A band is divided into two interlaced sets of 512 words each. The f_1 and l_1 digits of the first and last addresses determine the set. The remaining digits determine the word in the set. The complete addresses are interpreted modulo $(2000)_8$. If the first and last addresses call for words in different sets, the words in the set called for by the start address, starting with the word in the first address, will be transferred until the word in address $(777)_8$ is

reached. The words in the set called for by the last address will then be transferred up to and including the word in the last address. For example, f = $(0350)_8$ and l = $(1210)_8$ will transfer words in addresses $(0350)_8$ to $(0777)_8$ and $(1000)_8$ to $(1210)_8$ inclusively. Similarly f = $(1210)_8$ and l = $(0350)_8$ will transfer words in addresses $(1210)_8$ to $(1777)_8$ and $(0000)_8$ to $(0350)_8$ inclusively.

Operation 11.0 transfers from the drum to the high speed store via the Accumulator register. At the end of the order, the word in A is 0. MQ contains the drum addresses.

Order 11.1 transfers from the high speed Store to the drum via the Number register. At the end of the order, the word following the last word transferred is in NR. The MQ contains the drum addresses.

The block of information that is transferred to or from the drum is arbitrary. The block length can vary from one word to the full band of 1024 words. Any band address may be assigned as first or last address. Interpretation of band addresses is modulo $(2000)_8$. Requesting a transfer with first band address less than or equal to the last band address implies that the transfer is to start with the first address and end with the last address. Requesting a transfer with first band address larger than last band address implies that transfer is to start with the first address, continue to $(1777)_8$; resume with 0000, and continue until the word in the last band address has been transferred. For example, the entire band could be transferred by specifying any of the following band address combinations:

First band address	Last band address
0000	$(1777)_8$
0001	0000
$(0555)_8$	$(0554)_8$

The time required to execute a drum operation may be divided into four parts:

A) The selection time. 3.5 ms or 370 ms

This is the time required to select a position and a band. Whenever a new position is requested, the entire head assembly must be moved. This positioning operation may take as much as 370 ms. If the drum is already in position, the positioning time can be ignored. The band switching circuits do not remember a selection from one drum

operation to the next, hence a band selection is always required. The minimum selection time is the band selection time 3.5 ms.

B) Wait time for the first band address. 0 to 36 ms.

This is the time required after the selection is completed for the drum to revolve to the position that brings the first band address under the heads. It can vary from zero to a full drum revolution time of 36 milliseconds. The average wait time is expected to be 18 ms.

C) Word transfer time ~ 70 μs per word

The interlace used on the drum puts successive drum addresses in every other physical location. Two complete revolutions are required to transfer a complete block of 1024 words. Neglecting the small gap that is left in the band, this gives a transfer rate of 72 ms/1024 words ≃ 70 μs/word.

D) Band switching drop out time 5.5 ms

The band selection circuits require 5.5 ms to drop out after the word transfers have been completed.

Operation 11.0 transfers from the drum to the high speed store via the Accumulator register. At the end of the order, the word in A is 0. MQ contains the drum selection and addresses.

Operation 11.1 transfers from the high speed Store to the drum via the Number register. At the end of the order, the word following the last word transferred is in NR. The MQ contains the drum selection and addresses.

To summarize the drum operations:

Load MQ with a conditioning word as follows:

0 - 6	7 - 18	19 - 21	22 - 24	25 - 27	28 - 39
Not used	First Band Address 0000 to $(1777)_8$	Drum 0	Pos. 0 1 2	Band 0 1 2 3	Last Band Address 0000 to $(1777)_8$

Read Drum ← → 11.0, First Core Address
or
Write Drum 11.1, First Core Address

Timing:

Selection	Wait	Word transfer	Selection drop out
3.5 ms or 370 ms	0-36 ms	~ 70 μs/word	5.5 ms

12 INTERSECTION OR LOGICAL PRODUCT OPERATIONS

These operations form the bit by bit intersection of a word from the Store and the word in the Accumulator. The word from the Store comes via the Number register and complement gates so that various complementations can be performed prior to the intersection operation. Logical complements are obtained by changing 1's to 0's and 0's to 1's.

Operations 12.0 through 12.3 clear the Accumulator to zero at the start of the operation. Since the intersection of zero with any other word is zero the result of these operations regardless of the address is to clear the Accumulator to zero.

12.4 POSITIVE INTERSECTION PI

The bit by bit intersection of the word specified in the Store and the word in the Accumulator is formed and put in the Accumulator.

12.5 NEGATIVE INTERSECTION NI

The logical complement of the word specified in the Store is used to form the intersection with the word in A. The result is put in A.

12.6 CONDITIONAL POSITIVE INTERSECTION PMI

If the bit in position zero of the word from the Store is zero, form the intersection of this word with the word in A and put the result in A.

If the bit in position zero of the word from the Store is one, invert the digits of this word and form the intersection with the

word in A. Put the result in A.

12.7 CONDITIONAL NEGATIVE INTERSECTION NMI

If the bit in position zero of the word from the Store is one, form the intersection of this word with the word in A, and put the result in A.

If the bit in position zero of the word from the Store is zero, invert the binary digits of this word and form the intersection of this word with the word in A and put the result in A.

13 HALT AND TRANSFER CONTROL OPERATIONS

13.0	(13.4)	UNCONDITIONAL HALT AND TRANSFER	HTL HTR

Halt the computer. Operation is resumed, starting with the left (13.0) or right (13.4) operation of the instruction specified by the address, when the GO button is depressed.

13.1	(13.4)	HALT CONDITIONED BY SWITCH H_1 AND TRANSFER	H1L H1R
13.2	(13.6)	" " " " H_2 " "	H2L H2R
13.3	(13.7)	" " " " H_3 " "	H3L H3R

If the conditioning switch is on, these operations are unconditional transfer operations.

McGee

JOHNNIAC Operation Times

(6-6-55)

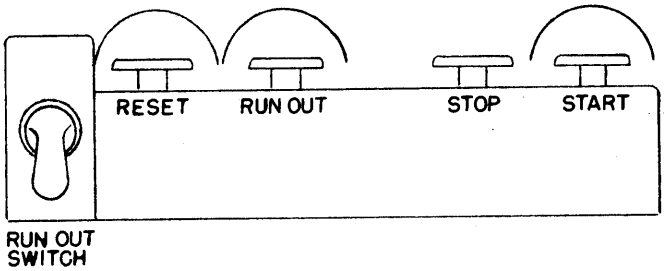
OP	
000	30 μs
004	36 μs
00X	31 μs
01X	39 μs
02X	80 μs
03X	1400 μs (Max)
04X	1400 μs (Max)
05X	35 μs
06X	90 μs
07X	15 μs + 18·n μs
100	See Input Output descriptions
105	
11X	See Drum operations
12X	80 μs

Input-Output Equipment

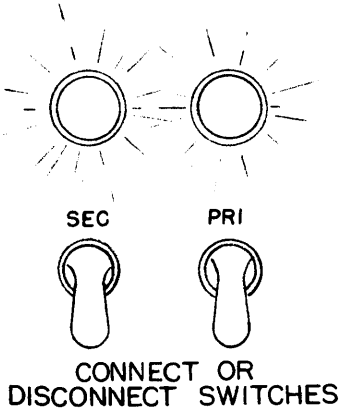
CARD READER (IBM COLLATOR MODEL 77)

The collator can read 80 columns from either the primary or the secondary feed at 240 cards per minute. Selection of the feed is controlled by the program. Operation 10.0 with address part 0000 selects the card reader and causes a card to feed in the primary feed. Operation 10.0 with address part 0001 selects the card reader and feeds a card in the secondary feed. Twelve copy instructions following the card feed cause the entire card image to be read. Holes are interpreted as 1's and blanks as zeros. The left forty columns are read and stored at the location specified in the address part of the copy instruction, and the right forty columns are read and put in the Accumulator. The standard board is wired 80-80. It ejects primary cards to hopper 1 and secondary cards to hopper 2. All collator features remain on the plug board after providing 160 connections to the JOHNNIAC and can be used. See plug board diagram

Collator Switches and Buttons



ORIGINAL SWITCHES



CONNECT OR DISCONNECT SWITCHES

Three new toggle switches and two indicator lights have been added to the collator. The two toggle switches that connect or disconnect the primary and the secondary feeds respectively are located on the end of the collator below the primary feed. An indicator light that is on when the feed is connected is located just above each switch. The Run Out Switch is located on the front of the collator near the original manual control buttons.

To Read Cards in Primary Feed Only

1. Connect primary. Disconnect secondary using new switch. Indicator light is on when feed is connected.

2. One blank card must be placed ahead of cards to be read. Collator requires all positions of feed to be filled before it gives the go ahead signal to JOHNNIAC. Cards are fed FACE DOWN 9 EDGE first.

3. Start button. Pressing start feeds cards to fill the feed. Collator halts in ready state with the motor running.

4. Empty hopper causes collator motor to stop and JOHNNIAC to wait. More cards can be added and motor restarted with start button, or cards can be run out to complete reading, using the one original run-out button only.

5. To run out cards, depress both the run-out button and the run-out switch. This empties both feeds. Note: If collator primary or secondary is connected and empty, JOHNNIAC will wait until they are disconnected before punching. No

information is lost.

To Read Cards in Secondary Feed

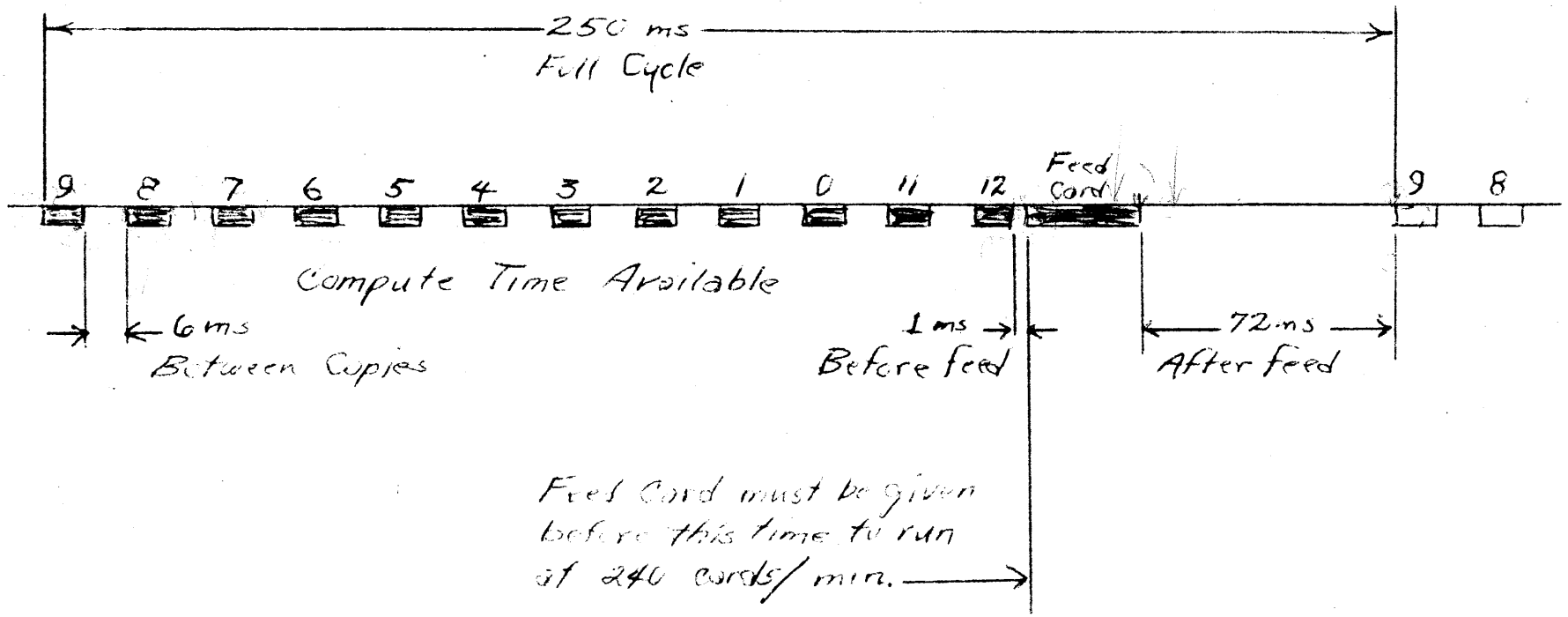
The same as primary except:

1. Connect secondary.

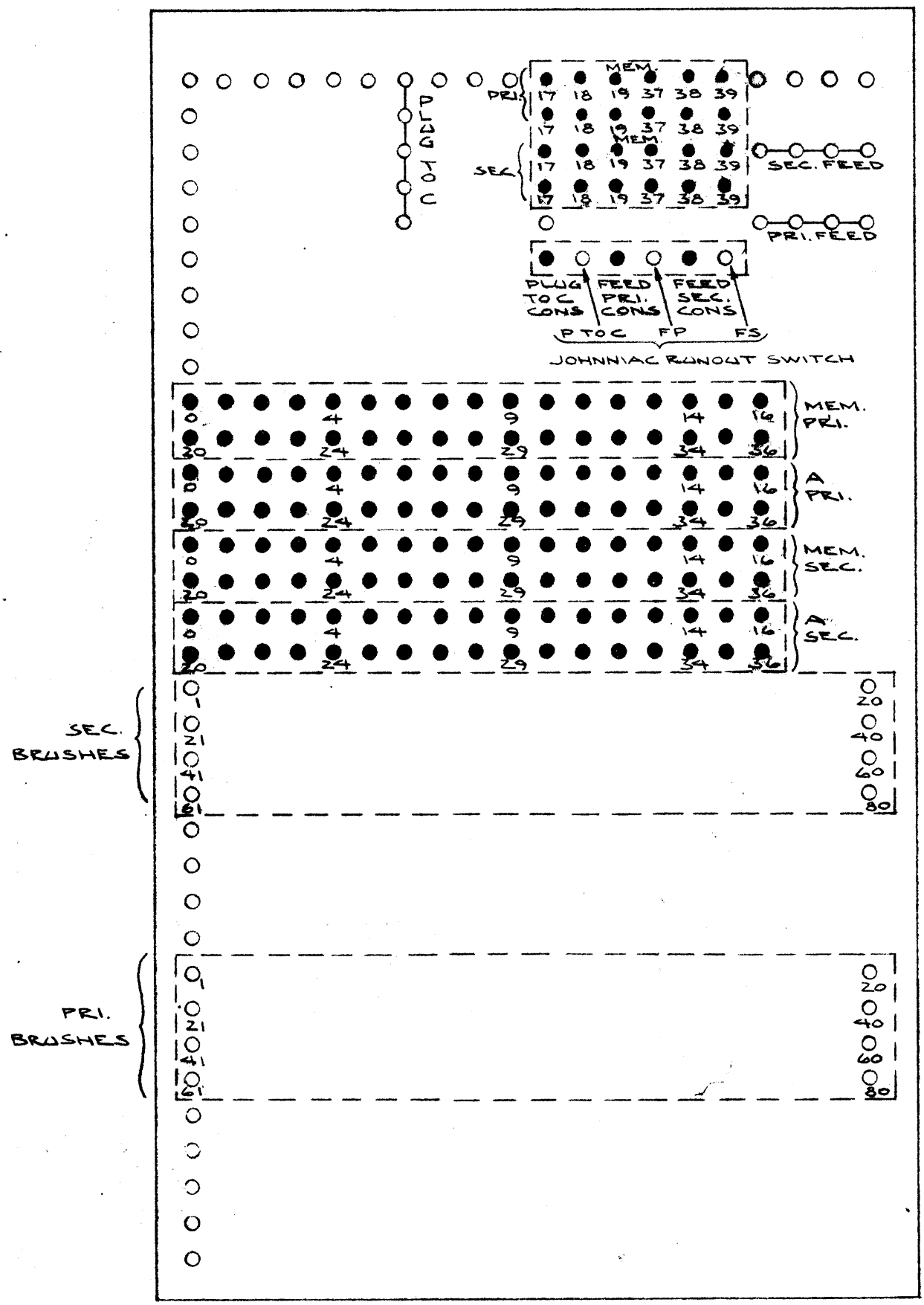
2. Two blanks ahead of cards to be read. (One is ejected when collator is made ready.)

3. Primary feed must be filled even though it is not connected to JOHNNIAC.

CARD READER — Approximate Timing



COLLATOR PLUG BOARD REVISED FOR JOHNNIAC



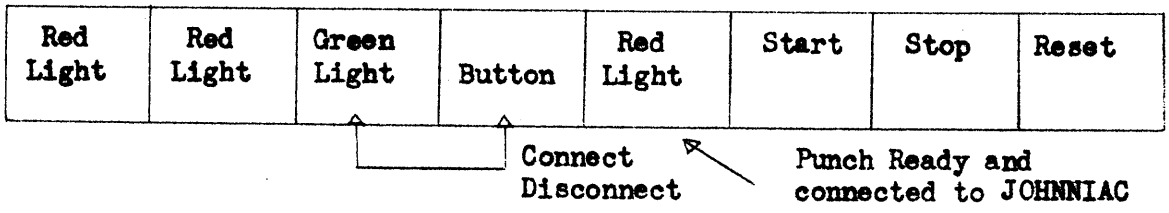
● = SOLID HUBS TO JOHNNIAC

○ = CIRCLE HUBS TO COLLATOR

CARD PUNCH (IBM Punch Model 523)

The card punch can punch 80 column cards or punch a card and echo the previously punched card. Operation 10.0 with address part 0002 selects the punch and feeds a card. Operation 10.0 with address part 0003 selects the punch and echo mode of operation and feeds a card. Twelve copies must be given to copy the card image to the card. The contents of the Accumulator is punched in the right 40 columns of the card, and the contents of the address part of the copy instruction is read into the Number register and punched in the left 40 columns of the card. Holes are punched corresponding to ones in the two words in the row under the punch dies when the copy is given. Cards are normally fed FACE DOWN 9 EDGE FIRST. The standard board is wired 80-80. Other boards can be wired to use the emitter, gang punch features, etc. if desired. Caution: The labeling on the punch boards assumes cards are fed face up 12 edge first.

Card Punch Buttons and Lights



To Punch Cards

1. Place cards in hopper FACE DOWN 9 EDGE first.

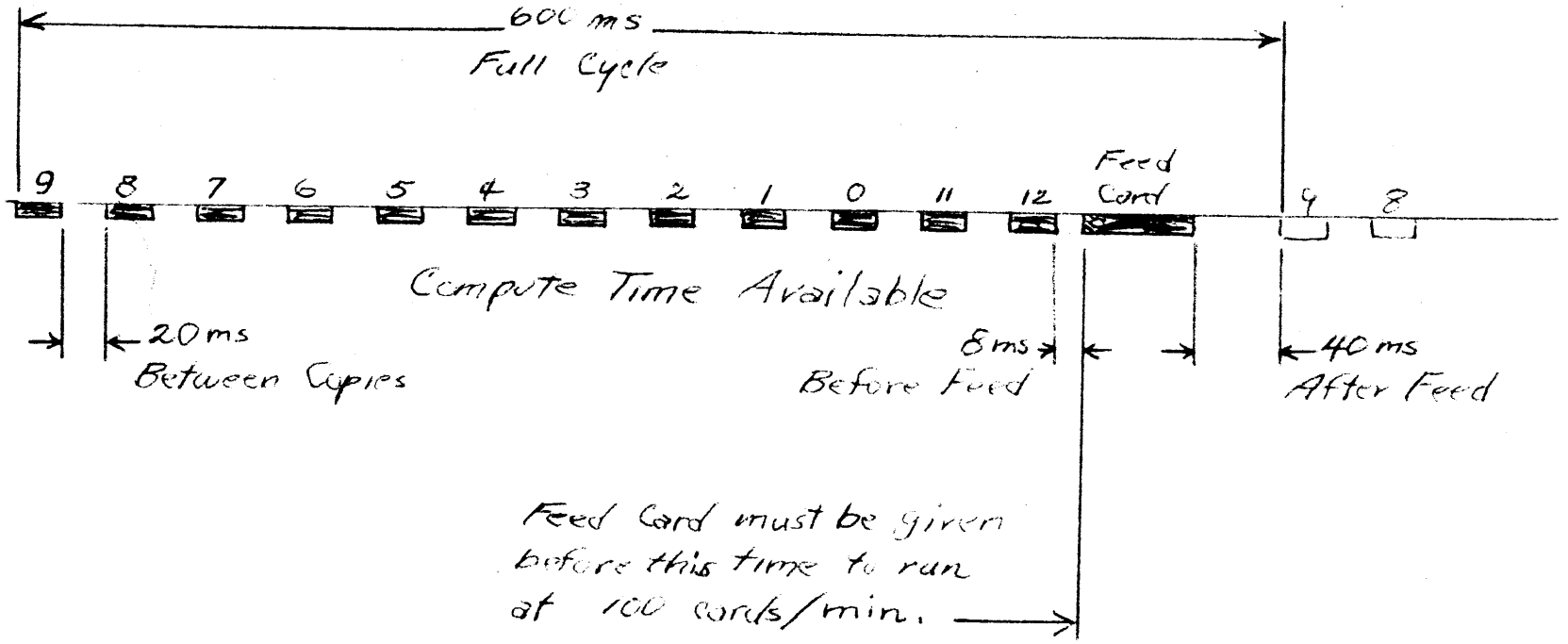
2. Hit the Start button to feed cards and put the punch in Ready State.

3. Connect the punch to JOHNNIAC. The Connect-Disconnect button counts. Cards can be fed, gang punched, etc. when punch is disconnected without affecting JOHNNIAC. If feed punch operation is called for, JOHNNIAC will wait until punch is connected. The stop button does not disconnect the punch. Always disconnect when adding more cards, correcting misfeeds, etc.

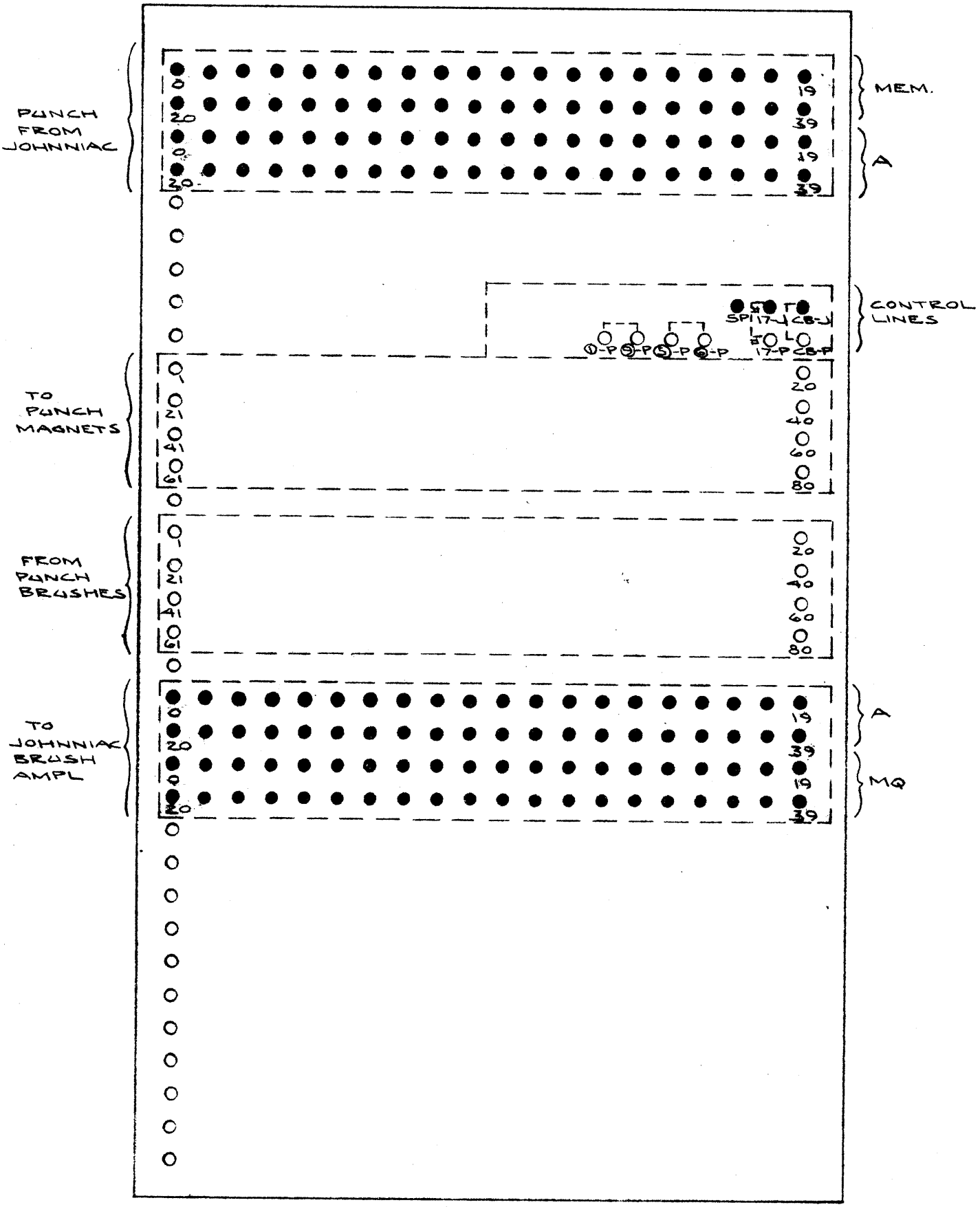
4. Empty hopper and Full Bin Stops cause JOHNNIAC to wait, but do not disconnect the punch from JOHNNIAC. Use the disconnect button to disconnect punch while adding more cards, removing punched cards, etc.

5. JOHNNIAC cannot feed the punch unless the collator primary feed contains cards or is disconnected. No information is lost if this condition is violated. The JOHNNIAC and punch just wait until the condition is corrected.

CARD PUNCH — Approximate Timing



523 PUNCH PLUG BOARD REVISED FOR JOHNNIAC



● = SOLID HUBS TO JOHNNIAC

O = CIRCLE HUBS TO PUNCH

-J = FROM JOHNNIAC

-P = FROM PUNCH

ANelex PRINTER

The printer can print up to 40 columns of decimal information at 900 lines per minute. Operation 10.0 with address part 0004 selects the printer and pulls the paper down one space. Operation 10.0 with address part 0005 selects the printer without spacing the paper. Operation 10.6 causes the printer to restore the page to the next fold when using paper from a flat pack. Decimal information is printed "card image" style. Ten copies are required to print a line. Zeros are printed in positions that have binary 1's in the word specified by the first copy instruction; ones are printed as indicated by the second copy, etc. Note that this image is copied 0, 1, 2, 3 ...9. Cards are punched 9's first.

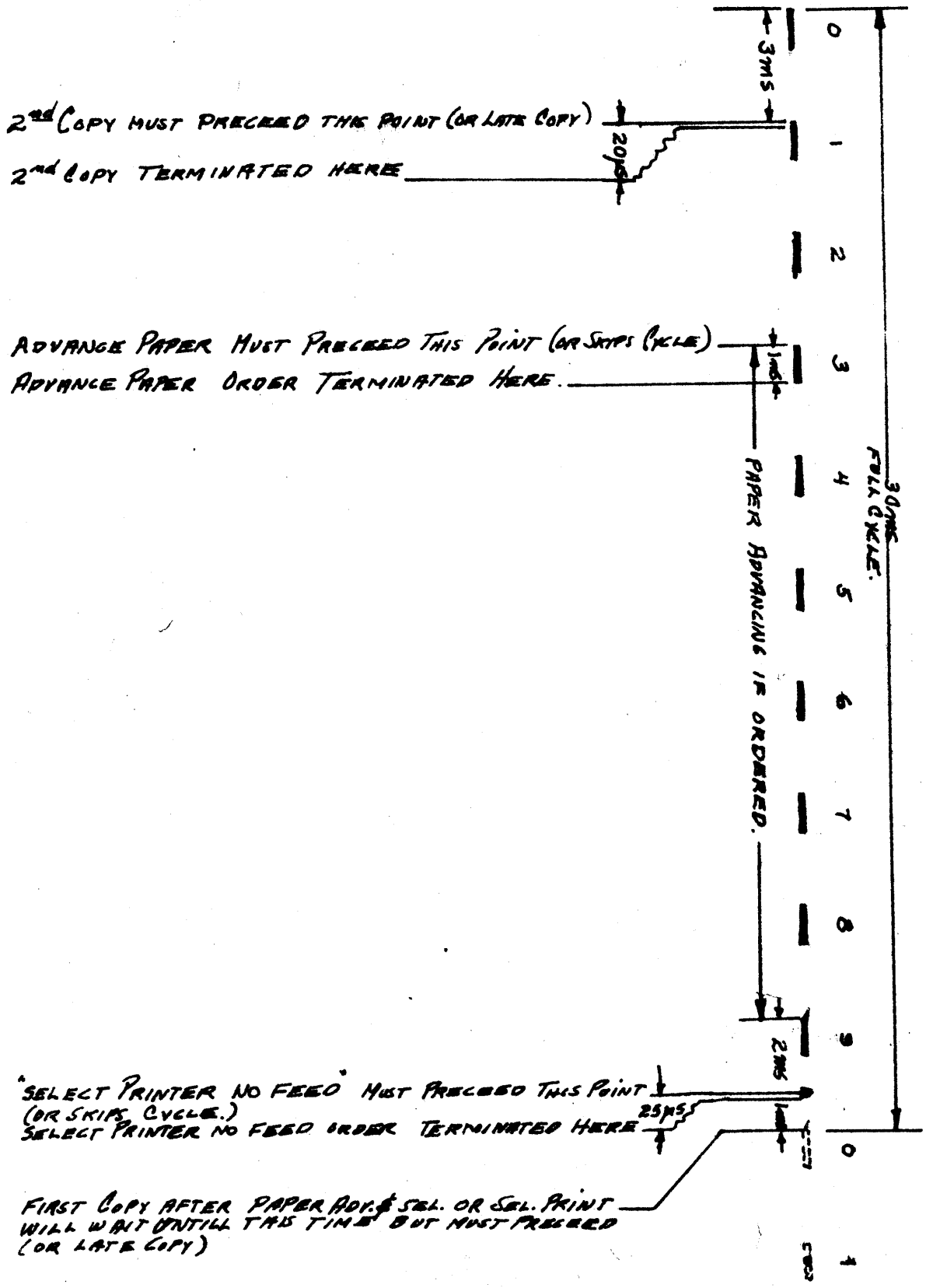
To Print

1. Feed paper down under the drum and paper pullers.

2. Turn on printer. Motor toggle switch is located on inside of printer. (Manual page restore is on front of printer.)

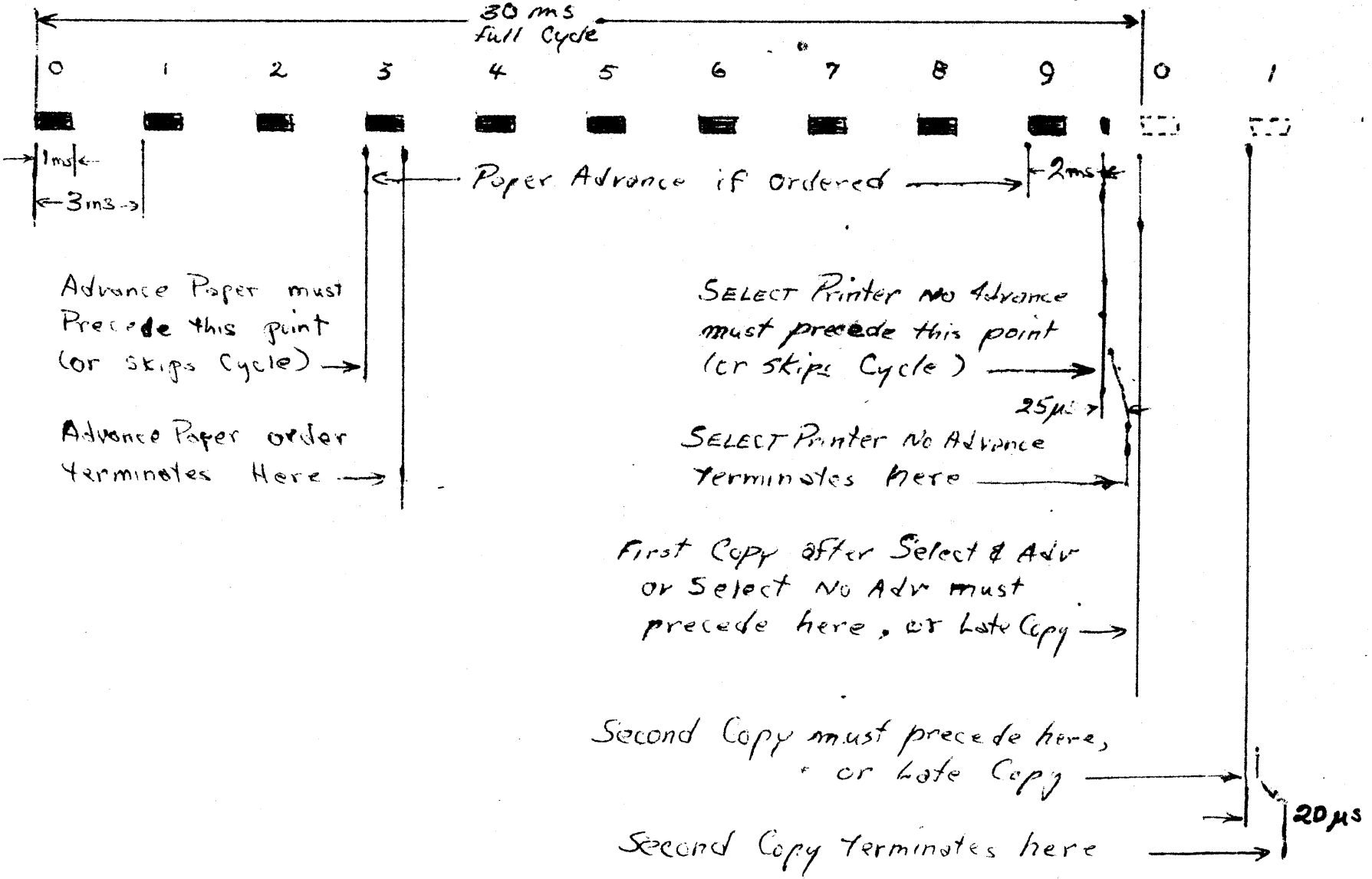
JOHNNIAC will wait for printer when motor switch is off. Printer also requires that collator punch card feeds be full or disconnected.

ANELEX PRINTER TIMING CHART (APPROXIMATE)



9/21/54

ANELEX PRINTER — Approximate Timing



6-21-57