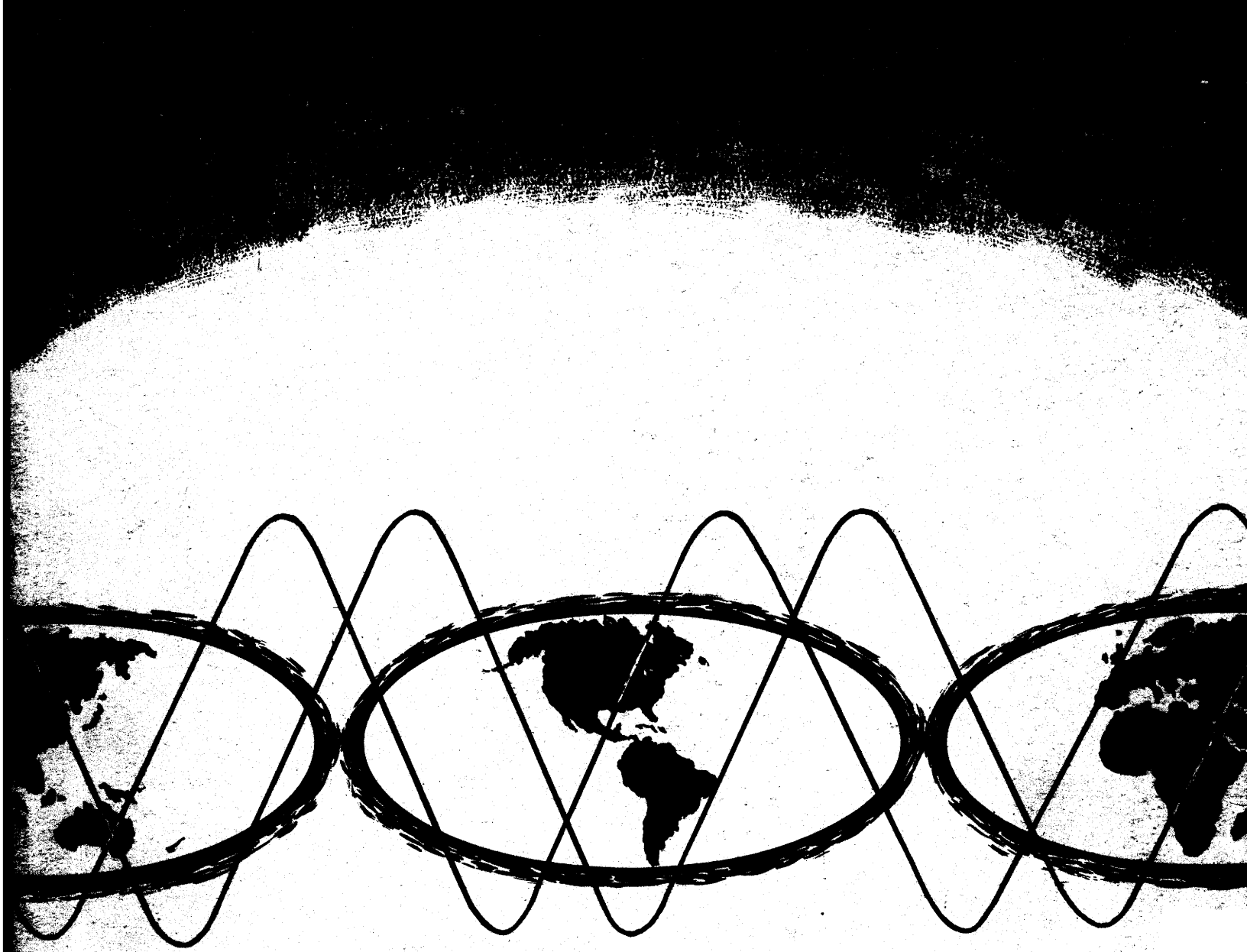
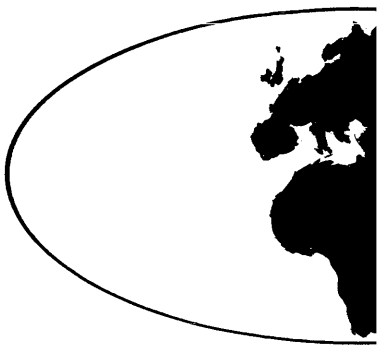


univac® model 1103A computer

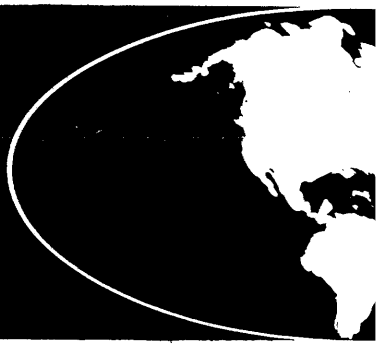
SCIENTIFIC

programming manual

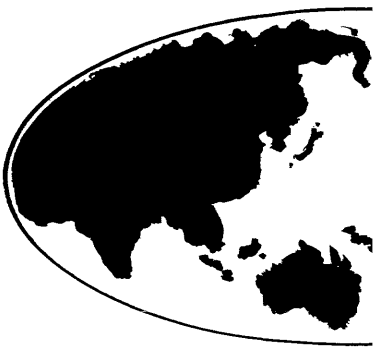


univac®

SCIENTIFIC



1103A



programming manual

ANOTHER SERVICE OF . . .
MANAGEMENT SERVICES AND OPERATIONS RESEARCH DEPARTMENT

Remington Rand Univac
DIVISION OF SPERRY RAND CORPORATION

TABLE OF CONTENTS

SECTION I

SECTION II

SECTION III

SECTION IV

SECTION V

APPENDIX PAGE

section 1

GENERAL DESCRIPTION

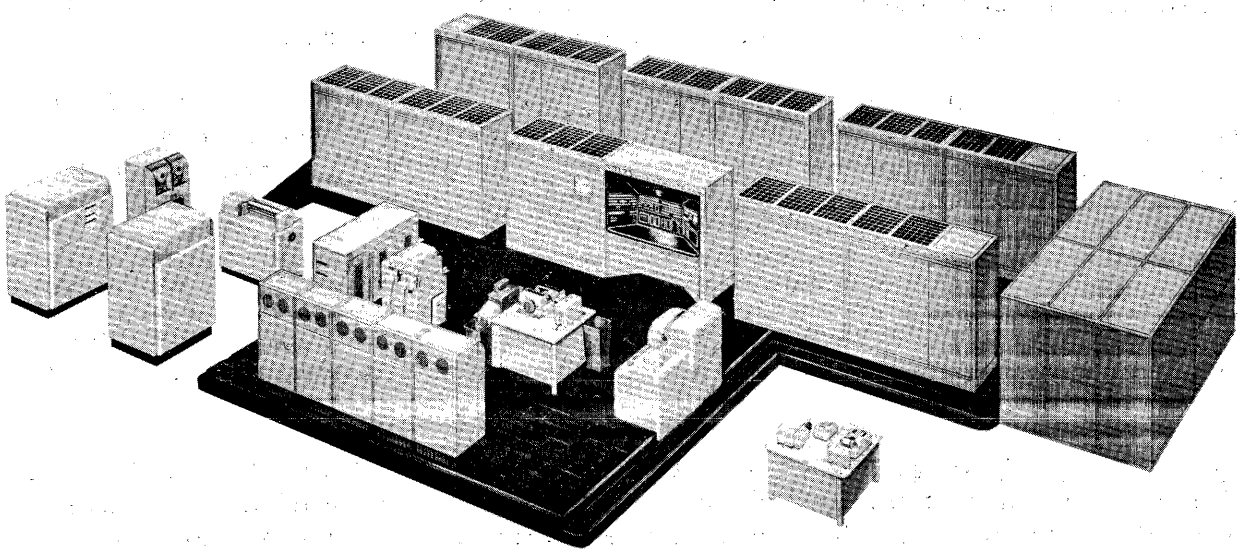


Figure 1-1

1-1. GENERAL

1-2. The Univac Scientific Computer model 1103A (see figure 1-1) is specifically designed for applications requiring great programming versatility, high operating speed, and large storage capacity. Maximum use of the high speed inherent in this computer is permitted by the unusual logical design and its unique Program Interrupt feature. In addition to performing large scale calculations, the system is adaptable to a wide variety of applications including simulation and control in real time.

1-3. Programs of internally stored instructions, capable of self-modification, determine the sequence of operations. Thus, the computing system is fully automatic. Its high speed results from parallel mode operation whereby all digits of a number are operated upon simultaneously.

1-4. DEFINITIONS.

1-5. Discussions of computer systems have led to the establishment of a computer "language." Some of the more common terms are defined in the following paragraphs.

The Univac Scientific utilizes binary notation in its expression of information. A binary digit is termed a bit. An array of binary digits is a word. Words are held in the computer at storage locations or in computer registers.

A single-length Univac Scientific computer word consists of 36 bits. A Univac Scientific computer word may be an instruction, data with numerical value, or data coded in some arbitrary fashion.

A computer instruction consists of an operation code and operand references. The operation code describes to the computer what it is to accomplish. The operand references provide the data for the computer operation.

The Univac Scientific employs two-address logic. Each Univac Scientific instruction has an operand code, a u-address portion, and a v-address portion. The u and v portions usually designate storage locations of operands for the instructions.

Each location which can be directly referenced in an instruction has an individual address. These locations constitute the addressable storage of the computer. The addressable storage in the Univac Scientific is the Magnetic Core, the Magnetic Drum, and two registers in the computer (the Accumulator and the Q Register).

Magnetic Core Storage provides storage locations for a total of 12,288 words. Of these, 4096 are standard with the computer, and the additional 8192 are available optionally. Each of these storage locations has an individual address.

The normally used storage capacity of the Magnetic Drum is 16,384 words, each individually addressed. In addition to this storage space, the Magnetic Drum has another 640 locations which are not normally addressable. The area of these locations is called the Reserve space on the drum.

Additional storage is provided by the Univac Scientific Magnetic Tape System. Data stored on magnetic tape are accessible in blocks of a fixed or variable number of computer words.

1-6. BASIC PRINCIPLES.

1-7. Automatic operation of the computer proceeds under the direction of an internally stored sequence of instructions. Such a sequence of instructions is called a program. The main steps followed in producing a program are: (1) analysis of the problem, (2) flow-charting the problem, (3) coding the problem, and (4) machine-checking the coding (called "debugging" the program).

1-8. The coded program is converted to a form which can be interpreted by external equipment as input to the computer. The input equipment places the program in the computer storage under control of a "loading" program. Certain loading programs may be permanently stored in the computer.

1-9. Once the program has been loaded into the computer, the computer is then automatically set to execute the instruction stored at a fixed location. The location address of any instruction desired to be the first instruction to be executed can be manually inserted into the appropriate register on the computer control panel. When the computer is then started, the desired instruction is extracted automatically from storage and placed in a control register where it is held temporarily during its execution. Under automatic control, each instruction, from each next consecutive address, is extracted from storage and executed. The procedure is continued until (1) an instruction is encountered which specifies the address of the next instruction to be executed, or (2) a program interrupt signal is received, or (3) a computer final stop is effected.

1-10. The Program Interrupt feature of the Univac Scientific provides a method of interrupting the execution of a program. A signal to interrupt a program may originate in a unit of an external equipment, or it may be initiated manually from the computer control panel. An interrupt signal from external equipment indicates that the equipment is ready to receive output from the computer or is ready to send input to the computer.

1-11. Stored instructions, and instructions not internally stored, may be executed under manual control by following a particular procedure at the computer control panel. The computer may be stopped arbitrarily at the computer control panel during either automatic or manual operation.

1-12. The Univac Scientific handles many types of input and output equipment. Included as part of the basic computer system are an Electric Typewriter, a Photoelectric Paper Tape Reader, and a High Speed Paper Tape Punch. Equipments are available to handle on-line communication with punched cards and magnetic tape. Off-line communication between magnetic tape and a variety of Univac peripheral equipment is also possible. The Univac High-Speed Printer produces printed copy at the maximum rate of 600 lines per minute. Additional direct communication is possible with a variety of devices, such as a teletypewriter, an oscilloscope display unit, and analog-to-digital converters.

1-13. CONTROL SECTION

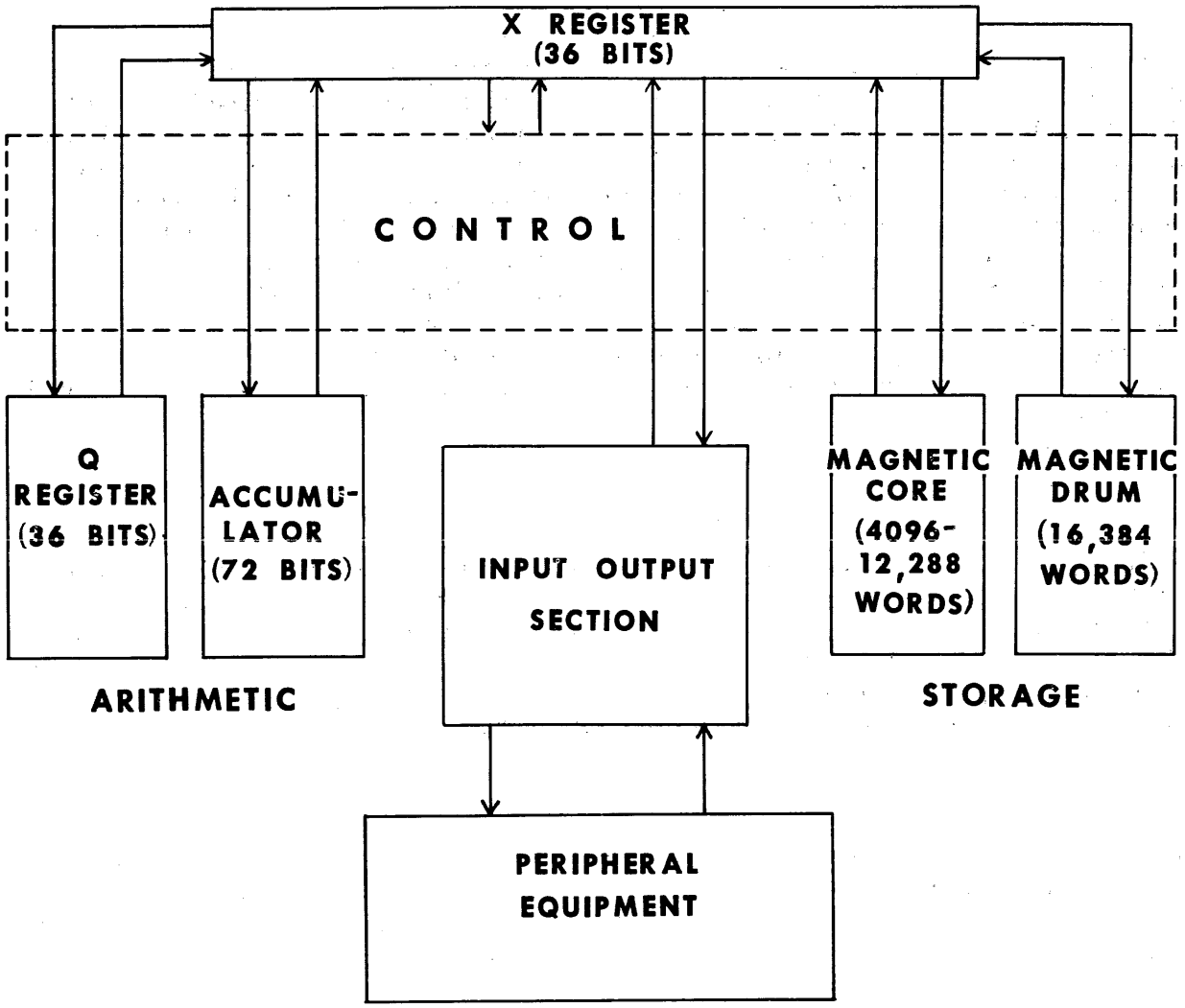


Figure 1-2

1-14. The Control Section of the Univac Scientific directs and regulates internal operations of the computer, and in addition oversees data transmission of the computer (see figure 1-2). Control functions are described in the following paragraphs (1-15 through 1-25).

1-15. The execution of an instruction has two phases: (1) the function of the instruction is accomplished, and (2) the next instruction to be executed is determined and then placed in the Program Control Register (PCR). Each instruction remains in PCR during its execution. The address of the next in-

struction to be executed is usually found in the Program Address Counter (PAK). Transmissions to or from storage of an instruction (or any computer word) are made according to the address held in the Storage Address Register (SAR). A computer word is acquired (from the location specified by the address in SAR) and placed in the X Register. From the X Register, the computer word is directed to another position in the computer according to the current operation. For example, if the computer word in X is the next instruction to be executed, the content of the X Register is transmitted to the Program Control Register. The basic timing control for the above operations is provided by the Master Clock and the Main Pulse Distributor.

1-16. MASTER CLOCK. Computer operations are synchronized by a central timing system called the Master Clock. The Master Clock propagates an electronic clock pulse every two microseconds, and controls the release of these clock pulses to the rest of the computer.

1-17. MAIN PULSE DISTRIBUTOR MPD. Certain pulses released by the Master Clock become Main Pulses. These pulses are denoted in the order of occurrence as Main Pulses zero, one, two, etc., through seven, MP0 ... MP7. The number of Main Pulses initiated depends upon the instruction being executed.

Each Main Pulse controls certain functional sequences of subsidiary pulses. The actions effected by the subsidiary pulses are defined as occurring on the Main Pulse. On MP6 and MP7 the instruction to be executed next is brought from storage to the Program Control Register. On two or more of the pulses MP0 through MP5, the function of the instruction in the Program Control Register is accomplished.

1-18. STORAGE ADDRESS REGISTER SAR. Each time a computer word is to be placed in storage (written), or extracted from storage (read), the address of this word is placed in the Storage Address Register. If writing is specified, the word in the X Register is recorded at the address held in SAR. If reading is specified, the word at the address in SAR is extracted from storage and placed in the X Register. During shifting operations, the rightmost seven bits of SAR are the shift count.

1-19. PROGRAM CONTROL REGISTER PCR. Each instruction is received from storage and held temporarily during its execution in the Program Control Register. This register consists of a Main Control Register MCR, of six bits; a U-Address Counter UAK, of 15 bits; and a V-Address Counter VAK, of 15 bits. These registers hold the v-address portion of the instruction.

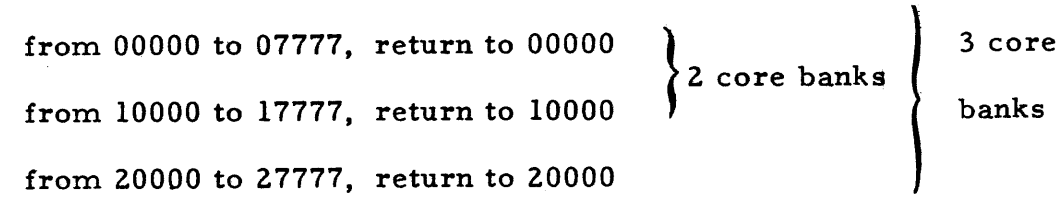
1-20. On Main Pulse Zero, MPO, the execution of the instruction in the Program Control Register is initiated. This assumes that the computer word in PCR is an instruction with a legal operation code. If the bits in the Main Control Register on MPO are detected to be a non-existent "operation code," a MCT (Main Control Translator) computer fault occurs, which stops computer oper-

ation immediately. Indication of the MCT fault is given on the computer control panel. The structure of the Program Control Register is of concern when the Repeat instruction is used. (The Repeat instruction directs the computer to repeat the instruction at the next consecutive address a prescribed number of times.) The instruction to be repeated is acquired from storage only once. The instruction being repeated remains in the Program Control Register until the termination phase of the Repeat operation. The u and v addresses of the repeated instruction are optionally advanced in UAK and VAK. Thus, the form of the instruction is altered in PCR, and not in storage.

1-21. According to the storage class, the advancement of addresses in UAK and VAK is as follows:

Magnetic Core Addresses	
computer with <u>one</u> core bank	advance from 00000 to 07777, return to 00000
computer with <u>two</u> core banks	advance from 00000 to 07777 to 10000 to 17777, return to 00000
computer with <u>three</u> core banks	advance from 00000 to 07777 to 10000 to 17777, to 20000 to 27777, return to 00000
Q Register address	advance from 31000 to 31777, return to 31000
Accumulator addresses	advance from 32000 to 32777, return to 32000 advance from 33000 to 33777, return to 33000 advance from 34000 to 34777, return to 34000 advance from 35000 to 35777, return to 35000 advance from 36000 to 36777, return to 36000 advance from 37000 to 37777, return to 37000
Magnetic Drum addresses	advance from 40000 to 77777, return to 40000

1-22. The option of cycling within <u>each</u> core bank, if more than one is provided, is also possible. This is usually done, however, only for maintenance and testing purposes. By setting a switch on the computer control panel, the advancement of addresses in UAK and VAK is as follows:



1-23. PROGRAM ADDRESS COUNTER PAK. Consecutive addresses are generated automatically in the Program Address Counter. During the sequen-

tial execution of consecutive instructions, the termination phase of each of these instructions places the next consecutive instruction in the Program Control Register. The following occurs on MP6: (1) the address in the Program Address Counter is transmitted to the Storage Address Register. (2) The content of PAK is now advanced by one. (3) The computer word at the address specified by SAR is transmitted to the X Register. Then, on MP7, the content of the X Register is transmitted to the Program Control Register.

According to the storage class, the automatic advancement of addresses in PAK is as follows:

Magnetic Core Addresses	advance as in UAK and VAK as detailed in the discussion of the Program Control Register. (The option of cycling within <u>each</u> core bank applies here also.)
Q Register Addresses	advance from 31000 to 31777 to 32000, resulting in an SCC fault
Accumulator Addresses	results in an immediate SCC fault
Magnetic Drum Addresses	advance from 40000 to 77777, return to 40000

The Storage Class Control SCC computer fault occurs when an Accumulator address is specified as the address of the next instruction. Computer operation is stopped immediately. The fault is detected when the address in SAR, specifying the instruction to be read, is found to be an Accumulator address.

1-24. The address of the next instruction to be executed, as taken from the Program Address Counter, is not always the address of the next consecutive instruction. Another address may be inserted in PAK during the execution of the current instruction. Then, on MP6 and MP7, the instruction at this address is transmitted from storage to the Program Control Register. An operation such as this effects a "jump."

The address of the next instruction to be executed is not always acquired from PAK. During the termination phase of a Repeat operation, a predetermined address is inserted directly into the Storage Address Register <u>if</u> a "jump" was not effected during the Repeat operation. Then, on MP6 and MP7, the instruction at the predetermined address is transmitted to PCR.

1-25. When computer operation is initially started, the Program Address Counter is automatically set to address 40000 and the Main Pulse Distributor is set to MP6. Then, unless PAK and MPD are manually altered at the computer control panel, the first instruction is acquired from address 40000. Address 40000 can thus be considered the starting address of a program if an appropriate instruction is stored there.

1-26. PRINCIPAL REGISTERS

1-27. The Accumulator, the Q Register, and the X Register are referred to as arithmetic registers but serve several purposes. The Accumulator and the Q Register can be referenced by the programmer since they are both addressable.

1-28. X REGISTER. _ The X Register is a 36-bit non-addressable register. As an exchange register, X temporarily holds words in their transmission from one computer location to another. As an arithmetic register, X holds the addend, subtrahend, multiplicand, and divisor in the corresponding arithmetic operations.

1-29. Q REGISTER. _ The Q Register is a 36-bit register. As an arithmetic register, Q holds the multiplier and the quotient in the corresponding arithmetic operations, and the logical multiplier. As an addressable storage register, Q provides temporary storage for a single 36-bit word. Any of the addresses 31000 through 31777* may be used to reference the Q Register.

1-30. ACCUMULATOR. The Accumulator A is a 72-bit register (referred to as a double- length register). The right-hand 36 bits of the Accumulator is termed the content of A_R; the left-most 36 bits of the Accumulator is termed the content of A_L. As an arithmetic register, the Accumulator holds the sum, difference, product, and dividend (and final remainder) in the corresponding arithmetic operations. As an addressable storage register, the Accumulator provides temporary storage for a 72-bit word. Any of the addresses 32000 through 37777* may be used to reference the Accumulator.

1-31. STORAGE

1-32. MAGNETIC CORE STORAGE, MCS.

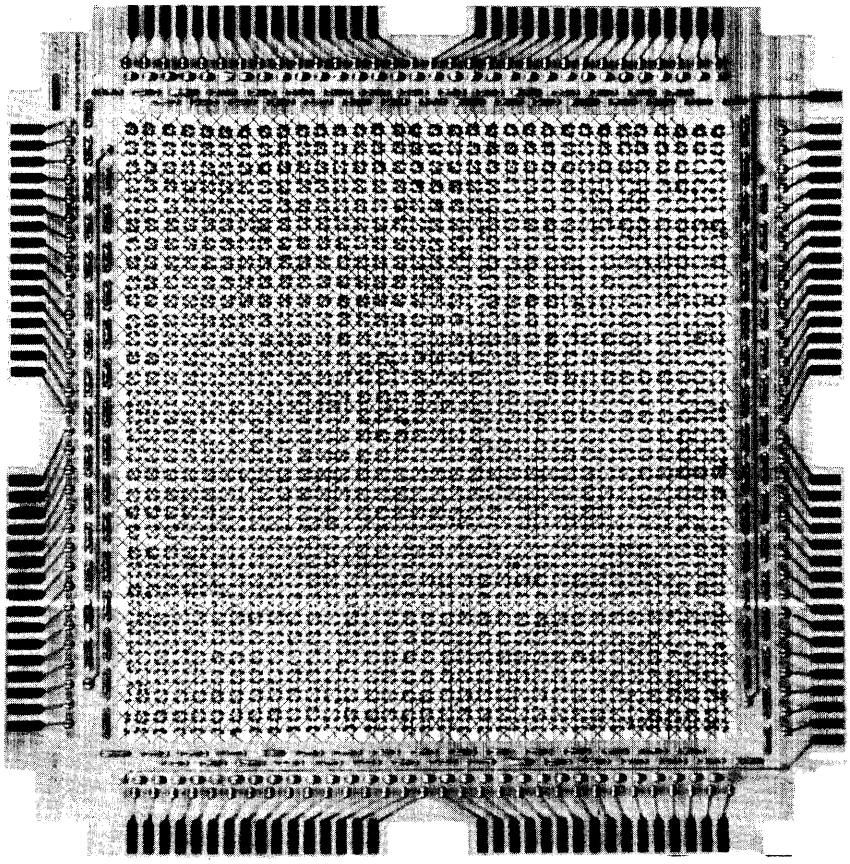


Figure 1-3

*Note. The address of a storage location is a 15-bit number, represented in octal code by five digits. Each octal digit represents three binary digits. Octal notation is used in this text for denoting specific storage addresses.

1-33. Magnetic Core Storage has the advantage of providing minimum access time to stored information. Words in core storage can be extracted or recorded in as few as eight microseconds. Each bank of Magnetic Core Storage is capable of storing 4096 computer words. Up to three banks may be obtained with the computer. Each bank consists of a stack of 36 64x64 core matrices (see figure 1-3). At the intersection of each row and column of a matrix is a core which may be magnetized in a "0" or "1" direction. A 36-bit computer word is stored by properly magnetizing a core in the same matrix position in each of the 36 matrices. This is illustrated in figure 1-4. The addresses assigned to the core locations are x0000 through x7777, where x is 0, 1, or 2, depending upon the particular bank referenced.

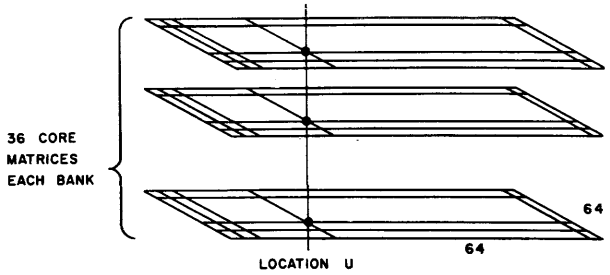


Figure 1-4

1-34. Since magnetic core storage provides rapid access storage, a program which is ready for execution plus the associated data are usually located in core storage. The automatic acquisition of instructions from consecutive addresses in magnetic core storage proceeds from one core bank to another where more than one core bank is provided. This is continued until the execution of the instruction at the last storage address of the available magnetic core storage. Then, unless the last instruction specifies a jump, the next instruction to be executed is acquired from the first address of core storage. The sequential procession is then resumed with instructions being obtained from consecutive addresses in the first core bank.

1-35. MAGNETIC DRUM STORAGE MD.

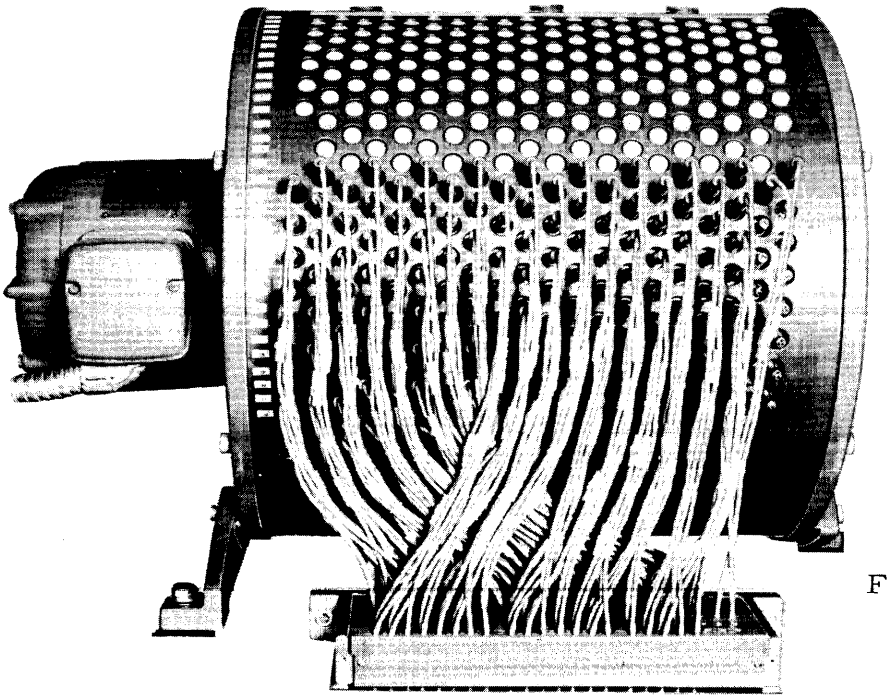


Figure 1-5

1-36. The access time for words in Magnetic Drum Storage is slower than the access time for words in Magnetic Core Storage. The Magnetic Drum is valuable for the storage of "blocks" of instructions (or data) which are to be transferred to core storage prior to their execution (or processing). A transfer rate of 31, 250 words per second is possible between the drum and core storage.

1-37. DESCRIPTION. The Magnetic Drum (see figure 1-5) contains a continuously rotating cylinder surfaced with a material that can be magnetized locally. Information is stored as magnetized spots on specific areas of the drum through recording heads attached around the outside of the drum case. The storage areas normally used provide addressable storage for a total of 16, 384 36-bit computer words. A word can be extracted from or recorded at a given location only once during a drum revolution. This results in a maximum access time of 34 milliseconds, the time consumed by a complete drum revolution. The most efficient use of drum storage is achieved by storing words at locations which are in position for reading or writing at the time the locations are referenced (this procedure is called minimum access coding.)

1-38. The addresses of the drum locations are divided into four groups, each of which has 4096 normally used locations addressable as x0000 through x7777, where x is 4, 5, 6, or 7. The sequential acquisition of instructions continues from 40000...47777 to 50000...57777 to 60000...etc...77777 to 40000...

1-39. VARIABLE INTERLACE SYSTEM. Each addressable storage location on the drum is "marked" according to a recorded timing track. An index position is recorded on a separate track. As the drum revolves, an Angular Index Counter, AIK, is advanced by the recorded timing track. As each drum location is in position for reading or writing, its address is momentarily in the Angular Index Counter. The Angular Index Counter is advanced from the index position through 4095 positions, or from x0000 to x7777, after which AIK is cleared to zero. These are the addresses of the normal drum storage locations. The advancement of AIK is resumed then from zero forward until the index position is reached, at which time AIK is again cleared. The addresses in AIK during this advancement are the addresses of the reserve space locations.

The time which elapses between the positioning of adjacent drum locations for reading or writing is approximately eight microseconds. Consecutive reading and/or writing in adjacent drum locations cannot be completed in this short a time. For example, if adjacent drum locations are chosen for successive reading or writing, a complete drum revolution would occur before the second read or write could be accomplished. A variable interlace system, which eliminates this problem, effects the assignment of consecutive addresses to drum locations which are non-adjacent but are spaced at fixed intervals; the fixed intervals being determined by the interlace selected. For example, an interlace of four, effects the assignment of consecutive addresses to every fourth drum location, and provides a time lapse of 32 microseconds between

the proper positioning of consecutively addressed drum locations. This allows the computer a maximum of 32 microseconds to complete one drum reference. Figure 1-6 illustrates the selection of drum locations with a four interlace (the layout of locations as illustrated does not comply strictly with the actual arrangement on the drum). The interlace is manually selected and is indicated by a light on the computer control panel. Interlaces 4, 8, 16, 32, and 64 are available.

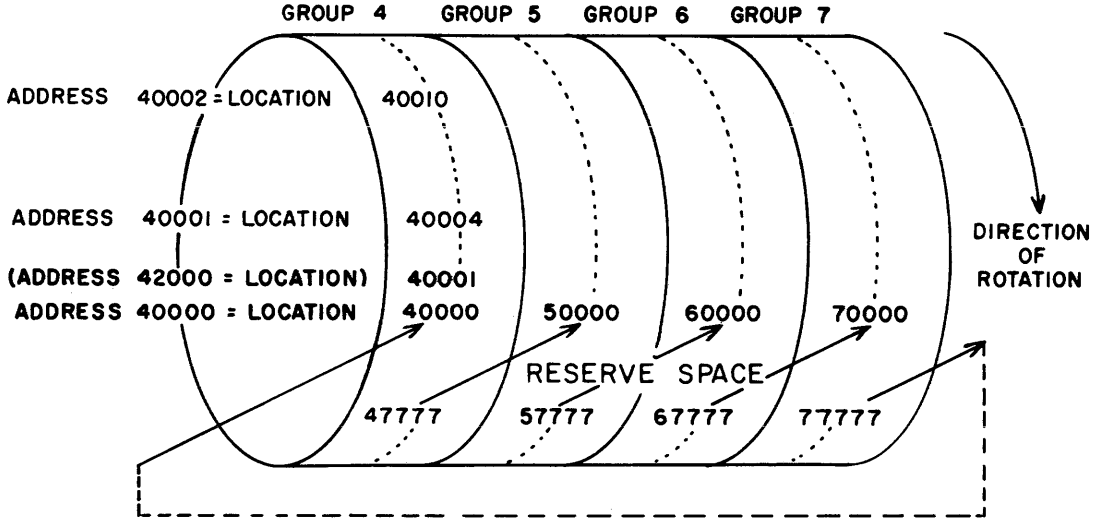


Figure 1-6

1-40. A coded address reference is converted to an "interlaced" address as shown in figure 1-7. This illustrates the conversion for a four interlace.

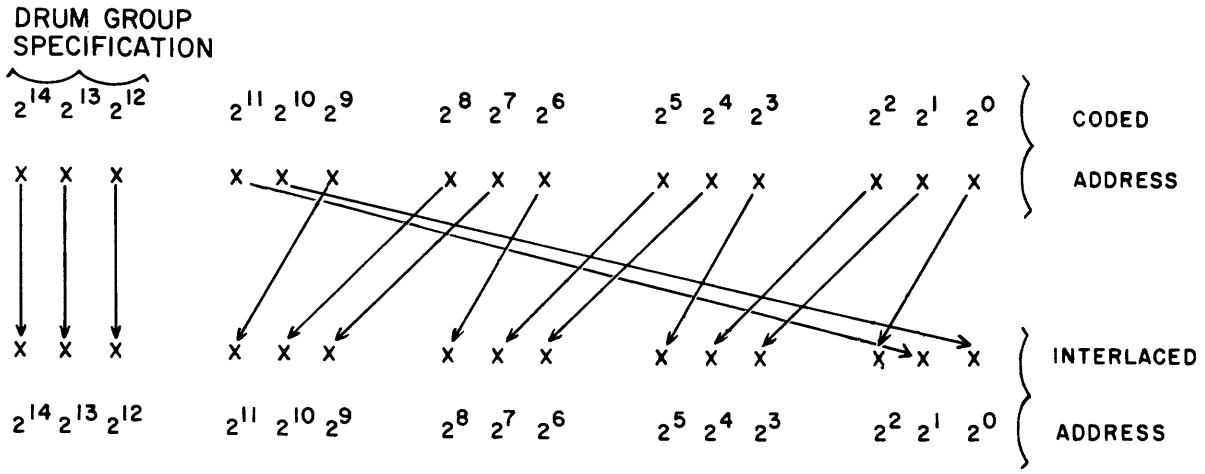


Figure 1-7

The connection lines for each interlace are from position 2^0 to position $(2^0 x 2^m)$, from position 2^1 to position $(2^1 x 2^m)$, etc., where 2^m equals the interlace expressed as a power of two.

1-41. A check is made for coincidence between the interlaced address and the address momentarily in the Angular Index Counter, and when coincidence occurs, a read or write operation occurs. During "Normal drum" operation, coincidence checks occur only when the Angular Index Counter is advancing through the normal drum locations (addresses x0000 through x7777).

1-42. RESERVE SPACE. Each drum group, in addition to the previously mentioned 4096 addressable locations, has other locations which are not normally addressable. The area of these locations is called the Reserve space. The area shown in figure 1-6 is between locations 47777 and 40000, 57777 and 50000, etc. Communication is established with these locations and broken with the rest of the drum by setting the NORMAL/ABNORMAL DRUM switch on the computer control panel to its abnormal setting. During the "Abnormal drum" operation, coincidence checks occur only when the Angular Index Counter is advancing through the Reserve space locations, 0 and forward, up to the index position. If an interlaced address is greater than the number of locations in the Reserve space, coincidence will not be achieved. A program can safely reference for reading no more than the known minimum of Reserve space locations. The practical known minimum is 160 per group, octal addresses 0000 through 0237, for each drum group 4, 5, 6, or 7. If more than this number of locations are known to be available, they can of course be referenced.

1-43. According to the interlace, the following number of drum locations can be referenced for each group during each drum revolution. This assumes that a total of 160 locations are available per drum group. The addresses which are coded to reference reserve space locations are listed in the appendix.

With a four interlace - 40 locations (for each of 4 revolutions)
With an eight interlace - 20 locations (for each of 8 revolutions)
With a 16 interlace - 10 locations (for each of 16 revolutions)
With a 32 interlace - 5 locations (for each of 32 revolutions)
With a 64 interlace - 3 locations for each of 32 revolutions and
2 locations for each of remaining 32 revolutions

The matter of addressing the reserve space on the drum should not ordinarily concern a programmer, since writing on the reserve space is accomplished only by maintenance procedures. The interlace and storage locations of routines stored on the reserve space are fixed and known for future reading after this initial writing. Normal usage of the reserve space is for safe storage of routines needed to load programs into the computer. Once transcribed on the drum, these routines are always available since it is not possible to destroy them by writing over them.

1-44. ADDRESSABLE STORAGE.

1-45. Following is a summary of the addresses assigned to the storage classes.

Addresses in Octal Notation	Class	
00000-07777	MCS-0	4096 36-bit words
10000-17777	MCS-1	4096 36-bit words (optional)
20000-27777	MCS-2	4096 36-bit words (optional)
31000-31777	Q Register	1 36-bit word
32000-37777	Accumulator	1 72-bit word
40000-77777	MD	16,384 36-bit words

1-46. Several locations in storage are automatically referenced during computer operation. Because of this fact, these locations are usually reserved by the programmer. The addresses of these locations are termed fixed addresses and are as follows.

fixed address F_0	the location addressed	as 40000
fixed address F_1	the location addressed as 00000	or 40001
fixed address F_2	the location addressed as 00001	
fixed address F_3	the location addressed as 00002	
fixed address F_4	the location addressed as 00003	

Fixed address F_1 may be assigned to the drum location addressed as 40001 by a manual selection on the computer control panel. The use of some of these addresses by certain of the computer instructions is pointed out in the section Instruction Repertoire.

1-47. The addresses listed subsequently are considered illegal under the circumstances noted. If reference to these locations is made, a computer fault occurs which immediately stops computer operation. The fault is indicated on the computer control panel as an SCC (Storage Class Control) computer fault.

30000-30777	Unassigned addresses	Illegal under all circumstances
10000-17777 20000-27777	MCS-1 MCS-2	Illegal if these banks of core storage are not provided
32000-37777	A	Illegal if any of these addresses are specified as the address of the next instruction.
31000-31777 32000-37777	Q A	Illegal if a partial replacement of the content of these locations is attempted. (This is clarified in the presentation of the instructions.)

40000-77777	MD	Illegal if an External Read instruction tries to place data (from external equipment) in drum storage; or if an External Write instruction tries to extract data (for external presentation) from drum storage.

1-48. MAGNETIC TAPE STORAGE.

Additional storage for the computer is provided by the use of the Magnetic Tape System. A complete discussion of the tape system can be found in the section describing input and output equipment. The Magnetic Tape System comprises a number of tape transporting devices known as Uniservos, and a control section which is located in the computer structure. This system provides for the transfer of information between the computer and a removable reel of tape on a Uniservo. A maximum of 10 functional Uniservos can be used with the tape system. Information is transferred (read from the tape or written on the tape) under program control. A reel of magnetic tape is shown in figure 1-8. Any reel of tape 1500 feet or less is acceptable for use on the Uniservos.

Figure 1-8

1-49. Information is recorded as magnetized areas in a "line" across the width of a tape. Six data bits are recorded in each line along with a check bit and a timing bit. Lines are recorded at a density of 128 or 50 per inch.

1-50. Information is recorded in one of three formats. These are Fixed Block Length Recording, Variable Block Length Recording, and Continuous Data Input Recording. The modes of operation for recording in either the Variable Block Length or Continuous Data Input are optional with the tape system.

1-51. In the Fixed Block Length Recording mode, information is recorded in block lengths of 120 computer words (720 lines). A blockette consists of 20 computer words (120 lines). Either of two unrecorded spaces exist between blocks. The space between blockettes, if a space is desired, also can be either of two lengths. The block and blockette spacing are chosen according to the use to be made of the information. Off-line processing of data is possible by using the recorded reel of tape on a variety of Univac peripheral equipments. Information can be both recorded and read in fixed block lengths. Based on a space of one inch between blocks of fixed length, and no space between blockettes, a maximum of approximately 326, 000 computer words can be stored on a 1500 foot reel of tape. The maximum transfer rate of information between the computer and the tapes is approximately 1800 computer words per second.

1-52. In the Variable Block Length Recording mode, information is recorded with a variable number of words in a block. Blocks of information are separated by unrecorded areas on the tape. Information can be recorded and read in variable block lengths at a maximum transfer rate of approximately 2100 computer words per second.

1-53. The Continuous Data Input mode reads information which has been recorded continuously on the tape. There is no "block" limitation except for the length of the tape. This form of recording is useful for real time observations which will not permit interruptions to format the information in fixed or variable block lengths.

1-54 STATED POINT COMPUTER ARITHMETIC

1-55 NUMBER NOTATION.

A quantity in the Univac Scientific carries a positive or negative value connotation according to its representation in one's complement notation. The leftmost bit of the quantity in one's complement notation is indicative of the sign of the number represented. If the number represented is positive, the sign bit is zero. If the number represented is negative, the sign bit is one. A number negative in value appears in the computer as the one's complement of its absolute value. Thus, each zero in the representation of the absolute value becomes a one, and each one becomes a zero. For example, the representation of decimal (+13) as a computer word is

000 000 000 000 000 000 000 000 000 000 001 101 .

Since this is also the representation of the absolute value of decimal (-13), the representation of decimal (-13) is the one's complement of this, or

111 111 111 111 111 111 111 111 111 111 110 010*

Any bits between the sign bit and the most significant bit of the number represented are also indicative of the sign of the number. Thus, the most significant bit of a quantity is the first bit to the right of the sign bit which differs from the sign bit. In the example above, note that each bit to the left of the most significant bit appears the same as the sign bit. This is less obvious when a number is represented in octal notation. The octal representation of decimal (+13) is 000000000015. The octal representation of decimal (-13) is 777777777762.

1-56. In stated point arithmetic operations, the natural arithmetic of the computer, the machine treats all quantities as integers. However, the integral capacity of the registers does not limit arithmetic operations to dealing only with numbers in this integral range. Numbers outside the integral capacity of a 36-bit register may be handled by programming appropriate binary scaling or by use of the instructions in the optional floating point arithmetic package. The latter provides automatic treatment of 36-bit signed quantities of the form, $x \cdot 2^y$.

1-57. The quantity zero has a positive and a negative representation in the computer, namely, in octal notation, 000000000000 or 777777777777. However, it is not possible for the representation of "negative" zero, 777777777777, to be generated as the result of an arithmetic operation. A "negative" zero, generated in some other way, is treated as the quantity zero in arithmetic operations.

1-58. In an n-bit register, no more than n-1 bits can represent the value of a signed integer since at least one bit represents the sign. Thus, the range of signed integers which can be represented in an n-bit register is $-(2^{n-1}-1)$ to $+(2^{n-1}-1)$. Hence, the positive integers which can be represented in a 36-bit register lie in the range 1 to $2^{35}-1$, or as expressed in octal notation, 000000-000001 through 377777777777. The negative integers which can appear in a 36-bit register lie in the range -1 to $-(2^{35}-1)$, or as expressed in octal notation, 777777777776 through 400000000000.

The range of numbers possible to represent in a 36-bit register can be illustrated as follows:

* The 36 bits of a computer word are grouped in this fashion to promote ease of reading, and converting to octal notation.

DECIMAL	BINARY
$34,359,738,367 = (2^{35}-1)$	011 111 111 111 111 111 111 111 111 111 111 111
$34,359,738,366 = (2^{35}-2)$	011 111 111 111 111 111 111 111 111 111 111 110
. .	.
. .	.
. .	.
$9 = (2^3+1)$	000 000 000 000 000 000 000 000 000 000 001 001
$8 = 2^3$	000 000 000 000 000 000 000 000 000 000 001 000
$7 = (2^3-1)$	000 000 000 000 000 000 000 000 000 000 000 111
$6 = (2^3-2)$	000 000 000 000 000 000 000 000 000 000 000 110
$5 = (2^3-3)$	000 000 000 000 000 000 000 000 000 000 000 101
$4 = 2^2$	000 000 000 000 000 000 000 000 000 000 000 100
$3 = (2^2-1)$	000 000 000 000 000 000 000 000 000 000 000 011
$2 = 2^1$	000 000 000 000 000 000 000 000 000 000 000 010
$1 = 2^0$	000 000 000 000 000 000 000 000 000 000 000 001
$0 =$	000 000 000 000 000 000 000 000 000 000 000 000
$-1 = -2^0$	111 111 111 111 111 111 111 111 111 111 111 110
$-2 = -2^1$	111 111 111 111 111 111 111 111 111 111 111 101
$-3 = -(2^2-1)$	111 111 111 111 111 111 111 111 111 111 111 100
. .	.
. .	.
. .	.
$-34,359,738,366 = -(2^{35}-2)$	100 000 000 000 000 000 000 000 000 000 000 001
$-34,359,738,367 = -(2^{35}-1)$	100 000 000 000 000 000 000 000 000 000 000 000

1-59. Basically, arithmetic operations involve the acquisition of operands from storage, the manipulation of these operands in the X Register, Q Register, and/or the Accumulator, and the storage (usually) of the derived results. When any 36-bit register, addressed as u, has its content, (u), transmitted to the 72-bit Accumulator, 36 additional bits are assumed to the left of the left-most sign bit of (u). This 72-bit word is termed a double-length extension of (u). The extended word is denoted as D(u) if the 36 bits to the left of the sign bit are the same as the sign bit. The 72-bit word D(u) retains in all cases the same numerical value as the 36-bit word (u). The extended word is denoted as a Split extension, S(u), if the 36 bits to the left of the left-most sign bit are all zeros. The value of the sign bit of (u) is disregarded for the Split extensions. Thus, for positive numbers only, S(u) has the value of (u) and D(u).

The following examples illustrate double-length extensions.

if (u) =	000 000 000 000 000 000 000 000 000 000 000 101	= decimal (+5)
*D(u) = 000 000...000	000 000 000 000 000 000 000 000 000 000 000 101	= decimal (+5)
S(u) = 000 000...000	000 000 000 000 000 000 000 000 000 000 000 101	= decimal (+5)

* The 72 bit content of the Accumulator is illustrated in this manner in the remainder of this text. The 36 bits of A_R are shown in full; the 36 bits of A_L are indicated as in this example.

if (u) = 111 111 111 111 111 111 111 111 111 111 111 010 = decimal (-5)

D(u) = 111 111...111 111 111 111 111 111 111 111 111 111 111 010 = decimal (-5)

S(u) = 000 000...000 111 111 111 111 111 111 111 111 111 111 010 = decimal $(2^{36}-6)$

Split extensions are useful in multiple precision arithmetic where a single number may be stored in n registers. In this case, the digit in the sign bit position may be used to represent a significant bit in n-1 of the n registers.

1-60. SHIFTING.

1-61. The Accumulator and the Q Register both have left circular, or end-around, shifting facilities. Bits shifted off the left end of these registers appear in order in the right-hand side. Multiplication of a number by 2^k is, accomplished by shifting the number left k places. Note: a shift cannot be considered as a multiplication by 2^k if the most significant digit of the number is shifted into (or through and end-around) the sign bit position. Such a shift of the most significant bit of a number into the sign-bit position of a register changes the value attached to the number from positive to negative, or from negative to positive.

The following examples illustrate shifting, in the Q Register, a number from register u.

(u) = 000 000 000 000 000 000 000 000 000 000 000 101 = decimal (+5)

initial content of Q,
$(Q)_i$ = 000 000 000 000 000 000 000 000 000 000 000 101 = (2^2+1)

after a left shift of 32 places, $(Q)_i \cdot 2^{32}$ = ... = $(2^2+1)2^{32} = 2^{34}+2^{32}$
(Q) = 010 100 000 000 000 000 000 000 000 000 000 000 = $2^{34}+2^{32}$

after a left shift of 33 places, $(Q)_i \cdot 2^{33}$ = ... = $(2^2+1)2^{33} = 2^{35}+2^{33}$
(Q) = 101 000 000 000 000 000 000 000 000 000 000 000 = $-(2^{34}+2^{33}-1)$

after a left shift of 34 places, $(Q)_i \cdot 2^{34}$ = ... = $(2^2+1)2^{34} = 2^{36}+2^{34}$
(Q) = 010 000 000 000 000 000 000 000 000 000 000 001 = $2^{34}+1$

(u) 111 111 111 111 111 111 111 111 111 111 111 010 = decimal (-5)

initial content of Q,
$(Q)_i$ = 111 111 111 111 111 111 111 111 111 111 111 010 = $-(2^2+1)$

after a left shift of 32 places, $(Q)_i \cdot 2^{32}$ = ... = $-(2^2+1)2^{32} = -(2^{34}+2^{32})$
(Q) = 101 011 111 111 111 111 111 111 111 111 111 111 = $-(2^{34}+2^{32})$

after a left shift of 33 places, $(Q)_i \cdot 2^{33}$ = ... = $-(2^2+1)2^{33} = -(2^{35}+2^{33})$
(Q) = 010 111 111 111 111 111 111 111 111 111 111 111 = $+(2^{34}+2^{33}-1)$

after a left shift of 34 places, $(Q)_i \cdot 2^{34} = ... = -(2^2+1)2^{34} = -(2^{36}+2^{34})$
$(Q) = 101\ 111\ 111\ 111\ 111\ 111\ 111\ 111\ 111\ 111\ 111\ 110 = (2^{34}+1)$

The instructions which order left shifts in the Accumulator or Q Register allow the programmer to designate a shift count k in the range $2^7 > k \geq 0$. A left shift of k places is equivalent to a right circular shift of 36-k places in Q or 72-k places in A.

1-62. ADDITION AND SUBTRACTION.

1-63. For the computer process of addition or subtraction, the addend or subtrahend is placed in the X Register. Then, if a subtraction is desired, either D(X) or S(X) is subtracted from the content of the Accumulator. If an addition is desired, the complement of D(X) or S(X) is subtracted from the content of the Accumulator. The subtractive and end-around borrow properties of the Accumulator eliminate the possibility of the generation of a negative zero representation during arithmetic operations. The subtraction process necessitiates an ability of the machine to perform an end-around borrow in the Accumulator, i.e., apply a borrow propagated past the bit A_{71} to the bit A_0. This process is illustrated in the following example, in which a 72-bit Accumulator and a 36-bit X Register are used.

initial content of A,

$(A)_i$ = 000 000...000 000 000 000 000 000 000 000 000 000 000 000 **110** = (+6)

minus
D(X) = 000 000...000 000 000 000 000 000 000 000 000 000 000 000 111 = -(+7)

111 111...111 111 111 111 111 111 111 111 111 111 111 111
end-around borrow of 1

$(A)_f$ = 111 111...111 111 111 111 111 111 111 111 111 111 111 110 = -(+1)
= final content of A

Essentially the Accumulator subtracts modulo $(2^{71}-1)$. For example, if the value of one is added to the largest positive number possible in the Accumulator, the result is the machine representation of the largest possible negative number in A. Care must be exercised in the accumulation of sums in order that neither the positive nor negative capacity of the Accumulator is exceeded.

1-64. MULTIPLICATION.

1-65. For the computer process of multiplication, the multiplier is placed in the Q Register and the multiplicand is placed in the X Register. The multiplication process forms the product in the Accumulator. The product is formed by adding the multiplicand, D(X), the appropriate number of times to the properly positioned content of the Accumulator. An addition of D(X) to (A) is performed for each bit of one in the multiplier, (Q). The process automatically accomodates a negative multiplier. Computer multiplication, using a 72-bit

Accumulator and a 36-bit X Register and Q Register, is illustrated in the following example.

D(X)= 000 000...000 010 000 000 000 000 000 000 000 000 000 000 000 000 $=2^{34}$

(Q)= 000 000 000 000 000 000 000 000 000 000 001 010 $=2^3+2^1$

(A)= 000 000...010 100 000 000 000 000 000 000 000 000 000 000 000 $=2^{37}+2^{35}$
= product

Imminent overflow is detected during an accumulative multiplication process. A check is made to determine if the addition of a product to the content of the Accumulator might cause an overflow of the sum into A_{71}. If this possibility exists, the accumulative multiplication operation is restrained by a stop of computer operation. An Overflow computer fault is indicated on the computer control panel.

1-66. DIVISION

1-67. For the computer process of division, the divisor is placed in the X Register and the dividend is assumed to be in the Accumulator. The registers are employed in such a manner that

$$(A)_i = (X) \cdot (Q) + R, \qquad \text{where } 0 \leq R < |(X)|$$

The content of the Accumulator before the division is denoted by $(A)_i$. The remainder, or the final content of A, is denoted by R. The remainder left in the Accumulator by the division process is always non-negative. The placement of the dividend in the double-length Accumulator allows the formation of a quotient with the maximum of 35 significant bits (and a single sign bit). This is true regardless of the value of the divisor in the X Register. The magnitude allowable for the quotient is limited only by the capacity of the Q Register. The following examples illustrate some of the facets of the division process.

Case I - dividend and divisor both positive

dividend $(A)_i$ =
000 000...010 100 000 000 000 000 000 000 000 000 000 000 100 $= (2^{37}+2^{35}+2^2)$

divisor (X) = 000 000 000 000 000 000 000 000 000 000 001 010 $= (2^3+2^1)$

derived quotient (Q) =
010 000 000 000 000 000 000 000 000 000 000 000 000 $= (2^{34})$

remainder, final content of A, =
000 000...000 000 000 000 000 000 000 000 000 000 000 100 $= (2^2)$

Case II - dividend and divisor both negative

dividend $(A)_i$ =
111 111...101 011 111 111 111 111 111 111 111 111 111 111 011 $= -(2^{37}+2^{35}+2^{2})$

divisor (X) = 111 111 111 111 111 111 111 111 111 111 111 110 101 $= -(2^{3}+2^{1})$

derived quotient (Q) =
010 000 000 000 000 000 000 000 000 000 000 001 $= (2^{34}+2^{0})$

remainder, final content of A =
000 000...000 000 000 000 000 000 000 000 000 000 000 110 $= (2^{3}-2^{2}+2^{1})$

Case III - positive dividend and negative divisor

dividend $(A)_i$ =
000 000...010 100 000 000 000 000 000 000 000 000 000 100 $= (2^{37}+2^{35}+2^{2})$

divisor (X) = 111 111 111 111 111 111 111 111 111 111 111 110 101 $= -(2^{3}+2^{1})$

derived quotient (Q) =
101 111 111 111 111 111 111 111 111 111 111 111 111 $= -(2^{34})$

remainder, final content of A, =
000 000...000 000 000 000 000 000 000 000 000 000 000 100 $= (2^{2})$

Case IV - negative dividend and positive divisor

dividend $(A)_i$ =
111 111...101 011 111 111 111 111 111 111 111 111 111 111 011 $= -(2^{37}+2^{35}+2^{2})$

divisor (X) = 000 000 000 000 000 000 000 000 000 000 000 001 010 $= (2^{3}+2^{1})$

derived quotient (Q) =
101 111 111 111 111 111 111 111 111 111 111 111 110 $= -(2^{34}+2^{0})$

remainder, final content of A, =
000 000...000 000 000 000 000 000 000 000 000 000 000 110 $= (2^{3}-2^{2}+2^{1})$

It is possible that the quotient that should be derived would exceed the capacity of the Q Register. If the conditions pointing to this fact are detected during the division process, computer operation is stopped immediately, and a Divide fault is indicated on the computer control panel.

1-68. SCALING.

1-69. The Univac Scientific treats all quantities as integers when handling them in arithmetic operations. Numbers not lying in the range $1-2^{35} \leq s \leq 2^{35}-1$ (for single precision operations) can be expressed as $s = s_1 \cdot 2^{s_2}$ where s_1 is an integer within the above range. Thus the machine representation s_m of a quantity s is actually the quantity times some power of two or the quantity "scaled" the number of places specified by the power. The term 2^{s_2} is called the scale factor; the term s^2 is the "scaling."

For example, one representation of the fraction $^1/_2$ in the form $s = s_1 \cdot 2^{s_2}$ is as follows:

$$s = {}^1/_2 = 2^{-1} = 0.1 \text{ in binary notation}$$

In the form $s = s_1 \cdot 2^{s_2}$.

$$0.1 = 1.0\ (2^{-1})$$

The machine representation 1.0 is the quantity 0.1 times 2^1, or the quantity 0.1 scaled one place.

1-70. Two quantities s_m and t_m can be added, subtracted, multiplied, or divided. However. if these quantities are scaled representations of s and t, the scaled representation of the <u>result</u> may not have the same scaling as that of the operands. The programmer must keep a record of the scaling of all operands and all results.

1-71. Also, the results may have to be adjusted, or re-scaled, if it is desired that the machine representations do not exceed the capacity of a 36-bit location. Consider the multiplication of two quantities s and t represented in the form $s = s_1 \cdot 2^{s_2}$ (and $t = t_1 \cdot 2^{t2}$). The scale factor 2^{s_2} is assigned a value, depending upon the range of s, such that s_1 does not exceed $2^{35}-1$ (for a single precision operation). Thus, the quantity s is scaled so that its machine representation s_m does not exceed the capacity of a 36-bit location. The multiplication of the machine representations of s and t yields the result $r_m = s_m \cdot t_m$. In terms of the quantities s and t, the result is in the form $s \cdot t = r = (s_1 \cdot t_1) \cdot 2^{s_2 + t2}$. The machine representation r_m is the result r scaled $-(s_2 + t_2)$. However, if the quantity r_m is to be stored at a 36-bit location, its magnitude must be checked since it may exceed the capacity of a 36-bit location.

For example, consider the multiplication of two binary quantities

$$\begin{aligned} s &= 0.1\ldots &&= 2^{-1} + \ldots \\ \text{and } t &= 0.1\ldots &&= 2^{-1} + \ldots \end{aligned}$$

If twenty significant bits are desired for the machine representation of each number, these numbers, represented in the form $s = s_1 \cdot 2^{s2}$, are

$$(0.1\ldots) = (2^{-1} + \ldots) = (2^{19} + \ldots)2^{+20}$$

The machine representations s_m and t_m are

000 000 000 000 000 01. ... ... $\quad (2^{19} + \ldots)$
000 000 000 000 000 01. ... ... $\quad (2^{19} + \ldots)$

These are the quantities s and t scaled up twenty places. The product r of the quantities s and t is in the range $1.0 > r \geq 0.01$ or $2^0 > r \geq 2^{-2}$.

However, the result is in the form

$$r = (2^{19} + \ldots)\,(2^{19} + \ldots)2^{-40}$$

where the product is scaled 40 places such that the machine representation of the product is in the range $2^{40} > r_m \geq 2^{38}$. In this case the product must be scaled down so that its machine representation is in the range $2^{35} > r_m \geq 2^{33}$. This is accomplished by a "right shift" of five places which insures that the most significant bit of r_m is in A_{34} or A_{33}. Now the result is in the form

$$r = \left[(2^{19} + \ldots)\,(2^{19} + \ldots)2^{-5}\right]2^{-40} \cdot 2^{+5}$$

where the machine representation of the result r_m.

Oxy ... ...• ... ... = $(2^{19}+\ldots)(2^{19}+\ldots)2^{-5}$

is the product scaled 35 places. In this example, either bit x in A_{34} <u>or</u> bit y in A_{33} is a one.

1-72. An example of the above multiplication using specific quantities is as follows.

$$s = 0.100 \qquad = 2^{-1}$$
$$t = 0.110 \qquad = 2^{-1} + 2^{-2}$$

In the form $s = s_1 \cdot 2^{s2}$.

$$0.100 = 2^{-1} = (2^{19})2^{-20}$$
$$0.110 = 2^{-1} + 2^{-2} = (2^{19} + 2^{18})2^{-20}$$

The machine representations s_m and t_m are

000 000 000 000 000 010 000 000 000 000 000 000 = 2^{19}
000 000 000 000 000 011 000 000 000 000 000 000 = $2^{19} + 2^{18}$

The product r of the quantities s and t is 0.011 000. However, the result is in the form

$$r = (2^{19})\,(2^{19} + 2^{18})2^{-40}$$

where the product is scaled 40 places such that the machine representation of the product in the Accumulator is

000 000...110 000 000 000 000 000 000 000 000 000 000 000 000 = $(2^{38}+2^{37})$

This product must be scaled down so that its most significant bit is in A_{34}. This can be accomplished by a "right" shift of only <u>four</u> places. Thus, the result is in the form

$$r = \left[(2^{38} + 2^{37})2^{-4}\right] \cdot 2^{-40} \cdot 2^{4}$$

where the machine representation of the result r_m,

011 000 000 000 000 000 000 000 000 000 000 000 $= (2^{34} + 2^{33})$

is the product scaled 36 places.

1-73. If the maximum number of significant bits are to be retained in the right half of the Accumulator, the most significant bit of the result must be found in A_{34}. A quantity n which is scaled to its maximum representation in a 36-bit location, such that n is in the range of $2^{35} > n \geq 2^{34}$, is said to be normalized. The quantity + 9,

000 000 000 000 000 000 000 000 000 000 001 001

in its normalized form is

010 010 000 000 000 000 000 000 000 000 000 000

The quantity above, expressed in octal notation is 000000000011; this quantity in its normalized form is 220000000000.

1-74. LOGICAL OPERATIONS.

1-75. Certain of the computer operations can be used for the formation of logical sums, logical products, and an "exclusive or" combination of computer words.

The "exclusive or" combination of binary digits is illustrated to the right. The "exclusive or" is one when either, and not both, of the binary digits is one. The "exclusive or" combination of two computer words x and y is found by adding each bit of x to the corresponding bit of y and disregarding any carry. The notation for this process is $x \oplus y$. A computer word formed by the "exclusive or" process is shown below.

0 1 0 1	
0 0 1 1	
0 1 1 0	

x = 000 101 110 000 100 010 000 011 110 010 100 011
y = 011 000 100 110 101 000 111 111 001 101 100 101

$x \oplus y$ = 011 101 010 110 001 010 111 100 111 111 000 110

The logical sum of two binary digits is illustrated to the right. The logical sum, "inclusive or", is one when either, or both, of the binary digits is one. The logical sum of two computer words x and y is formed by placing a one in each bit position where either or both x or y has a one in the corresponding position. A logical sum is illustrated below.

0 1 0 1	
0 0 1 1	
0 1 1 1	

```
x =            000 101 110 000 100 010 000 011 110 010 100 011
y =            011 000 100 110 101 000 111 111 001 101 100 101

logical sum = 011 101 110 110 101 010 111 111 111 111 100 111
```

The logical product of two binary digits is illustrated to the right. The logical product, "conjunction", is one when, and only when, both of the binary digits are one. The logical product of two computer words x and y is found by multiplying each bit of x by the corresponding bit of y. The notation for a logical product is $x \otimes y$. A logical product is illustrated below.

```
0 1 0 1
0 0 1 1

0 0 0 1
```

```
x =      000 101 110 000 100 010 000 011 110 010 100 011
y =      011 000 100 110 101 000 111 111 001 101 100 101

x⊗y = 000 000 100 000 100 000 000 011 000 000 100 001
```

1-76. FLOATING POINT NUMBERS IN THE COMPUTER

1-77. A quantity expressed as a floating point number is in the form $x \cdot 2^y$. The quantity x is termed the mantissa. The quantity y is termed the characteristic. If the mantissa and characteristic are stored in individual registers, the floating point representation is said to be unpacked. If the mantissa and characteristic are stored in the same register, the floating point representation is said to be packed. The Univac Scientific uses a packed representation.

1-78. In the Univac Scientific floating point system, certain restrictions are placed on the range allowable to the quantities x and y. The mantissa x is restricted to a normalized representation such that $\frac{1}{2} \leq x < 1$ (for non-zero numbers). The characteristic y is restricted to $-128 \leq y < 128$.

1-79. The characteristic y is biased for its representation in the computer such that the biased characteristic is always positive. The biased characteristic is y + 128 such that $0 \leq y + 128 < 256$. The bits $u_{34} \ldots u_{27}$ are the biased characteristic of a floating point number packed in register u.

1-80. The mantissa x is represented in one's complement notation by a sign bit and 27 bits of significance. The bits $u_{26} \ldots u_0$ are the significant bits of the mantissa of a floating point number packed in register u. The most significant bit of the normalized mantissa is u_{26}. The binary point of the fractional mantissa x is considered to be between bits u_{27} and u_{26}.

1-81. The sign bit of a floating point number is u_{35}, the normal computer sign bit of register u. The packed floating point representation of the quantity zero has each bit in register u equal to zero.

1-82. A negative floating point number has as its representation the complement of the word denoting the packed number. Note that the complement is formed of both the biased characteristic and the mantissa of the number.

1-83. The representation of packed floating point numbers according to the preceding conventions retains the use, where applicable, of stated point instructions in floating point programming. Instructions which provide such operations as data transmissions, testing for sign, and testing for magnitude are valid.

1-84. Examples of floating point numbers in their packed form are given below. The digits to the right of the radix point represent the mantissa. The digits to the left of the radix point represent the biased characteristic and the sign. The left-most binary digit is the sign bit of the number.

$+1 = +(1/2 \cdot 2^{1})$	binary	010	000	001	.	100	000	000	000	000	000	000	000	000	
	octal	2	0	1	.	4	0	0	0	0	0	0	0	0	
	decimal	(+)	129						2^{-1}						
$-1 = -(1/2 \cdot 2^{1})$	binary	101	111	110	.	011	111	111	111	111	111	111	111	111	
	octal	5	7	6	.	3	7	7	7	7	7	7	7	7	
	decimal	(-)	complement of 129				complement of 2^{-1}								

$1/2 \cdot 2^{0}$	octal	200.400000000
$-(1/2 \cdot 2^{0})$	octal	577.377777777
$1/2 \cdot 2^{-128}$	octal	000.400000000
$3/4 \cdot 2^{127}$	octal	377.600000000
$-(3/4 \cdot 2^{127})$	octal	400.177777777

section 2
INSTRUCTION REPERTOIRE

2-1. PREVIEW OF INSTRUCTIONS

2-2. A Univac Scientific 36-bit word is defined by the notation $i_{35}...i_0$. The composition of a 36-bit instruction word is described as $i_{35}...i_{30}$, $i_{29}...i_{15}$, $i_{14}...i_0$. The six bits $i_{35}...i_{30}$ designate the operation code; the 15 bits $i_{29}...i_{15}$ describe the u-address protion of the word; and the 15 bits $i_{14}...i_0$ describe the v-address portion of the word.

2-3. Each addressable storage location can be directly referenced in an instruction. An arbitrary address, such as u, enclosed in parentheses, (u), denotes the information stored at that address. Thus, (u) is read as "the content of the location addressed as u." Programmed access to information in storage is achieved by specifying in a computer instruction the address of the storage location containing that information.

2-4. The types of operations provided by the Univac Scientific repertoire of instructions are as follows.

a. internal transfers of data
b. stated point arithmetic
c. floating point arithmetic
d. logical operations
e. decision operations which may involve comparisons of data or manual selections, and which may cause a break in the normal sequential acquisition of instructions
f. input and output of data between the computer and external equipment

2-5. Several instructions interrupt the process of acquiring from consecutive addresses the instructions to be executed sequentially. These instructions are

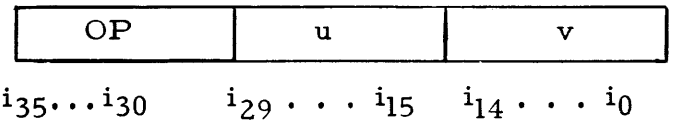
a. Jump instructions: A Jump instruction specifies the address of the next instruction to be executed.
b. a Repeat instruction: A Repeat instruction tells the computer to repeat the instruction at the next consecutive address a prescribed number of times. The form of the instruction being repeated may or may not be modified after each repeated execution. Any modification of the instruction is done in a control register, not in storage. A repeat instruction may be considered a jump instruction since it specifies the address of the instruction to be executed next after the repeated instruction.

c. Stop instructions: A Stop instruction tells the computer to stop operation. If operation is re-started, the next instruction to be executed is either specified by the stop instruction, or acquired automatically from a fixed location.

2-6. INSTRUCTION REPERTOIRE

2-7. NOTATION.

2-8. A Univac Scientific instruction comprises 36 binary digits. An instruction word is logically divided into three parts. These parts are an operation code of six bits, a u-address portion of 15 bits, and a v-address portion of 15 bits. For the sake of brevity, these parts are often referred to as u and v. The figure below illustrates the logical parts of an instruction.



2-9. The u-address portion and the v-address portion of the instruction may designate storage addresses of operands or other instructions. However, in certain of the instructions, other symbols appear in the u-address and v-address portions. These symbols are as follows:

j one octal digit* representing the three bits $i_{29}\ i_{28}\ i_{27}$

n four octal digits* representing the twelve bits $i_{26}\ i_{25} \ldots i_{15}$

k the octal digits* representing the seven bits $i_6\ i_5 \ldots i_0$; or in one case the bits $i_{21}\ i_{20} \ldots i_{15}$.

The functions of j, n, and k are explained later in the description of the functions of the instructions.

2-10. Following is a list of other symbols which are used in the description of the instructions:

(u)	content of the location whose address is u
(u)'	complement of (u)
A	the 72 bit Accumulator
Q	the 36 bit Q Register
Q_i, i=0, 1, . . . , 35	the i^{th} bit in the Q Register
A_i, i=0, 1, . . . , 71	the i^{th} bit in the Accumulator

*Each octal digit represents three bits.

(A_R)	the right-most 36 bits of (A)
(A_L)	the left-most 36 bits of (A)
D(u)	a 72-bit word with 36 sign bits to the left of (u)
S(u)	a 72-bit word with 36 zeros to the left of (u)
$(u) \oplus (v)$	the "exclusive or" combination of (u) and (v): the bit-by-bit sum <u>without carry</u> of (u) and (v).
$(u) \otimes (Q)$	the logical product of (u) and (Q); the bit-by-bit product of (u) and (Q).
$(v) \otimes (Q)'$	the logical product of (v) and (Q)': the bit-by-bit product of (v) and the complement of (Q).
F_1	fixed address 00000 (or 40001 as set by a switch)
F_2	fixed address 00001
F_3	fixed address 00002
F_4	fixed address 00003
(PAK)	the address held in the Program Address Counter.
the word whose address is (PAK)	the 36 bits stored at the address held in PAK

2-11. INSTRUCTION PRESENTATION

2-12. The instruction presentation in this section includes detailed information on each instruction. The instructions are grouped according to basic similar functions. Programming hints and reminders of the peculiarities of certain of the instructions are given in the text, or are illustrated by coding short routines using the instructions. The coding examples presented are not meant to portray the most efficient or only way of obtaining the intended result.

Information on an instruction is organized as follows.

INSTRUCTION — SHORTHAND* NOTATION

FUNCTION

OPERATION SEQUENCE	NON - REPEATED	REPEATED
Sequential listing of the events which occur during the execution of the instruction. This is a comprehensive picture of the function of the instruction. It should be consulted particularly when the u or v address of the instruction references the Accumulator or the Q Register.	Tables which show the time consumed by the computer in executing each instruction, repeated and non-repeated. The tables here cover the cases where u and/or v of the instructions specify Magnetic Core Storage, MC, the Accumulator, A, and the Q Register, Q** If the cases where u or v = A or Q are not practical, or the repeat of the instruction is not realistic, no time is quoted. The times given are in microseconds.	
The events listed in this column do not include those occurring during the termination phase of the instruction. Termination sequences are listed in paragraph 2-14 and with the discussion of the Repeat Instruction.	The non-repeated times in this column equal the microseconds necessary to execute the instruction from its initiation to the time when the next instruction is ready for execution; i.e., from one MPO to the next MPO.	The repeated times in this column are listed as ( )n where n is the number of actual executions of the repeated instruction. The quantity ( )n equals the term R_n in the expression $54+R_n+p$, the time given for the Repeat instruction. The term p is the execution time of the instruction stored at F_1. The instruction at F_1^1 is normally the Manually Selective Jump Instruction which causes a jump to w, where w is specified in the Repeat instruction, RPjnw.

*The SHORTHAND NOTATION for an instruction is OPuv. The operation code OP is given both in octal notation, represented by a pair of digits, and in mnemonic notation, represented by a pair of letters.

**No execution time is given for the cases where u and/or v specifies MD storage. The maximum access time for information on the Magnetic Drum is 34 milliseconds. Minimum access time for MD storage is six microseconds. Random access time for the drum is thus 17 milliseconds. Random access programming for the drum does not achieve the minimum drum access time.

2-13. Each non-repeated instruction is concluded by the Normal Termination Sequence. This sequence prepares the computer for the execution of the next instruction, the address of which is acquired from the Program Address Counter, PAK. The instruction at this address is placed in the Program Control Register, PCR, and (PAK) is advanced by one. The address in PAK at the time this sequence is initiated is either y+1, where y is the address of the instruction just executed, or it is the u or v address of a jump instruction just executed.

2-14. The events of the Normal Termination Sequence are listed below. This sequence is understood to follow the events listed in the OPERATION SEQUENCE of each non-repeated instruction.

(PAK)⟶SAR

Advance (PAK)

the word (instruction) whose address is (SAR)⟶X

(X)⟶PCR

The termination sequences for repeated instructions are listed with the discussion of the Repeat instruction.

2-15. Guides for the interpretation of some of the events listed in the OPERATION SEQUENCE column are noted below.

Clear A, Q, or X	Replace each bit in the register with a zero.
Complement Q or X	Replace each bit in the register with its complement.
(u)⟶X	Read the contents of u and place in X
(X)⟶v	Write the contents of X at v.

Data transmissions in general are denoted by an arrow,⟶. This portrays a transmission from one computer location, to the left of the arrow, to another computer location, to the right of the arrow. Transmissions can be classified as follows.

Note: In none of the cases given below, does the transmission destroy the data at the location to the left of the arrow.

Case I - Thirty-six bits are transmitted, or read, from one location to another. The 36 bits at the right-hand location are replaced by the 36 bits at the left - hand location. There are only two special cases where u (or v) is an Accumulator address. These are as follows.

if u = A, (u)⟶X denotes the transmission (A_R)⟶X unless it is stated to omit this transmission.

if v = A, (X)⟶v denotes the events Clear A and Add D(X) to A, unless it is stated to omit the transmission (X)⟶v.

Case II - Certain specified bits from a 36-bit location are transmitted to another 36-bit location. If this is the case, only the data in the specified positions of the right-hand location is replaced. Transmissions such as those shown below are "partial writes" of (X) at the u- or v-address portion of a location.

$X_{29} \ldots X_{15} \rightarrow v_{29} \ldots v_{15}$	Replace the u-address portion, or the v-address portion of the location addressed as v. NOTE: Partial writing of the Accumulator or the Q Register is not possible. If the location v is A or Q, an attempt to partial write causes a Storage Class Control fault and computer operation is stopped. Indication of the fault is given on the computer control panel.
$X_{14} \ldots X_0 \rightarrow v_{14} \ldots v_0$	
$X_{14} \ldots X_0 \rightarrow$ v-address portion of (F_1)	

Case III - Certain transmissions involve locations with inequal capacities. Such transmissions are defined as follows.

(PAK) $\rightarrow X_{14} \ldots X_0$	Replace only the lower 15 bits of X with the 15 bits in PAK
(I0A) $\rightarrow X_7 \ldots X_0$	Replace only the lower eight bits of X with the eight bits in the Input Output Register A, I0A
$X_7 \ldots X_0 \rightarrow$ I0A	Replace the eight bits of I0A with the lower eight bits of the X Register

Case IV - Some miscellaneous transmissions are as follows.

jn $\rightarrow$ PAK	Replace the content of PAK with the 15-bit quantity jn.
w $\rightarrow X_{14} \ldots X_0$	Replace the content of only the lower 15 bits of X with the 15-bit address w.
k $\rightarrow$ lower order bits of X	Replace the lower order bits of X with the contents, k, of the Shift Counter.
u $\rightarrow$ PAK	Replace the contents of PAK with the 15-bit address u (or v). NOTE: If u is an Accumulator address, and an attempt is made to obtain the next instruction from A, an SCC computer fault occurs. If u is a Q Register address, the next instruction can be obtained from Q. If (Q) is a legal instruction, it is executed. Unless the instruction is a jump instruction, its execution is continued until

the address generated by successive advancements of (PAK) is 32000. Since 32000 is an Accumulator address, an SCC fault is incurred as noted earlier.

2-16. TRANSMISSIVE INSTRUCTIONS

2-17. INSTRUCTION: Transmit Positive

TPuv
11uv

FUNCTION: Transmit (u) to v

OPERATION SEQUENCE	EXECUTION TIME NON - REPEATED				REPEATED	
(u)→X	u \ v	MC	A	Q	u \ v	MC
(X)→v	MC	38	34	30	MC	24n + 2
If v is A, Clear A	A	30	28	24	A	16n + 2
Add D(X) to (A)	Q	32	30	26	Q	18n + 2

2-18. INSTRUCTION: Transmit Magnitude

TMuv
12uv

FUNCTION: Transmit the absolute value of (u) to v

OPERATION SEQUENCE	EXECUTION TIME NON - REPEATED				REPEATED	
(u)→X	u \ v	MC	A	Q	u \ v	MC
Complement (X) if negative	MC	38	36	32	MC	24n + 2
(X) v	A	32	30	26	A	18n + 2
If v is A, Clear A Add D(X) to (A)	Q	34	32	28	Q	20n + 2

2-19. INSTRUCTION: Transmit Negative

TNuv
13uv

FUNCTION: Transmit the complement of (u) to v

OPERATION SEQUENCE	EXECUTION TIME NON - REPEATED				REPEATED	
(u)→X	u \ v	MC	A	Q	u \ v	MC
Complement (X)	MC	38	34	30	MC	24n + 2
(X)→v	A	30	28	24	A	16n + 2
If v is A, Clear Add D(X) to (A)	Q	32	30	26	Q	18n + 2

2-20. INSTRUCTION: Transmit U Address TUuv
15uv

FUNCTION: Transmit the u-address portion of (u) to the u-address portion of v.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED		REPEATED	
	v / u	MC	v / u	MC
(u)→ X	MC	38	MC	24n+2
$X_{29} \ldots X_{15} \rightarrow v_{29} \ldots v_{15}$ (v of A or Q gives SCC fault)	A	30	A	16n+2
	Q	32	Q	18n+2

2-21. INSTRUCTION: Transmit V Address TVuv
16uv

FUNCTION: Transmit the v-address portion of (u) to the v-address portion of v.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED		REPEATED	
	v / u	MC	v / u	MC
(u)→X	MC	38	MC	24n+2
$X_{14} \ldots X_0 \rightarrow v_{14} \ldots v_0$ (v of A or Q gives SCC fault)	A	30	A	16n+2
	Q	32	Q	18n+2

2-22. INSTRUCTION: Left Transmit LTjkv
22jkv

FUNCTION: Left circular shift (A) by k places, k being $i_{21} \ldots i_{15}$. Then transmit (A_L) if j is 0 (or even), or (A_R) if j is 1 (or odd) to v.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED				REPEATED	
	v	MC	A	Q	v	MC
Shift (A) left k places						
If j is 0, (A_L)→X						
If j is 1, (A_R)→X		32+2k	30+2k	26+2k		(18+2k)n+2
(X)→v if v is A Clear A Add D (X) to A						

2-23. The Transmissive instructions provide a means of transmitting a word or a portion of a word from one computer location to another. The word may be altered during the transmission process. The content of the storage location receiving the operand is replaced only as specified by the particular instruction. For example, only 15 bits of (v), its u address, are replaced by the instructions T·Uuv. The instruction LTjkv provides a direct transmission of (A_L) or (A_R) to storage. Other transmissions from the Accumulator to a 36-bit storage location take (A_R) only. The operation of certain of the Transmissive instructions is illustrated below.

Transmit Positive instruction:

```
                    ⎧ initial content of 00100 = 203004115244
11 00100 00200*     ⎨ final content of 00100   = 203004115244
                    ⎩ final content of 00200   = 203004115244
```

Transmit U Address instruction:

```
                    ⎧ initial content of 00300 = 000101000000
                    ⎪ final content of 00300   = 000102000000
15 00300 00400      ⎨ initial content of 00400 = 210000001021
                    ⎩ final content of 00400   = 210102001021
```

Left Transmit instruction:

```
                    ⎧ initial content of Accumulator = 050000000001 700000100001
22 00001 00500      ⎨ final content of Accumulator   = 120000000003 600000200002
   (j=0, k=1)       ⎩ final content of 00500         = 120000000003
```

* An instruction may be coded, as shown here, in octal notation. The format used serves to differentiate between the operation code, the u-address, and v-address portions of the instruction. The operation code could have been expressed as well by its mnemonic notation.

2-24. ARITHMETIC INSTRUCTIONS

RAuv
2luv

2-25. INSTRUCTION: Replace Add

FUNCTION: Form in A the sum of D(u) and D(v). Then replace (u) with (A_R).

OPERATION SEQUENCE	EXECUTION TIME NON - REPEATED			REPEATED		
(u) → X	v / u	MC	A	Q	v / u	MC	A	Q
Clear A Add D(X) to (A)	MC	60	54	56	MC	46n+2	40n+2	42n+2
(v) → X	A	46	40	42	A	36n		
Add D(X) to (A) (A_R)→X	Q	50	44	46	Q	38n		
(X) → u Omit if u is A								

RSuv
23uv

2-26. INSTRUCTION: Replace Subtract

FUNCTION: Form in A the difference D(u) minus D(v). Then replace (u) with (A_R).

OPERATION SEQUENCE	EXECUTION TIME NON - REPEATED			REPEATED		
(u)→X	v / u	MC	A	Q	v / u	MC	A	Q
Clear A Add D(X) to (A)	MC	62	56	58	MC	48n+2	42n+2	44n+2
(v) → X	A	48	42	44	A	38n		
Subtract D(X) from (A) (A_R)→X	Q	52	46	48	Q	40n		
(X)→ u Omit if u is A								

ATuv
35uv

2-27. INSTRUCTION: Add and Transmit

FUNCTION: Add D(u) to (A). Then transmit (A_R) to v.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED				REPEATED			
	u \ v	MC	A	Q	u \ v	MC	A	Q
(u)→X								
Add D(X) to (A)								
(A_R)→X	MC	44	36	38	MC	30n+2	26n	26n
(X)→v	A	38	30	32	A	24n+2		
Omit if v is A								
	Q	40	32	34	Q	26n+a		

STuv
36uv

2-28. INSTRUCTION: Subtract and Transmit

FUNCTION: Subtract D(u) from (A). Then transmit (A_R) to v.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED				REPEATED			
	u \ v	MC	A	Q	u \ v	MC	A	Q
(u)→X								
Subtract D(X) from (A)								
(A_R)→X	MC	46	38	40	MC	32n+2	28n	28n
(X)→v	A	40	32	34	A	26n+2		
Omit if v is A								
	Q	42	34	36	Q	28n+2		

2-29. INSTRUCTION: Multiply

MPuv
7luv

FUNCTION: Form in A the 72 bit product of (u) and (v), leaving in Q the multiplier (u).

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	EXECUTION TIME REPEATED
(u)⟶X Clear A (X)⟶Q (v)⟶X Form in A the product of (Q) and (X)	u \ v: MC: 116, A: 110, Q: 112 A: 110, 104, 106 Q: 112, 106, 108 each plus $4u_0 + 8\sum_{i=1}^{35} u_i + 14u_{35}$	Minimum times, where $(u_{35})_j$ and $(u_0)_j$ are zeros, and the additional time required for $(u34\ldots u_1)_j$ of ones is given by the summation below the table. u \ v: MC: 104n, A: 98n, Q: 100n A: 98n, 92n, 94n Q: 100n, 94n, 96n each plus $8\sum_{j=1}^{n}\left(\sum_{i=1}^{34} u_i\right)_j$ Maximum times, where $(u_{35}\ldots u_0)_j$ are all ones, are given below. u \ v: MC: 402n, A: 396n, Q: 394n A: 396n, 390n, 392n Q: 398n, 392n, 394n
	If the contents of $(u_{35}\ldots u_0)_j$ are known: u = MC, v = MC $104n + \sum_{j=1}^{n}\left\{4(u_0)_j + 14(u_{35})_j + 8\sum_{i=1}^{35}(u_i)_j\right\}$ u = Q, v = Q $96 + n\left\{4Q_0 + 14Q_{35} + 8\sum_{i=1}^{35} Q_i\right\}$	

2-30. INSTRUCTION: Multiply Add

MAuv
72uv

FUNCTION: Add to (A) the 72 bit product of (u) and (v), leaving in Q the multiplier (u).

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	EXECUTION TIME REPEATED
(u)→X (X)→Q Shift (A) left 36 places. (v)→X Check for an Overflow fault Add to $(A)_i$ the product of (Q) and (X)	see table below	see tables below

u \ v	MC	A	Q
MC	188	182	184
A	182	176	178
Q	184	178	180

each plus $4u_0 + 8\sum_{i=1}^{35} u_i + 14u_{35}$

Minimum times where $(u_{35})_j$ and $(u_0)_j$ are zeros, and the additional time required for $(u_{34}\ldots u_1)_j$ of ones is given by the summation below the table.

u \ v	MC	A	Q
MC	176n	170n	172n
A	170n	164n	166n
Q	172n	166n	168n

each plus $8\sum_{j=1}^{n}\left(\sum_{i=1}^{34} u_i\right)_j$

Maximum times, where $(u_{35}\ldots u_0)_j$ are all ones, are given below.

u \ v	MC	A	Q
MC	474n	468n	466n
A	468n	462n	464n
Q	470n	464n	466n

If the contents of $(u_{35}\ldots u_0)_j$ are known:

u = MC, v = MC.

$$176n + \sum_{j=1}^{n}\left\{4(u_0)_j + 14(u_{35})_j + 8\sum_{i=1}^{35}(u_i)_j\right\}$$

u = Q, v = Q

$$168 + n\left\{4Q_0 + 14Q_{35} + 8\sum_{i=1}^{35}Q_i\right\}$$

2-31. INSTRUCTION: Divide DVuv
73uv

FUNCTION: Divide the 72-bit number in A by (u), putting the quotient in Q, and leaving in A the non-negative remainder, R. Then replace (v) by (Q). The quotient and the remainder are defined by $(A)_i = (u) \cdot (Q) + R$ where $0 \leq R < |(u)|$. $(A)_i$ denotes the initial contents of A.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED				EXECUTION TIME REPEATED	
	u \ v	MC	A	Q	u \ v	MC
(u)→X	MC	482	480	476	MC	$468n+8 \sum_{j=1}^{n} (A_{71})_j + 2$
Clear Q Divide (A) by (X) placing the quotient in Q and leaving R in A. During the process, check for a Divide fault.	A	476	474	470		
	Q	478	476	472		
	each plus 8 A_{71}					
(Q)→X						
(X)→v if v is A, Clear A Add D(X) to A						

2-32. STATED POINT ARITHMETIC INSTRUCTIONS

2-33. The stated point arithmetic instructions utilize the computer operations of stated point addition, subtraction, multiplication, and division, as discussed in the section entitled Stated Point Computer Arithmetic.

2-34. The instructions RAuv, RSuv, ATuv, and STuv, order the formation of a sum or difference in the Accumulator and then order a transmission of (A_R) to the u or v-addressed location. If the sum or difference which was formed in A is greater than $2^{35}-1$ in absolute value, the quantity transmitted to a 36-bit storage location is only the lower-order 36 bits of that sum or difference. This quantity, if it is obtained later from u or v, is not interpreted as the correct value in an arithmetic operation. (The transmission of the content of A to a u or v-addressed location does not disturb the contents of A.) Consider the operations of the following two instructions. A preliminary inspection would give the impression that the Accumulator is always left in its cleared state.

RA c0 c1* Add the double-length extension of (c0), D(c0), to D(c1) in A. Then replace (c0) with (A_R).

ST c0 A Subtract D(c0) from (A).

The results of these operations, assigning quantities to (c0) and (c1), can be shown as follows.

RA	c0	c1	initial content of c0 =		010000600010
			initial content of c1 =		000000000001
			final content of c1 =		000000000001
			content of Accumulator =	000000000000	010000600011
			final content of c0 =		010000600011
ST	c0	A	final content of c0 =		010000600011
			final content of Accumulator =	000000000000	000000000000

However, if, for example, c0 initially contains 377777777777 and c1 contains 000000000001, the Subtract and Transmit instruction subtracts the quantity 777777777777 400000000000 from the quantity000000000000 400000000000 in the Accumulator. This leaves $2^{36}-1$ in A.

2-35. In the case of the Replace Add instruction, an overflow can occur only in A_{35}, since A is cleared before any addition. An overflow is evidenced by a sum greater than $2^{35}-1$ in absolute value, i.e., $|(u)_i+(v)| > 2^{35}-1$. In the case of the Add and Transmit instruction, an overflow can occur not only in A_{35} but in A_{71} (and, of course, in all intervening places). These overflows are evidenced by a sum greater in absolute value than $2^{35}-1$ or $2^{71}-1$, respectively. In this case the absolute value of the sum is $|(A)_i + (v)|$. Programmed checks can be made for any of these overflow conditions. In the case of the Subtract and Transmit instruction, an "overflow" result is stored at a v-addressed 36-bit location if $|(A)_i - (u)| > 2^{35}-1$.

2-36. The Replace Add instruction is useful for incrementing the u and v address portions of instructions. An example is given below.

RA	d1	h1	initial content of d1 =	11	u	v
			final content of d1 =	11	u+1	v
			initial content of h1 =	00	00001	00000
			final content of h1 =	00	00001	00000

2-37. The inability to generate, in the Accumulator, a negative zero representation during arithmetic operations is exemplified by the following instructions.

RS A A Clear the Accumulator to zero
AT g1 A Add D(g1) to (A) where (g1) = 777777777777

Since the addition of D(g1) to (A) is accomplished by the subtraction of the complement of D(g1) from (A), the Accumulator is left all zeros.

* An instruction may be coded, as shown here, in mnemonic notation. The u and v address portions are assigned arbitrary addresses.

2-38. The Multiply instruction clears any previous content from the Accumulator before forming the product in A. The Multiply Add instruction provides for adding the product to any number currently in the Accumulator. During the execution of the Multiply Add instruction, but before the product is formed, a special test is made of the content of the Accumulator. The test made checks for the condition $2^{71} > (A)_i \geq 2^{70}$. This condition indicates imminent overflow of the Accumulator when the product of the multiplication is added to (A). (Actually, the check is for $A_{35} \neq A_{34}$ since the content of A is shifted left 36 places before the test is made.) The detection of an imminent overflow condition prevents the occurrence of this multiplication unless the computer operator specifically desires it. The overflow possibility is indicated on the computer control panel as an Overflow fault, and the computer is stopped immediately. However, operation may be resumed by manual selections at the control panel, and the operation might possibly be performed correctly.

2-39. To illustrate this overflow check feature, consider the following example where $(A)_i$ is assumed to be 2^{70}, in binary notation,

010 000...000 000 000 000 000 000 000 000 000 000 000 000 000 000.

Any product less than 2^{70} can be added to this quantity without causing an overflow. However, the overflow check stops the computer and indicates imminent overflow. In this case, multiplication and addition would be carried out correctly if operation were resumed after the computer stop.

2-40. A check is made during the execution of the Divide instruction to determine if the quotient which should be derived would exceed the capacity of the Q Register. If the conditions pointing to this fact are detected, computer operation is stopped immediately. This condition is indicated as a computer Divide fault on the computer control panel. Operation may be resumed by manual selections at the control panel in which case the division is brought to a conclusion. However, the contents of the Accumulator and the Q Register cannot be regarded as the correct remainder and quotient.

The following examples illustrate the Divide check.

Case I - the dividend in the Accumulator is $2^{70} - 2^{35} - 1$
the divisor obtained from (u) is $2^{35} - 1$
the quotient in the Q Register will be $2^{35} - 1$
the remainder left in the Accumulator is $2^{35} - 2$

Case II - the dividend in the Accumulator is $-(2^{70} - 2^{35} - 1)$
the divisor obtained from (u) is $-(2^{35} - 1)$

Computer operation is stopped when it is detected that the correct quotient (with a non-negative remainder) is $+(2^{35})$, and the correct non-negative remainder is +1.

Case III - the dividend in the Accumulator is $2^{70} - 2^{35} - 1$
the divisor obtained from (u) is $-(2^{35} - 1)$
the quotient in the Q Register will be $-(2^{35} - 1)$
the remainder left in the Accumulator is $2^{35} - 2$

Case IV - the dividend in the Accumulator is $-(2^{70} - 2^{35} - 1)$
the divisor obtained from (u) is $+(2^{35} - 1)$

Computer operation is stopped when it is detected that the correct quotient (with a non-negative remainder) is $-(2^{35})$, and the correct non-negative remainder is +1.

2-41. SPLIT INSTRUCTIONS

SPuk
2-42. INSTRUCTION: Split Positive Entry 31uk

FUNCTION: Form S(u) in A. Then left circular shift (A) by k places, k being $i_6 \ldots i_0$.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED		EXECUTION TIME REPEATED
(u)→X	u		u
Clear A	MC	32+2k	MC (20+2k)n
Add S(X) to (A)			
Shift (A) left k places	A	26+2k	where k is not altered by the Repeat Sequence. For k = 0 and 1, use value of k = 2
	Q	28+2k	

SAuk
2-43. INSTRUCTION: Split Add 32uk

FUNCTION: Add S(u) to (A). Then left circular shift (A) by k places, k being $i_6 \ldots i_0$.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED		EXECUTION TIME REPEATED
(u)→X			
Add S(X) to (A)	u		u
Shift (A) left k places	MC	32+2k	MC (20+2k)n
	A	26+2k	where k is not altered by the Repeat Sequence. For k = 0 and 1, use value of k = 2
	Q	28+2k	

2-44. INSTRUCTION: Split Negative Entry SNuk
33uk

FUNCTION: Form in A the complement of S(u). Then left circular shift (A) by k places, k being $i_6 \ldots i_0$.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED		REPEATED	
(u)→X	u		u	
Clear A	MC	34+2k	MC	(22+2k)n
Subtract S(X) from (A)				
Shift (A) left by k places	A	28+2k	where k is not altered by the Repeat Sequence.	
	Q	30+2k	For k = 0 and 1, use value of k = 2	

2-45. INSTRUCTION: Split Subtract SSuk
34uk

FUNCTION: Subtract S(u) from (A). Then left circular shift (A) by k places, k being $i_6 \ldots i_0$.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED		REPEATED	
(u)→X	u		u	
Subtract S(X) from (A)	MC	34+2k	MC	(22+2k)n
Shift (A) left by k places	A	28+2k	where k is not altered by the Repeat Sequence.	
	Q	30+2k	For k = 0 and 1, use value of k = 2	

2-46. The Split instructions order the addition or subtraction of split extensions into the Accumulator, and then provide an optional left shift k places of (A). Split extensions are discussed in the section Stated Point Computer Arithmetic. The Split Positive Entry instruction coded with u = A and k = 0 clears A_L of its content and restores the original content of A_R. The Split Subtract instruction with u = A and k = 0 leaves (A_L) undisturbed and clears A_R. When A_R contains zeros, the content of A_L is never disturbed by a Split Add instruction with k = 0. The split instructions can be employed as follows to enter a double precision number into the Accumulator. *

Location	Op Code	u address	v address	Explanation
a1	SP	c1	00044	Clear the Accumulator Enter S(c1) in A Shift (c1) into A_L
a2	SA	c2	00000	Add S(c2) to A
c1	01	06002	00001	36 most significant bits of no.
c2	00	00000	00007	36 least significant bits of no.

The content of the Accumulator after the SP instruction is (expressed in octal)
01060020001 000000000000

The content of the Accumulator after the SA instruction is (expressed in octal)
01060020001 000000000007

2-47. Split instructions could also be included in the categories Arithmetic instructions and Shift instructions. If two quantities are to be added, one of which is already in the Accumulator and the other of which is known to be positive, and the sum is to be left in the Accumulator, the execution of the Split Add instruction consumes less time than the other add instructions. A similar statement can be made for subtracting using the Split Subtract instruction.

2-48. The Split Add and Split Subtract instructions are useful in rounding and scaling operations. Consider the case where the number of significant bits in the Accumulator is known, and this quantity is to be scaled so that it is held in A_R. Suppose the number in A is known to be positive and to have not more than 40 significant bits. The following instructions round-off at the sixth place, and scale the quantity to 36 bits. The concluding Transmit Positive instruction clears the least significant bits from A_L after the shifting operation. These bits must be discarded if the content of A is to be involved in further arithmetic operations.

* The format used in this example is typical for the transcription of coding. The left-most "location" column gives the storage address of each instruction.

Location	Op Code	u address	v address	Explanation
b1	SA	c3	00103	Round-off by adding S(c3) to (A) "Right shift" 5 or 72-k places
b2	TP	A	A	Eliminate unwanted bits in A_L
c3	00	00	00020	binary 010 000

If the quantity in the Accumulator initially is assumed to be
000 000...001 100 100 000 000 000 000 000 000 000 000 000 100 100

After the addition of S(c3) to (A), the content of A is
000 000...001 100 100 000 000 000 000 000 000 000 000 000 110 100

After the "right shift" five places, (A) =
101 000...000 000 011 001 000 000 000 000 000 000 000 000 000 001

After the Transmit Positive instruction, (A) =
000 000...000 000 011 001 000 000 000 000 000 000 000 000 000 001

2-49. If the quantity to be scaled and rounded by this method is negative, the Split Add instruction is replaced by the Split Subtract instruction. This method minimizes the round-off error to be in the range of plus or minus one-half in the least significant place. A shift instruction used in place of the Split Add or Split Subtract would allow a round-off error of a maximum of plus or minus one. (An instruction which shifts only, without an addition or subtraction, truncates the contents of A.)

2-50. If the bits $i_{14}...i_0$ of the Split instructions are not zeros, the next instruction to be executed might not be obtained from the next consecutive address y. The events which determine this are as follows. The v portion of a Split instruction is placed in SAR, where the lower seven places are used as a shift counter. At the completion of k shifts, the lower seven bits of SAR are zeros. During the termination phase of the Split instruction, the logical sum of (PAK) and the current (SAR) is formed in SAR. SAR is not cleared before this procedure. The next instruction is then acquired from the address (logical sum) now in SAR. (The logical sum of two bits is one if either or both of the two bits is one.)

(PAK) containing address of next instruction $= y_{14}y_{13}y_{12}\ y_{11}y_{10}y_9\ y_8y_7y_6\ y_5y_4y_3\ y_2y_1y_0$

(SAR) after shifting k places $= i_{14}i_{13}i_{12}\ i_{11}i_{10}i_9\ i_8i_70\ 0\,0\,0\ 0\,0\,0$

(SAR) during termination sequence = logical sum

2-51. For example, if the instruction SP u 77103 were coded, note the following conditions in the registers. The assumption is made that this instruction is stored at address y-1 = 02200.

(PAK) containing address y	=	000 010 010 000 001
(SAR) after shifting 67 places	=	111 111 000 000 000
(SAR) during termination sequence	= logical sum	111 111 010 000 001

The contents of address 77201 is now read from storage. If it is an instruction, the instruction is executed. The next instruction is acquired from address y+1 = 02202, unless (77201) is a jump instruction. If the address formed in SAR is an Accumulator address, the attempt to acquire the next instruction from the Accumulator causes a Storage Class Control fault.

2-52. LOGICAL INSTRUCTIONS

2-53. INSTRUCTION: Controlled Complement — CCuv / 27uv

FUNCTION: Replace (A_R) with (u) $\oplus$ (v). Replace (u) with (A_R). The effect of this instruction is to complement those bits of (u) which correspond to one's in (v).

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED			EXECUTION TIME REPEATED		
(u)→X	u \ v	MC	A	Q	MC	A	Q
Clear A_R							
Complement (X)	MC	52	46	48	38n+2	32n+2	34n+2
Complement $A_{35} \ldots A_0$ if corresponding $X_{35} \ldots X_0$ is zero.	A	38	32	34	28n		
	Q	42	36	38	30n		
(v)→X							
Complement (X)							
Complement $A_{35} \ldots A_0$ if corresponding $X_{35} \ldots X_0$ is zero.							
(A_R)→X							
(X)→u Omit if u is A							

2-54. INSTRUCTION: Q-Controlled Transmit

QTuv
5luv

FUNCTION: Form in A the split extension of (u) ⊗ (Q). Then replace (v) with (A_R).

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED				REPEATED			
(u)→X Clear A Form in X the bit-by-bit product of (Q) and (X) Add S(X) to (A) (X)→v Omit if v is A	u \ v	MC	A	Q	u \ v	MC	A	Q
	MC	44	36	38	MC	30n+2	26n	26n
	A	38	30	32	A	24n+2		
	Q	40	32	34	Q	26n+2		

2-55. INSTRUCTION: Q-Controlled Add

QAuv
52uv

FUNCTION: Add to (A) the split extension of (u) ⊗ (Q). Then replace (v) with (A_R).

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED				REPEATED			
(u)→X Form in X the bit-by-bit product of (Q) and (X) Add S(X) to (A) (A_R)→X (X)→v Omit if v is A	u \ v	MC	A	Q	u \ v	MC	A	Q
	MC	46	38	40	MC	32n+2	28n	28n
	A	40	32	34	A	26n+2		
	Q	42	34	36	Q	28n+2		

2-56. INSTRUCTION: Q-Controlled Substitute

QSuv
53uv

FUNCTION: Form in A the arithmetic sum of the split extension of (u) ⊗ (Q) and the split extension of (v) ⊗ (Q)'. Then replace (v) with (A_R). The effect of this instruction is to replace the bits of (v) with the bits of (u) where there are "1's" in Q.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED			REPEATED		
(u)→X	u \ v	MC	A	Q	u \ v	MC	A	Q
Clear A	MC	74	60	64	MC	60n+2	50n	52n
Form in X the bit-by-bit product of (Q) and (X)	A	68	54	58	A	54n+2		
Add S(X) to (A)	Q	70	56	60	Q	56n+2		
Complement (Q)								
(v)→X								
Form in X the bit-by-bit product of (Q) and (X)								
Add S(X) to (A)								
Complement (Q)								
(A_R)→X								
(X)→v Omit if v is A								

2-57. In the formation of logical products, either the content of the Q Register or the content of u may be used as a "mask." The logical product of a quantity and a mask has one's only where the quantity and the mask both have one's in the same position.

2-58. It is interesting to note that the Controlled Complement instruction can be used to generate all one's in the Accumulator. One such means of doing this is as follows.

Location	Op code	u address	v address	Explanation
d0	RS	A	A	Clear the Accumulator to zero
d1	SS	e0	0	Subtract S(e0) from A; i.e., from 77 77777 77777 in A_L
d2	CC	A	e0	Complement the zeros in A_R; i.e., form negative zero in A
e0	77	77777	77777	

Note that the Controlled Complement instruction does not disturb (A_L).

2-59. The Q Controlled Substitute instruction can be used to form the logical sum of two 36-bit words. Assume that initially w1 and w2 contain the arguments, and that (w1) are to be replaced by the desired sum.

Location	Op code	u address	v address	Explanation
y0	TP	w2	Q	Transmit (w2) to the Q Register
y1	QS	Q	w1	Replace the bits of (w1) with the bits of (w2) where there are 1's in w2; i.e., form the logical sum of (w1) and (w2) in (w1).
w1	00	00000	00003	
w2	00	00000	00005	

Using the contents of w1 and w2 as shown above, the initial and final binary contents of these registers are as follows.

$(w2)$ = 000 000 000 000 000 000 000 000 000 000 000 101
$(w1)_i$ = 000 000 000 000 000 000 000 000 000 000 000 011
$(w1)_f$ = 000 000 000 000 000 000 000 000 000 000 000 111

2-60. SHIFT INSTRUCTIONS

2-61. INSTRUCTION: Left Shift in A

LAuk
54uk

FUNCTION: Replace (A) with D(u). Then left circular shift (A) by k places, k being $i_6 \ldots i_0$, and replace (u) with (A_R). If u is A omit the first and last step.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED			REPEATED	
	r* / u	u	A	Q	r* / u	u
Clear X (u)→X Omit if u is A	MC	44+2k	36+2k	38+2k	MC	(30+2k)n+2
Clear A Omit if u is A	A	30+2k	30+2k			
Add D(X) to (A)	Q	34+2k	32+2k	34+2k		
Shift (A) left k places (A_R)→X						
(X)→u (=r*) Omit if u is A						

*See text which follows.

2-62. INSTRUCTION: Left Shift in Q

LQuk
55uk

FUNCTION: Replace (Q) with (u). Then left circular shift (Q) by k places, k being $i_6 \ldots i_0$, and replace (u) with (Q).

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED			REPEATED	
	r* / u	u	A	Q	r* / u	u
(u)→X	MC	42+2k	40+2k	36+2k	MC	(28+2k)n+2
(X)→Q Shift (Q) left k places (Q)→X	A	34+2k	34+2k			
(X)→u(=r*) If u is A Clear A Add D(X) to (A)	Q	32+2k	32+2k	32+2k		

*See text which follows.

2-63. The Shift instructions provide the means of displacing the bits of a number to the left, a prescribed number of places. Preliminary statements on shifting are found in the section Stated Point Computer Arithmetic. The shifting is done in the Accumulator or the Q Register in a circular manner. A "right shift" is obtained from a left shift by observing that a left shift of k places is equivalent to a right shift of 36-k places in the Q Register or 72-k places in the Accumulator. The value of the shift count is in the range $2^7 > k \geq 0$.

2-64. Note that if the most significant bit of the number is shifted into the sign-bit position of the register, the sign of the shifted number must be regarded as opposite in sign to the initial number if this shifted number is involved later in an arithmetic operation. However, if this shifted number is referenced for later use by one of the Split instructions, no signed value is assumed for the quantity.

2-65. The instructions LAuk and LQuk can be coded in such a manner that the shifted number remains in A or Q instead of replacing the original number in the u-address location. The address to which the final content of A_R or Q is transmitted is determined in the following manner.

2-66. The v address portion of a Shift instruction is placed in SAR, where the lower seven places are used as a shift counter. At the completion of k shifts the lower seven bits of SAR are zeros. In order to write the shifted quantity at the u address, the u address must be inserted in SAR. However, SAR is not cleared before this insertion. The shifted quantity is written at the current content of SAR, which is the <u>logical sum</u> of the u address and the previous content of SAR. This address is arbitrarily called the r address. (The logical sum of two bits is one if either or both of the two bits is one.)

u address location = $i_{29}i_{28}i_{27}$ $i_{26}i_{25}i_{24}$ $i_{23}i_{22}i_{21}$ $i_{20}i_{19}i_{18}$ $i_{17}i_{16}i_{15}$

(SAR) after shifting k places = $i_{14}i_{13}i_{12}$ $i_{11}i_{10}i_9$ i_8i_7 0 0 0 0 0 0 0

(SAR) = r address = logical sum

For example, if the instruction LA 03001 32015 were coded, note the following conditions in the registers.

u address of instruction	=	000 011 000 000 001
(SAR) after shifting 13 places	=	011 010 000 000 000
(SAR) = r address	= logical sum	011 011 000 000 001

The r address in this case is 33001, an Accumulator address. Therefore the shifted quantity is left undisturbed in the Accumulator, and the original quantity in u is left undisturbed.

2-67. This characteristic allows these instructions to serve in some cases as transmissive instructions. The u address can specify the location from which a quantity is obtained without having this quantity replaced later. This feature may be valuable if u is a Magnetic Core address and a quantity (shifted or not shifted) is desired in the Accumulator and/or the Q Register. Note the operations of the following instructions:

LQ	00100	32005	Transmit (00100) to Q Shift (Q) left 5 places Clear A Add D(Q) to (A)
LA	00100	31005	Clear A Add D(00100) to (A) Shift (A) left 5 places Transmit (A_R) to Q

Note that any Magnetic Core address or a Q address may be coded as u in the first instruction LQ u 32000+k. However, if core addresses other than addresses x0xxx or x1xxx are coded as u in the second instruction, LA u 31000+k, the shifted quantity will remain in A. The instruction LA 31xxx 32xxx shifts $(Q)_i$ in A and retains the shifted quantity in A. If the v address is a drum address, i.e., bit i_{14} is a one, the shifted quantity is always transmitted to the drum.

2-68. REPEAT INSTRUCTION

RPjnw

2-69. INSTRUCTION: Repeat 75jnw

FUNCTION: This instruction calls for the next instruction to be executed n times, $0 \leq n < 4096$, its u and v address being modified or not according to the value of j. Normally n executions are made and the program is continued by the execution of the instruction stored at a fixed address F_1. However, if the repeated instruction is a jump or stop instruction, the content of its u or v address may be taken as the next instruction.

The u and v addresses of the repeated instruction are advanced by one unit according to the value of j as follows.

If j = 0, neither the u nor the v address is advanced.
j = 1, the v address is advanced after each execution.
j = 2, the u address is advanced after each execution.
j = 3, both the u and v addresses are advanced after each execution.

During a repeat operation, PAK is used as a counter to determine the number of times the instruction is executed.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	EXECUTION TIME REPEATED
Clear X w → $X_{14} \ldots X_0$ $X_{14} \ldots X_0$ → v-address of (F_1) the word whose address is (PAK) → X jn → PAK Complement (PAK) (X) → PCR	54 + (R_n + p) where p is the execution time of (F_1) and R_n is the execution time of the repeated instruction. If n=0, R_n = 0.	If two Repeat instructions are programmed consecutively, the second takes precedence over the first. The content of PAK, as left by the first Repeat, is used as the address of the instruction to be repeated. This address is the complement of jn-1 where jn is from the first RPjnw.

Advance (PAK)
Investigate contents of PAK
If n = 0, proceed to NORMAL REPEAT TERMINATION SEQUENCE
If n ≠ 0, continue with following.

Perform the OPERATION SEQUENCE OF REPEATED INSTRUCTION.

If jump or stop condition was met, proceed to NORMAL TERMINATION SEQUENCE or JUMP (REPEAT) TERMINATION SEQUENCE
If jump or stop was not indicated, continue with following.

Advance (PAK)
Test (PAK) for n repeats
If n executions have occurred, proceed to NORMAL REPEAT TERMINATION SEQUENCE
If n executions have not occurred, continue with following.

Advance UAK and/or VAK
Repeat the OPERATION SEQUENCE OF REPEATED INSTRUCTION and continue with the events following it as shown above.

2-70. The Repeat instruction states that the execution of (F_1) follows n executions of the next consecutive instruction. This assumes that the Repeat operation is terminated by the Normal Repeat Termination Sequence (which is always true if n = 0). If n > 0, certain other termination sequences, which do not refer to (F_1), may be used. Essentially, the termination sequence which occurs is that which provides the specified jump if a jump condition is met during the repeat of an instruction.

NORMAL REPEAT TERMINATION SEQUENCE	NORMAL TERMINATION SEQUENCE	JUMP (REPEAT) TERMINATION SEQUENCE
This sequence terminates all repeated instructions if n = 0; all repeated non-jump instructions; and all repeated jump instructions if no jump condition is satisfied during their repeat n times. During the Repeat instruction, w is stored at the v-address of F_1. If (F_1) is a Manually Selective Jump instruction, the execution of (F_1) is followed by the execution of (w).	This sequence terminates a repeated jump instruction (except the Equality Jump and the Threshold Jump) after the first execution which satisfies a jump condition. During this execution, the jump address is inserted into PAK. This sequence is used (1) after <u>one</u> execution of the instructions Interpret Return Jump Q-Jump Sign Jump Zero Jump Manually Selective Stop (2) after the jump condition is met by the instructions Index Jump Manually Selective Jump	This sequence terminates the instructions Equality Jump and Threshold Jump if a jump condition is satisfied during their repeated executions. Note that the termination sequence sets aside the current (PAK) before inserting the jump address into PAK. The quantity set aside indicates the number of repeated executions. Note also that the NORMAL TERMINATION SEQUENCE concludes this sequence.
(F_1) → X (X) → PCR	(PAK) → SAR advance (PAK) the word whose address is (SAR) → X (X) → PCR	clear X (PAK) → $X_{14}...X_0$ clear Q (X) → Q v → PAK (PAK) → SAR advance (PAK) the word whose address is (SAR) → X (X) → PCR

2-71. If the Normal Repeat Termination Sequence is used, and (F_1) is not a jump instruction, the instruction executed after (F_1) is determined by the quantity left in PAK by the Repeat operation. This quantity is interpreted as

address 40000 if j was 0
address 70000 if j was 1
address 60000 if j was 2
address 50000 if j was 3

2-72. The value j may also be 4, 5, 6, or 7. These values correspond to values of j of 0, 1, 2, 3, such that if

j = 4, neither address is advanced
j = 5, the v address is advanced
j = 6, the u address is advanced
j = 7, both addresses are advanced

Values of j = 4, 5, 6, or 7 cause an unterminated repeat of the next instruction. The u and v addresses of the repeated instruction are advanced according to their storage class. The advancement of MC, Q, and A addresses in UAK and VAK is discussed in the Control section.

2-73. The Repeat instruction may be used to economize on both storage space and computer operating time when (1), a number of instructions with the same operation code are to be executed consecutively and (2) the u and/or v addresses of these instructions are to be increased by one. The instruction to be repeated is stored following the Repeat instruction. The repeated instruction is acquired from storage only once and held in the Program Control Register during its execution. The u and/or v addresses are incremented, according to a j term, by increasing the content of UAK and/or VAK by one. (The form of the instruction in storage is not altered.)

2-74. Normally the instruction is repeated n times and the program is continued by the execution of the instruction stored at a fixed address F_1. The instruction usually stored at F_1 is a Manually Selective Jump which orders a jump to its v address. Since the Repeat instruction stores an address w in the v-address portion of F_1, this means that (w) is the next instruction executed after (F_1).

2-75. The overall time in microseconds for a Repeat operation, not including the execution of (w), is thus

$$\begin{pmatrix} n \times \text{repeated} \\ \text{execution time} \end{pmatrix} + \begin{pmatrix} \text{execution time of} \\ \text{the Repeat instruction} \end{pmatrix} + \begin{pmatrix} \text{execution time of the in-} \\ \text{struction at } F_1 \end{pmatrix}$$

The term (n x repeated execution time) is defined as R_n and is given in the tables for the repeated execution time of each instruction. The execution time of the Repeat instruction is 54 microseconds. The execution time of the instruction at F_1 is defined as p. Thus, the time in microseconds for a Repeat operation is

$$R_n + 54 + p.$$

If (F_1) is a Manually Selective Jump instruction, p = 18. In those cases where (F_1) is not executed, p = 0.

2-76. The execution time of a repeated instruction is less than the non-repeated time since each execution of a repeated instruction need not be concluded by a termination sequence to acquire another instruction from storage. However, to provide any real economy of overall operating time, the following expression must be true. (The quantity n is the number of times the non-repeated instruction is executed. The non-repeated instruction is assumed to be stored at n consecutive locations.)

$$(R_n + 54 + p) < (n \times \text{non-repeated execution time})$$

Thus it may not be advantageous timewise to use the Repeat instruction to accomplish only a few repeats of an instruction.

2-77. By using the Repeat instruction, a long sequence of operations can be performed with only two or three references to storage for instructions to govern the sequence. This is evidenced by the following examples.

2-78. To transfer a block of words from one series of consecutive storage locations to another, the two instructions below suffice.

```
RP 3 n w
TP d0 c0
```

The addresses d0 and c0 are advanced by a unit increment, according to the j of three, until n words have been transferred.

2-79. Two precautions must be taken in using the Repeat instruction. Remember that after the last address of available core storage has been generated in VAK or UAK, the address generated next is the first address of core storage. Then, suppose the address d0 above is a drum address, the address c0 is core address 07000, and n is 2000 (octal). If only one bank of core storage is available, the addresses 07000 through 07777 and addresses 00000 through 00777 would receive words from the drum. The instruction at F_1 (00000), containing the address w of the next instruction to be executed has been destroyed. Consider another case where c0 is 010000, n is (octal) 7000, and w is 07000. The addresses 010000 through 07777 would receive words from the drum. The instruction at F_1 has not been altered but the instruction originally at w has been destroyed.

2-80. The Repeat instruction is used advantageously in the accumulation of products, such as,

$$S = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n = \sum_{i=1}^{n} a_i b_i$$

The Multiply Add instruction MAuv adds the product of (u) and (v) to the number already in the Accumulator. Repeating the Multiply Add instruction as shown below generates the scalar product of two n-vectors.

Location	Op code	u address	v address	Explanation
dl	RS	A	A	Clear Accumulator to zero
d2	RP	3 n	d4	Form sum of products s = (ul)(vl)
d3	MA	ul	vl	+ (u2)(v2) +. . . + (un)(vn)
d4				Next instruction
ul			al	
. .			. .	vector a
. un			. a_n	
vl			b_1	
. .			. .	vector b
. vn			. bn	

2-81. UNCONDITIONAL JUMP INSTRUCTIONS

IP--

2-82. INSTRUCTION: Interpret

14--

FUNCTION: Let y represent the address from which the current instruction was obtained. Replace the v-address portion of (F_1) with y + 1. Then take (F_2) as the next instruction.

F_1 is storage address 00000 (or 40001) F_2 is storage address 00001.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	EXECUTION TIME REPEATED
Clear X	u and v irrelevant	
(PAK) i.e., y + 1 $\longrightarrow X_{14} \ldots X_0$	30	
$X_{14} \ldots X_0 \longrightarrow$ v address of (F_1)		Note that the complement of (jn-1) is transmitted to $X_{14} \ldots X_0$ if IP-- is preceded by RPjnw.
Set PAK to address F_2		

RJuv

2-83. INSTRUCTION: Return Jump 37uv

FUNCTION: Let y represent the address from which the current instruction was obtained. Replace the v-address portion of (u) with y + 1. Then take (v) as the next instruction.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	EXECUTION TIME REPEATED
Clear X (PAK) i.e., $y + 1 \rightarrow X_{14} \ldots X_0$ $v \rightarrow$ PAK (v of A gives SCC fault) $X_{14} \ldots X_0 \rightarrow u_{14} \ldots u_0$ (u of Q or A gives SCC fault)	v / u \ MC MC | 36	v / u \ MC MC | 24 Note that the complement of (jn-2) is transmitted to $X_{14} \ldots X_0$ if R juv is preceded by RP_{jnw}.

2-84. INSTRUCTION: Manually Selective Jump MJjv 45jv

FUNCTION: If j is 0, take (v) as the next instruction. If j is 1, 2, or 3, and a corresponding manual jump selection is made, take (v) as the next instruction. If a manual jump selection is not made, continue with the present sequence of instructions. A manual jump selection of 1, 2, or 3 is made by depressing a correspondingly numbered Select Jump button on the computer control panel. Depressing a Select Jump button illuminates a light located just above the button.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	EXECUTION TIME REPEATED
If a jump is indicated v ⟶ PAK (v of A gives SCC fault)	NO JUMP v | MC 18 JUMP v | irrelevant 18	NO JUMP v | MC 10n JUMP v | irrelevant 6

2-85. The Unconditional Jump instructions order the computer to execute next the instructions addressed as v. The jumps are unconditional in so much as the instructions do not order any testing or comparing of operands to determine if a jump should occur. The instruction RJuv is used primarily for entering subroutines where v is the entrance address of the subroutine and u is exit address of the subroutine. The content of u is usually a Manually Selective Jump instruction. Thus, if RJuv is at address y, a return to address y+1 occurs after executing the instruction MJ - y+1 at address u. Each time

this instruction at y is executed, a jump to address v occurs. However, if it is desired that the subroutine be entered by the Return Jump only the first time the Return Jump is executed, the instruction at address y can be coded initially as RJ y v. Hereafter, the execution of this instruction at y will cause only a "jump" to the next instruction.

2-86. The instruction IP-- can be interpreted as an RJuv where u and v are fixed as F_1 and F_2 respectively. This permits the programmer to use the u and v-address portions of the Interpret instruction to specify a pseudo instruction code. The subroutine entered by jumping to address F_2 can be coded to interpret this pseudo code.

2-87. The instructions IP-- and RJuv are never executed more than once when they are preceded by a Repeat instruction. This is true regardless of the number of repeats, n > 1, specified by the Repeat instruction. The instruction MJjv when j = 0 is never executed more than once. The instruction MJjv with j = 1, 2, 3, is repeated only if no corresponding manual selection of j=1, 2, 3, has been made.

2-88. CONDITIONAL JUMP INSTRUCTIONS

IJuv
4luv

2-89. INSTRUCTION: Index Jump

FUNCTION: Form in A the difference D(u) minus one. If (A) is then i.e., (u)≤0 negative, i.e. (u)≤0, continue the present sequence of instructions. If (A) is non-negative, i.e., (u) > 0 replace (u) with (A_R) and take (v) as the next instruction.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED		EXECUTION TIME REPEATED	
Clear X	v / u	NO JUMP irrelevant	v / u	NO JUMP irrelevant
(u)→X	MC	42	MC	34n
Omit if u is A				
Clear A	A	36	A	28n
Omit if u is A	Q	38	Q	30n
Add D(X) to (A)				
Subtract one from (A)	v / u	JUMP MC	v / u	JUMP MC
(A_R)→X	MC	52	MC	34r + 6
If A_{71} is zero				
v→PAK	A	38	A	28r - 2
(X)→u				
Omit if u is A	Q	42	Q	30r
			where r = number of executions necessary to satisfy the jump condition.	

TJuv

2-90. INSTRUCTION: Threshold Jump (Not repeated) 42uv

FUNCTION: Subtract D(u) from (A). If (A) is then negative, i.e., D(u)>(A), take (v) as the next instruction. If (A) is non-negative, continue the present sequence of instructions. Then in either case, restore (A) to its initial state.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	REPEATED
(u)→X Subtract D(X) from (A) V→PAK if A_{71} is one (v of A gives SCC fault) Add D(X) to A	NO JUMP v / u: irrelevant MC: 42 A: 36 Q: 38 JUMP v / u: MC MC: 42 A: 36 Q: 38	

2-91. INSTRUCTION: Threshold Jump, (Repeated). The operations below are preceded the first time by the operations of the Repeat instruction. See instruction RPjnw for these operations and the termination operations.

FUNCTION: Subtract D(u) from (A). If (A) is then negative, i.e., D(u)>(A), place j n-r in $Q_{14}...Q_0$, where $Q_{35}...Q_{15} = 0$, and take v as the next instruction. If (A) is non-negative, repeat the execution unless n-r = 0. In either case restore (A) to its initial state.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	EXECUTION TIME REPEATED
(u) → X Subtract D(X) from (A) If A_{71} is zero, Add D(X) to (A) If A_{71} is one, Complement (PAK) Add D(X) to (A) Perform Jump Termination		NO JUMP v / u: irrelevant MC: 30n A: 24n Q: 26n JUMP v / u: MC MC: 30r+10 A: 34 Q: 36 where r = number of executions necessary to satisfy the jump condition. If a jump is to occur, it will occur during the first execution when u is A or Q.

2-92. INSTRUCTION: Equality Jump (Not Repeated) EJuv 43uv

FUNCTION: Subtract D(u) from (A). If (A) is then zero, i.e., D(u)=(A), take (v) as the next instruction; if (A) is not zero, continue the present sequence. In either case, restore (A) to its initial state.

OPERATION SEQUENCE	EXECUTION TIME	
	NON-REPEATED	REPEATED
(u)→X	NO JUMP	
	u \ v: irrelevant	
Subtract D(X) from A	MC: 54	
If (A) is zero*		
v→PAK	A: 48	
(v of A gives SCC fault)	A: 50	
Add D(X) to (A)	JUMP	
	u \ v: MC	
	MC: 54	
	A: 48	
	Q: 50	

2-93. INSTRUCTION: Equality Jump, (Repeated). The operations below are preceded the first time by the operations of the Repeat instruction. See instruction RPjnw for these operations and the termination operations.

* The content of A is tested for a zero condition by subtracting one from (A) and noting whether an end-around borrow is propagated. If a borrow was propagated past A_{71}, (A) was all zeros. The one is of course added back in to A.

FUNCTION: Subtract D(u) from (A). If (A) is then zero, i.e., D(u)=(A), place j n-r in $Q_{14}...Q_0$, where $Q_{35}...Q_{15} = 0$, and take (v) as the next instruction; if (A) is not zero, repeat the execution unless n-r = 0. In either case, restore (A) to its initial state.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	EXECUTION TIME REPEATED

(u)→X
 	
Subtract D(X) from (A)
If (A) is not zero*
Add D(X) to (A)
 	
If (A) is zero*
Complement (PAK)
Add D(X) to (A)
Perform Jump Termination
		NO JUMP
		v / u: irrelevant
		MC: 42n
		A: 36n
		Q: 38n
		
		JUMP
		v / u: MC
		MC: 42r+10
		A: 46
		Q: 48
		
		where r = number of executions necessary to satisfy the jump condition. If the jump is to occur, it will occur during the first execution when u is A or Q.

* The content of A is tested for a zero condition by subtracting one from (A) and noting whether an end-around borrow is propagated. If a borrow was propagated past A_{71}, (A) was all zeros. The one is of course added back in to A.

2-94. INSTRUCTION: Q-Jump

QJuv
44uv

FUNCTION: If Q_{35} is one, take (u) as the next instruction; if Q_{35} is zero, take (v) as the next instruction. Then in either case, left circular shift (Q) by one place.

OPERATION SEQUENCE	EXECUTION TIME	
	NON-REPEATED	REPEATED
If Q_{35} is one, u→PAK	u or v	u or v
If Q_{35} is zero, v→PAK (u or v of A gives SCC fault)	MC 18	MC 6
Shift (Q) left one place		

2-95. INSTRUCTION: Sign Jump

SJuv
46uv

FUNCTION: If A_{71} is one, take (u) as the next instruction; if A_{71} is zero, take (v) as the next instruction.

OPERATION SEQUENCE	EXECUTION TIME	
	NON-REPEATED	REPEATED
If A_{71} is zero, v→PAK	u or v	u or v
If A_{71} is one, u→PAK	MC 18	MC 6
(u or v of A gives SCC fault)		

2-96. INSTRUCTION: Zero Jump

ZJuv
47uv

FUNCTION: If (A) is not zero, take (u) as the next instruction; if (A) is zero, take (v) as the next instruction.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	EXECUTION TIME REPEATED
v → PAK if (A) is zero*	u or v	u or v
u → PAK if A is not zero*	MC | 30	MC | 18
(u or v of A gives SCC fault)		

2-97. If a certain condition is met, the Conditional Jump instructions order the computer to execute next a specified instruction, the instruction located at address v. Any of the conditional jump instructions can be used as "decision" instructions.

2-98. The instructions Index Jump, Equality Jump, and Threshold Jump are one-way conditional jump instructions, i. e, a jump in one direction only is possible. The instruction IJuv is extremely useful in repeating an iterative cycle a prescribed number of times, n. If the Index Jump is executed before entering the loop to be repeated, the quantity n is initially stored in the "counter", (u). If the Index Jump is executed after each execution of the loop the quantity n-1 is initially stored in the counter. After the loop is repeated n times, the counter, (u), is left all zeros. Note that the Index Jump instruction destroys any initial content of the Accumulator.

2-99. The following general subroutine for adding two vectors illustrates uses of the instructions discussed thus far. The two vectors are x and y with coordinates x_i and y_i, respectively. The sum vector is z with coordinates z_i. Thus, it is necessary to form the sums

$$z_i = x_i + y_i, \quad i = 1, 2, 3, \ldots, n$$

Tabular values of x_i and y_i are stored in consecutive order somewhere in storage and tabular values of z_i are to be stored in consecutive order.

* The content of A is tested for a zero condition by subtracting one from (A) and noting whether an end-around borrow is propagated. If a borrow was propagated past A_{71}, (A) was all zeros. The one is of course added back into A.

Location	Op code	u address	v address	Explanation
d0	TP	h0	h5	set working counter to n-1
d1	TU	h1	d4	set location of x_1, (i = 1)
d2	TU	h2	d5	set location of y_1, (i = 1)
d3	TV	h1	d5	set location of z_1, (i = 1)
d4	TP	[ ]	A	transmit x_1 to Accumulator
d5	AT	[ ]	[ ]	form $x_1 + y_1 = z_1$, and transmit to g_1
d6	RA	d4	h3	Modify the instructions at d4 and
d7	RA	d5	h4	d5 so that i + 1 → i
d10	IJ	h5	d4	test working counter for n repeats
d11	MJ	0	[ ]	exit
h0				n-1
h1	00	e1	g1	where (ei) = x_i, i=1, 2, 3, ..., n
h2	00	f1	0	(fi) = y_i, i=1, 2, 3, ..., n (gi) = z_i, i=1, 2, 3, ..., n
h3	00	00001	00000	
h4	00	00001	00001	
h5				working counter

The Manually Selective Jump instruction at d11 provides an exit back to a main routine. If this vector add subroutine were entered by the instruction RJ d11 d0, arbitrarily located at address y, the execution of the instruction at d11 provides a jump back to address y + 1.

2-100. The Equality Jump and Threshold Jump instructions both test the contents of the Accumulator against a double length extension D(u). Such a test is not valid then if (A) is a split extension, S(u).

2-101. An overflow into A_{35}, as described in the arithmetic instructions, can be determined by executing the Equality Jump instruction with its u address referencing the Accumulator, e.g., EJ A v. The equality of $D(A_R)$ and (A) is tested by this instruction. If there has been an overflow into A_{35}, $D(A_R) \neq (A)$.

2-102. The instructions EJuv and TJuv are noteworthy because of the feature which restores A to its initial condition. However, if the initial content of A is all ones, a negative zero, the restoration process places all zeros in A. If (A) is all ones and an Equality Jump or Threshold Jump is executed, note the following peculiarities:

If (u) is $2^{36}-1$, (negative zero):

EJuv shows an equality, takes (v) as the next instruction
TJuv results in a positive (A), continues present sequence

If (u) is all zeros, (positive zero):

EJuv does not show an equality, continues sequence
TJuv results in a negative (A), takes (v) as the next instruction

2-103. The repeated instructions Equality Jump and Threshold Jump provide a count of the number of executions that were necessary to satisfy the jump condition. The count is placed in the Q-Register if the jump condition is met before n executions or on the $n^{\underline{th}}$ execution. The number n is the number of repeated executions specified by the Repeat instruction RPjnw. The quantity placed in the Q-Register is jn-r where r is the number of times the instruction was executed to find the jump condition. The quantity jn-r is the lower 15 bits of the Q register. (Q is clear except for this quantity.)

For example, if the instruction EJuv was preceded by the instruction RP 20010 w (j=2, n=8), and equality was found on the sixth execution, the contents of Q would be as follows.

jn = 010 000 000 001 000
jn-r = 010 000 000 000 010

(Q) = 000 000 000 000 000 000 000 010 000 000 000 010

2-104. Following is an example of a use for the repeated Equality Jump instruction. A file of quantities is stored at locations $u_1 \ldots u_n$. A table look-up is accomplished by testing each file item for equality with an item already in the Accumulator. When an equality is found with an item at address u_i, a jump is made to a subroutine starting at address v of EJuv. This subroutine is coded to acquire the quantity r, which is the number of times the equality test was made. The quantity r can be used to determine the location of the file item in the table. If no equalities are found, a jump is made to address w.

Location	Op code	u address	v address	Explanation
g1	RP	2n	w1	Check (u_1) ... (u_n) for
g2	EJ	u1	v1	equality with (A)
v1	TP	h1	A	Transmit 2n to A
v2	ST	Q	L1	Subtract 2 n-r from 2n store r at L1
h1	00	00000	2n	

2-105. The two-way conditional jump instructions order the computer to execute next either the instruction addressed as u or the instruction addressed as v. The direction of the jump is conditional upon the current content of the Accumulator or the Q Register. Note that in each instruction a zero condition, of Q_{35}, A_{71}, or (A), causes a jump to the v-addressed instruction.

2-106. These instructions are never executed more than once when preceded by a Repeat instruction, regardless of n of RPjnw being greater than one. The jump to either the u or v-addressed location on the first execution prevents any repeat of the instruction.

2-107. SCALE FACTOR INSTRUCTION

2-108. INSTRUCTION: Scale Factor

SFuv
74uv

FUNCTION: Replace A with D(u) unless u is A. Shift (A) until the most significant bit is at A_{34}. Then replace the v-address portion of (v) with the number of left shifts k which would be necessary to return (A) to its original position.

OPERATION SEQUENCE	EXECUTION TIME	
	NON-REPEATED	REPEATED
Clear X (u)→X omit if u is A Clear A omit if u is A Add D(X) to A Shift (A) left 36 places Continue shifting until A35 ≠ A34 Clear X (SK) = k→lowest order bits of X $X_{14} \ldots X_0 \rightarrow v_{14} \ldots v_0$ (v of A or Q gives SCC fault)	v / u: MC MC: 122 + 2γ A: 116 + 2γ Q: 118 + 2γ γ is non-negative and equal to 36-k mod 72, where 0≤71. ≤ For k = 37, use value of k = 38.	v / u: MC MC: (108 + 2γ)n+2

2-109. The Scale Factor instruction automatically scales a number to its maximum representation by 36 bits so that the most significant bit is positioned immediately to the right of the sign bit. The number is acquired from the u-address location and shifted in the Accumulator until the most significant bit is found in A_{34}. Then the number of left shifts necessary to restore the number to its original position in A is stored as the right-most bits of the v-address location.

2-110. If u of SFuv specifies MC or MD storage or the Q Register, D(u) is placed in A. If u specifies the Accumulator, the 72-bit number in A is retained. After a preliminary left shift 36 places of this initial content of A, $(A)_i$, shifting is continued until $A_{35} \neq A_{34}$. If the initial $A_{35} \neq A_{34}$, and $(A_L)_i$ has no significant bits, a shift count of k = 0 is derived. If (A) is all ones or all zeros, a shift count of k = 37 is derived.

2-111. The range of k, if u is MC, MD, or Q, is k = 0 or $37 \leq k \leq 71$. In this case, the most significant bit appears initially in A_R, and an end-around left shift exceeding 36 places is necessary to shift this bit from A_{34} back to its original position. If u is A, the range of k is $0 \leq k \leq 71$. In this case the most significant bit may appear initially in either A_R or A_L. If it appears in $(A_L)_i$, a left shift of k places, $k \leq 36$, shifts the most significant bit back to its original position.

2-112. Effectively, the shifting process forms the maximum 36-bit machine representation of (u) such that (u) = $(A_R)_f \cdot 2^{s2}$, or $(A)_i = (A)_f \cdot 2^{s2}$, if $(A_L)_i$ has no significant bits. (The term 2^{s2} has been defined as the scale factor in a discussion of scaling in the Stated Point Arithmetic section.) If $37 \leq k \leq 71$, the term $s2 = k-72$. In other words, when $37 \leq k \leq 71$, $(A)_f$ is $(A)_i$ shifted left 72-k places, or $(A_R)_f$ is (u) scaled 72-k places. If $0 \leq k \leq 36$, the term $s2 = k$. In this case, when $0 \leq k \leq 36$, $(A)_f$ is $(A)_i$ shifted "right" k places, or $(A_R)_f$ is (u) = $(A)_i$ scaled <u>down</u> k places. Note that care must be taken in using the 72-bit content of A, $(A)_f$, when k is in the range $0 < k \leq 36$, since $(A_L)_f$ may have bits which the computer interprets as the most significant.

2-113. Two examples are given below. The first has k in the range $37 < k \leq 71$; the second, $0 < k \leq 36$.

u = MC, MD, Q, or A:

(u) =	000 010 000 000 000 000 000 000 000 000 000 001
$(A)_i$ = 000...000	000 010 000 000 000 000 000 000 000 000 000 001
$(A)_f$ = 000...000	010 000 000 000 000 000 000 000 000 000 001 000

In this example $(A)_f$ is $(A)_i$ scaled three places = 72-k places where k = 69. $(A)_i = (A)_f \cdot 2^{-3}$ and (u) = $(A_R)_f \cdot 2^{-3}$.

u = A:

$(A)_i$ = 001 000...001	010 000 000 000 000 000 000 000 000 000 000 000 001
$(A)_f$ = 100 000...010	010 000 000 000 000 000 000 000 000 000 000 000 010

In this example, $(A)_f$ is $(A)_i$ scaled down 35 places where k = 35. However, $(A)_i \neq (A)_f \cdot 2^{35}$ because of the bits of $(A_R)_i$ now in $(A_L)_f$. $(A_R)_f \cdot 2^{35} = (A)_i$ to 36 bits of representation.

2-114. To illustrate the use of the Scale Factor instruction, consider the problem of normalizing the product of two numbers x_m and y_m, where x_m and y_m are machine representations of x and y in the form $x = x_m \cdot 2^p$ and $y = y_m \cdot 2^q$.

The normalized product is desired in $(A)_f$ in the form $x \cdot y = (A)_f \cdot 2^{p+q+s}$ where $x_m y_m = (A)_i = (A)_f \cdot 2^s$. The subroutine below stores this normalized product and also records the number of places, $-(p+q+s)$, the true product xy is scaled to form the normalized machine product. (It is assumed that $-(p+q)$ has been formed and stored previously.) The subroutine is as follows:

Location	Op code	u address	v address	Explanation
D1	MP	v1	v2	form product $x_m y_m$
D2	SF	A	L1	form $(x_m y_m)2^{-s}$ in A_R
D3	TP	A	L2	store normalized product
D4	TP	L1	A	place k in A
D5	TJ	L3	D7	jump to D7 if k<37
D6	RA	v3	L4	forms $-(p+q+s)$ in v3
D7	RS	v3	L1	
D10				next instruction
L1	00	00000	00000	Storage for k
L2	00	00000	00000	Storage for normalized product
L3	00	00000	00045	decimal 37
L4	00	00000	00110	decimal 72
v1				x_m
v2				y_m
v3				$-(p+q)$ initially

Note that in order to interpret correctly (L1) as k, the 21 left-most bits of L1 must be zeros. Note also that the routine must distinguish between $k \leq 36$ and $k \geq 37$.

Applying this subroutine to the example used in the paragraphs on scaling in the Stated Point Computer Arithmetic section, the following results are obtained:

If $(v1) = (2^{19}+\ldots) = x_m$

$(v2) = (2^{19}+\ldots) = y_m$

$(v3)_i = +40 \quad = -(p+q)$

then $k = 5 = s$ if $2^{40} > x_m y_m \geq 2^{39}$

$-(p+q+s) = 40-5 = 35$

or $k = 4 = s$ if $2^{39} > x_m y_m \geq 2^{38}$

$-(p+q+s) = 40-4 = 36$

2-115. STOP INSTRUCTIONS

MSjv
56jv

2-116. INSTRUCTION: Manually Selective Stop

FUNCTION: If j = 0, stop computer operation. If j = 1, 2, or 3, and a corresponding manual stop selection is made, stop computer operations. A stop selection of 1, 2, or 3, is made by depressing a correspondingly numbered Select Stop button on the computer control panel. Depressing a Select Stop button illuminates a light located just above the button. Whether or not a stop occurs, (v) is the next instruction unless (PAK) is changed before a re-start of computer operation.

OPERATION SEQUENCE	EXECUTION TIME	
	NON-REPEATED	REPEATED
v→PAK (v of A gives SCC fault) Stop computer operation if stop was selected	NO STOP v | MC 18 STOP v | irrelevant 4	NO STOP v | MC 6 STOP v | irrelevant Subtract 10 from the Repeat time, i.e., 54-10+p

PS--
2-117. INSTRUCTION: Program Stop 57--

FUNCTION: Stop computer operations and indicate by a red light, denoting a final stop, on the computer control panel.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	EXECUTION TIME REPEATED
Stop computer operation	u and v irrelevant 2	u and v irrelevant Subtract 12 from the Repeat time, i.e., 54-12+0

The Stop instructions order the computer to stop operation. The occurence of a stop is indicated by a red light denoting the type of stop, on the computer control panel.

2-118. INPUT OUTPUT INSTRUCTIONS

PR-v
2-119. INSTRUCTION: Print 61-v

FUNCTION: Transmit the right-hand six bits of (v) to TWR. The typewriter interprets the typewriter code in TWR, and prints a character or responds otherwise (spaces, shifts to upper case or lower case, etc).

OPERATION SEQUENCE	EXECUTION TIME* NON-REPEATED	EXECUTION TIME* REPEATED
(v)→X When the typewriter is ready (TWR is then cleared), transmit "1's" from $X_5 \ldots X_0$ to $TWR_5 \ldots TWR_0$ The typewriter interprets (TWR) and responds accordingly.	v | MC | A | Q 34 | 30 | 32 * These are computer operating times. The typewriter cycle time of approximately 105 ms can increase the overall time consumption substantially.	v | MC | A | Q 26n | 20n | 22n

2-120. INSTRUCTION: Punch

PUjv
63jv

FUNCTION: Transmit the right-hand six bits of (v) to HPR. The punch responds to (HPR) by punching its content. If j=0, seventh level hole is not punched; if j=1, the seventh level hole is punched.

OPERATION SEQUENCE	EXECUTION TIME*	
	NON-REPEATED	REPEATED
(v)→X When the punch is ready (HPR is then cleared), transmit "1's" from $X_5 \ldots X_0$ to $HPR_5 \ldots HPR_0$ If j=1, place "1" in HPR_6. Punch (HPR)	v | MC A Q 34 30 32 * These are computer operating times. The punch cycle time of 16.7 ms can increase the overall time consumption substantially.	v | MC A Q 26n 20n 22n

2-121. INSTRUCTION: External Function

EF-v
17-v

FUNCTION: As indicated by (v), select a unit of external equipment and instruct it to perform the designated operation. (A selective code is assumed to be stored at v.)

OPERATION SEQUENCE	EXECUTION TIME*	
	NON-REPEATED	REPEATED
(v)→X When previous operations involving IOB are completed (X)→IOB According to (IOB), select equipment and instruct it to function.	v / u | MC A Q 28 22 24 * These are computer operating times. Additional lockout times are possible.	

2-122. INSTRUCTION: External Read

ER
76jv

FUNCTION: If j=0, replace the right-most eight bits of (v) with (IOA) and the remaining bits of (v) with zeros. If j=1, replace (v) with (IOB).

This instruction must have been preceded by an External Function which instructed the equipment to transmit information to IOA or IOB.

OPERATION SEQUENCE	EXECUTION TIME*				
	NON-REPEATED			REPEATED	
	v	MC	A Q	v	MC
Clear X					
When IOA or IOB receives data from equipment: (IOA)→$X_7 \ldots X_0$ or (IOB)→X		30	28 24		16n+2
Clear IOA or IOB (X)→v	* These are computer operating times. Lockout times while the computer waits for data are possible.				
If v is A Clear A Add D(X) to A (v of MD gives SCC fault)					

2-123. INSTRUCTION: External Write

EWjv
77jv

FUNCTION: If j=0, replace (IOA) with the right most eight bits of (v). If j=1, replace (IOB) with (v). This instruction must have been preceded by an External Function which instructed the equipment to sense the information in IOA or IOB.

OPERATION SEQUENCE	EXECUTION TIME*					
	NON-REPEATED			REPEATED		
	v	MC	A Q	v	MC	A Q
(v)→X (v of MD gives SCC fault)		28	22 24		20n	14n 16n
When previous operations involving IOA or IOB are completed: $X_7 \ldots X_0$→IOA or (X)→IOB	* These are computer operating times. Additional lockout times are possible.					
External equipment senses (IOA) or (IOB)						

2-124. The Input Output instructions govern the transfer of information between computer storage and the external equipment. In the case of the punch and the typewriter, only one instruction is necessary to transfer a unit of data. Communication with the punch is through the High-speed Punch Register, HPR, a seven bit register. Communication with the typewriter is through the Typewriter Register, TWR, a six bit register.

2-125. The Punch instruction produces on paper tape a direct representation of the bits at referenced storage addresses. The Print instruction produces on a printed page the typewriter characters which are represented by the 6-bit typewriter codes at referenced storage addresses. Each Punch or Print instruction, PUjv or PR-v, uses only the right-most six bits of the location addressed as v. Therefore, any location v, where the bits $v_5 \ldots v_0$ are non-essential to the instruction at v, can be used for storing a 6-bit unit of information. For example, note the following use of available space in (v).

	PR	0	v	Print the number "1" (whose octal typewriter code is 52)
where (v) =	TP	u	31052, i.e.,	transmit (u) to Q
or (v) =	AT	u	32052, i.e.,	add D(u) to (A)
or (v) =	PS	-	00052, i.e.,	stop computer operation

2-126. The following routine causes the typewriter to print in octal one 36-bit word.

Location	Op code	u address	v address	Explanation
c0	TP	d4	d5	Set working counter
c1	PR	0	e10	Carriage return
c2	LQ	d1	3	Position word for printing
c3	QT	d2	A	Mask out 3 bits of word
c4	AT	d3	c5	Set up print instruction
c5	PR	0	ei	Print one octal digit
c6	IJ	d5	c2	Test for 12 prints
c7				Next instruction
d1				Word to be printed
d2	00	0	7	Mask
d3	PR	0	e0	Dummy print instruction
d4	0	0	13	n-1 (counter)
d5	0	0	0	working counter
e0	0	0	37	Typewriter codes for the digits 0 through 7
.	.	.	.	
.	.	.	.	
.	.	.	.	
e7	0	0	72	
e10	0	0	45	Typewriter carriage return code

2-127. Communication with external equipments, other than the typewriter and punch, is through the Input-Output Register A, IOA, or the Input-Output Register B, IOB. IOA is an eight bit register, and IOB is a 36 bit register. The External Function instruction uses the IOB register to establish communication with a particular piece of external equipment. Then, the IOA and IOB registers are used by External Write and External Read instructions to transfer units of data to or from this selected equipment. Data is transferred to the location addressed as v in the External Read instruction, ERjv, or from the location addressed as v in the External Write instruction, EWjv. If v is an MD address, a computer fault occurs and stops computer operation.

NOTE. The Input and Output section should be consulted for details on external equipment, and programming for external equipment.

2-128. FLOATING POINT INSTRUCTIONS

2-129. A packed floating point operand is defined as follows. For non-zero numbers, $x \cdot 2^y$, the mantissa x, is represented by 27 significant bits and a sign bit. The value of x is in the range $1/2 \leq |x| < 1$. The characteristic y is restricted to $-128 \leq y < 128$.

2-130. For machine representation, the characteristic is biased by the addition of 128 such that $0 \leq y + 128 < 256$. The biased characteristic is represented by bits u_{34} through u_{27} of register u.

2-131. The fractional mantissa is represented by bits u_{26} through u_0 of register u. The mantissa x is normalized with its most significant bit in u_{26} such that $2^{26} \leq |x \cdot 2^{27}| < 2^{27}$. The sign bit of a floating point number is represented by bit u_{35}. The value of zero is represented by all zeros in a computer register.

2-132. A negative floating point number is represented by the word which is the complement of the 36 bits denoting the packed number. Examples of packed floating point numbers in the computer are given below.

in register u	u_{35}	$u_{34} \ldots u_{27}$	$u_{26} \ldots u_0$
machine representation	sign bit	biased characteristic	mantissa
of number $x \cdot 2^y$ =	0	y + 128	$x \cdot 2^{27}$
of number $-(x \cdot 2^y)$ =	1	complement of (y + 128)	complement of $x \cdot 2^{27}$
of number 3 =	0	10 000 010	110 000 000 000 000 000 000 000 000
of number -(3) =	1	01 111 101	001 111 111 111 111 111 111 111 111
of zero =	0	00 000 000	000 000 000 000 000 000 000 000 000

2-133. Each floating point arithmetic instruction has the following major processes.

a. The operands are unpacked, with the mantissas being placed in the arithmetic registers appropriate for the arithmetic process to follow, and the characteristics being placed in special floating point storage registers. A packed negative floating point number has its representation unpacked in such a way that its biased characteristic, not the complement of its biased characteristic, is extracted and temporarily set aside in a storage register.

b. The appropriate arithmetic process is performed with the unpacked operands.

c. The mantissa is positioned in the Accumulator in preparation for the rounding process. The term pseudo-normalization is given to the process of adjusting the result of the arithmetic process such that the significant bits of the mantissa appear as bits $A_{61}...A_{35}$. Pseudo-normalization leaves the significant bits of the mantissa one place to the right of their final normalized position in A_L.
NOTE: A result with a mantissa equal to zero, if such occurs, is detected during the pseudo-normalization process. If such a mantissa is found, the processes of rounding, normalizing, and packing as described below, do not occur. The Accumulator and the Q Register are both cleared. Hence, the final content of Q and A is zero when the floating point arithmetic result has a mantissa of zero.

d. The non-zero mantissa of the result is rounded, at the programmer's option, to 27 significant bits. Rounding is accomplished by adding a one (or adding a negative one if the number is negative) to bit A_{34}. This rounds the least significant bit, A_{35}, of the pseudo-normalized number.

e. The result is normalized such that the most significant bit of the mantissa is A_{62} and the remaining significant bits appear as bits $A_{61}...A_{36}$. All of the bits $A_{71}...A_{63}$ are sign bits of the number. At the completion of the normalization, the representation in A_L of a non-zero mantissa x is in the range $2^{26} \leq |x \cdot 2^{27}| < 2^{27}$.

f. The packed representation of a non-zero result is placed in the Q Register. The mantissa is obtained from $A_{71}...A_{36}$, and the characteristic is obtained from a special floating point register. The result is packed in the X Register, according to the conventions stated previously for the representation of positive and negative floating point numbers. The content of x is then transmitted to the Q Register.

Before the mantissa and the characteristic are obtained for the packing procedures, the size of the biased characteristic is checked. The biased characteristic of the result is checked for the range $256 > y+128 \geq 0$. If $y+128 < 0$, the final contents of the Accumulator and the Q Register are zero, since A and Q are cleared. If $y+128 > 255$, where $2^{26} \leq |x \cdot 2^{27}| < 2^{27}$, a computer fault occurs. The result cannot be packed in this case. Computer operation is stopped, and a Characteristic Overflow fault is indicated on the computer control panel.

The same check on the size of the biased characteristic is made on the intermediary product derived by the instructions Floating Point Polynomial Multiply and Floating Point Inner Product. If the biased characteristic is (1) less than zero, the product is regarded as zero, or (2) greater than 255, a computer fault stop occurs.

2-134. The Floating Point instructions are presented subsequently. In the tabular and text presentations, the processes of unpacking, pseudo-normalization, rounding, normalization, and packing are mentioned only briefly. A

reference to these processes connotes the details of the processes as discussed above. These details must be remembered in order to fully interpret the abbreviated OPERATION SEQUENCE tables.

2-135. The following conventions must also be observed in reading the OPERATION SEQUENCE tables.

a. (u) or (v) is the packed representation of a positive or negative floating point operand, $x \cdot 2^y$. This representation has been described previously.

b. $(X)_m$, where the operand is positive, has

$$X_{26} \ldots X_0 = x \cdot 2^{27}$$
$$X_{35} \ldots X_{27} = 000\ 000\ 000$$

c. $(X)_m$, where the operand is negative, has

$$X_{26} \ldots X_0 = \text{complement of } -x \cdot 2^{27}$$
$$X_{35} \ldots X_{27} = 111\ 111\ 111$$

d. $(X)_c$, where the operand is either positive or negative, has

$$X_{34} \ldots X_{27} = y+128$$
$$X_{35} = 0$$
$$X_{26} \ldots X_0 = \text{all zeros}$$

FRj-
2-136. INSTRUCTION: Floating Point Round Option 05j-

FUNCTION: If j = 1, cause the mantissa of the result of all subsequent floating point arithmetic processes to be normalized without rounding. If j = 0, return to the normal process of rounding the mantissa before normalizing. The condition which causes the omission of the rounding remains in effect until FRj- with j = 0 is executed or until a computer Master Clear is effected.

OPERATION SEQUENCE		EXECUTION TIME	
		NON-REPEATED	REPEATED
j = 1 cause the omission of rounding	j = 0 return to the normal process of rounding	18	

After the execution of the FRj- with j = 1, a condition is established such that each subsequently executed floating point instruction has the mantissa of each result of an arithmetic operation normalized without rounding. This condition prevails until the execution of FRj- with j = 0, or until any program restart preceded by the depression of the computer Master Clear button. If either of

these occurs, the normal condition which provides the automatic round is re-established. The execution time of a floating point instruction is not increased to accommodate the rounding process.

2-137. INSTRUCTION: Floating Point Add

FAuv
64uv

FUNCTION: Form in Q the normalized, rounded (optional), and packed floating point sum of (u) and (v).

OPERATION SEQUENCE	EXECUTION TIME	
	NON-REPEATED	REPEATED
(u)→X Clear A and Q Unpack (X) $(X)_m$→X and Q Add D(X) to A	u \ v: MC MC: 156 A: 150	u \ v: MC
(v)→X Unpack (X) $(X)_m$→X	Q: 152	Q: 140n
	each time, repeated and/or non-repeated, plus	
If $(u)_c < (v)_c$ Clear A Add D(X) to A (Q)→X	$2\,u_{35} + 2\left\|(u)_c - (v)_c\right\| + 4s$, and + 12 if $(u)_c < (v)_c$	
	microseconds for each execution,	
Align mantissas Add D(X) to A	where $(u)_c$ = biased characteristic of (u)	
Pseudo-normalize mantissa sum in A	$(v)_c$ = biased characteristic of (v)	
	s = shift count necessary to pseudo-normalize the sum of the mantissas.	
Clear X and Q	The range of s is $0 \leq s \leq 35$. If s = 35, deduct 16 microseconds.	
If A ≠ 0, round (optional) bit A_{35}	NOTE: if $\left\|(u)_c - (v)_c\right\| \geq 2$, then $s \leq 1$.	
Normalize non-zero mantissa sum in A_L		
Pack non-zero operand sum in X characteristic too small, Clear A and X characteristic too large, computer fault stop (X)→Q		

2-138. INSTRUCTION: Floating Point Subtract

F Suv
65uv

FUNCTION: Form in Q the normalized, rounded (optional), and packed floating point difference of (u) and (v).

OPERATION SEQUENCE	EXECUTION TIME	
	NON-REPEATED	REPEATED
(u) → X Clear A and Q Unpack (X) (X)$_m$ → X and Q Add D(X) to A	v / u: MC: 156; A: 150; Q: 152	v / u: MC; Q: 140n
(v) → X Unpack (X) (X)$_m$ → X		
If $(u)_c < (v)_c$ Clear A Add D(X) to A (Q) → X	each time, repeated and/or non-repeated, plus $2\,u_{35} + 2\left\|(u)_c - (v)_c\right\| + 4s$, and + 12 if $(u)_c < (v)_c$ microseconds for each execution,	
Align mantissas Subtract D(X) from A	where $(u)_c$ = biased characteristic of (u)	
Pseudo-normalize mantissa difference in A	$(v)_c$ = biased characteristic of (v)	
Clear X and Q	s = shift count necessary to pseudo-normalize the difference of the mantissas.	
If A ≠ 0, round (optional) bit A_{35}	The range of s is $0 \leq s \leq 35$. If s = 35, deduct 16 microseconds.	
Normalize non-zero mantissa difference in A_L	NOTE: if $\left\|(u)_c - (v)_c\right\| \geq 2$, then $s \leq 1$.	
Pack non-zero operand difference in X characteristic too small, Clear A and X characteristic too large, computer fault stop (X) → Q		

2-139. The equation for the formation of the sum (difference) of two floating point numbers is

$$x \cdot 2^y \pm r \cdot 2^t = (x \pm r \cdot 2^{t-y})\ 2^y$$

The arithmetic processes of the Floating Point Add (Subtract) instruction must thus accomplish the following:

(1) Align the mantissas of the operands in accordance with the value of their characteristics.

(2) add (subtract) the aligned mantissas.

The sum (difference) of the mantissas resulting from the arithmetic process is one of the following:

$x \pm r \cdot 2^{t-y}$	if $y-t \leq 34$
x	if $y-t > 34$
$\pm r$	if $t-y > 34$
$x \pm r$	if $t-y = 0$

2-140. If the difference in the value of the characteristics is found to be less than or equal to 34, as is the case in the formation of the sum (difference) $x \pm r \cdot 2^{t-y}$ and $x \pm r$, the sum (difference) of the aligned mantissas is formed in the Accumulator. This sum (difference) is then pseudo-normalized by left shifting (A) until A_{61} is the most significant bit. If 35 left shifts occur without finding a significant bit, the sum (difference) of the mantissas is zero, and the final content of the Accumulator and the Q register is zero.

2-141. If the difference in the value of the characteristics is greater than 34, the smaller of the two numbers is too small to change the value of the larger number by addition (subtraction). If this is the case, the larger of the two mantissas, x or r as shown above, is pseudo-normalized in A_L.

2-142. If the mantissa of the result is determined to be non-zero, the process of rounding, (optional), normalization, and packing follow.

2-143. INSTRUCTION: Floating Point Multiply

FMuv
66 uv

FUNCTION: Form in Q the normalized, rounded (optional), and packed floating point of (u) and (v).

OPERATION SEQUENCE	EXECUTION TIME

(u) ⟶ X
Clear Q
Unpack (X)
  $(X)_m$ ➝ X
  If (u) is negative, complement (X)
  (X) ⟶ Q

(v) ⟶ X
Clear A
Unpack (X)
  $(X)_m$ ➝ X
  if (u) is negative, complement (X)

Form the product of (Q) and (X) in A

Shift (A) left eight places
Pseudo-normalize mantissa product in A

Clear X and Q

If (A) ≠ 0, round (optional) bit A_{35}

Normalize non-zero mantissa product in A_L

Pack non-zero operand product in X
  Characteristic too small, Clear A and X
  Characteristic too large, Computer fault stop
  (X) ⟶ Q

NON-REPEATED

u \ v	MC	A
MC	170	164
A	164	158
Q	166	160

REPEATED

u \ v	MC
Q	154n

each time, repeated and non-repeated, plus

$$8 \sum_{i=1}^{26} Q_i + 4\,Q_0 + 4s$$

microseconds for each execution,

where s = shift count necessary to pseudo-normalize the product of the mantissas. The range of s is $0 \leq s \leq 2$. If s = 2, deduct 16 microseconds.

2-144. The equation for the formation of the product of two floating point numbers is

$$(x \cdot 2^y) \cdot (r \cdot 2^t) = x \cdot r \cdot 2^{y+t}$$

The arithmetic processes of the Floating Point Multiply instruction must thus accomplish the following.

(1) multiply the mantissas of the operands

(2) add the characteristics of the operands

2-145. The mantissas are multiplied according to the multiplication process for stated point numbers. The multiplier (Q) is always positive. If the operand (u) is the representation of a negative number, the mantissa portion of (u), not its complement, is placed in Q with sign bits of zero. Then, the complement of the mantissa portion of (v) and the complement of the sign bits of (v) are used as multiplicand in the X Register.

2-146. During pseudo-normalization, no more than a shift count of two is necessary to either place the most significant bit of the product in A_{61}, or determine if the product is zero. (The product is always left shifted a fixed amount of eight places before a shift counter is set up.) If a shift counter set to the value of two is reduced to zero, the product is known to be zero. In this case, the final content of the Q Register and the Accumulator is zero. If the product is determined to be non-zero, the processes of rounding (optional), normalization, and packing follow pseudo-normalization.

2-147. INSTRUCTION: Floating Point Divide

FDuv
67uv

FUNCTION: Form in Q the normalized, rounded (optional), and packed floating point quotient obtained by dividing (u) by (v).

OPERATION SEQUENCE	EXECUTION TIME	
	NON-REPEATED	REPEATED

(u) → X
Clear A
Unpack (X)
  $(X)_m$ → X
  if (u) is negative, complement (X)
  Add D(X) to A

(v) → X
Unpack X
  $(X)_m$ → X
  if (u) is negative, complement (X)

Clear Q
Shift (A) left 34 places
Divide (A) by (X) forming the quotient in Q.
  During the process, check for a Divide Fault.

(Q) → X
Clear (A)
Add D(X) to A
Shift (A) left 27 places

Pseudo-normalize quotient in (A)

Clear X and Q

If A ≠ 0, round (optional) bit A_{35}

Normalize non-zero mantissa quotient in A_L

Pack non-zero operand quotient in X
  Characteristic too small, Clear A and X
  Characteristic too large, computer fault stop
  X → Q

v / u	MC	Q
MC	650	646
A	644	640
Q	646	642

v / u	MC	Q
Q	634n	630n

each time, repeated and non-repeated, plus

$$4s + 6R$$

microseconds for each execution,

where s = shift count necessary to pseudo-normalize the quotient of the mantissas. The range of s is $0 \leq s \leq 2$. If s = 2, deduct 16 microseconds.

R = 0 or 1. These six additional microseconds are sometimes necessary to yield a non-negative remainder.

2-148. The equation for the formation of the quotient of two floating point numbers is

$$(x \cdot 2^y) \div (r \cdot 2^t) = (x \div r)\ 2^{y-t}$$

The arithmetic processes of the Floating Point Divide instruction must thus accomplish the following.

(1) divide the mantissa of the dividend by the mantissa of the divisor

(2) subtract the characteristic of the divisor from the characteristic of the dividend.

2-149. The mantissas are divided according to the division process for stated point numbers. Before the division of (A) by (X) occurs, the mantissa and the sign bits of the dividend in A are left shifted 34 places. The dividend in the Accumulator is always positive. If the operand (u) is the representation of a negative number, the mantissa portion of (u), not its complement, and sign bits of zero are added to (A) cleared. Then, the complement of the mantissa portion of (v) and the complement of the sign bits of (v) are used as the divisor in the X Register. The quotient is formed in the Q Register and a non-negative remainder is formed in the Accumulator. However, this remainder is destroyed later when the quotient is transmitted to the cleared Accumulator for the processes of pseudo-normalization, etc. As in stated point division, a Divide Overflow fault is possible. If such occurs, computer operation is stopped, and the fault is indicated on the computer control panel.

2-150. During pseudo-normalization of the quotient in the Accumulator, no more than a shift count of two is necessary to either place the most significant bit of the quotient in A_{61}, or determine if the quotient is zero. (The quotient is always shifted a fixed amount of 27 places before a shift counter is set up.) If a shift counter set to the value of two is reduced to zero, the quotient is known to be zero. In this case, the final content of the Q Register and the Accumulator is zero. If the quotient is determined to be non-zero, the processes of rounding (optional), normalization, and packing follow pseudo-normalization.

2-151. INSTRUCTION: Floating Point Polynomial Multiply — FPuv / 0 1 uv

FUNCTION: Form the normalized, rounded (optional*) floating point product of (u) and $(Q)_i$. Then form in Q the normalized, rounded (optional*), and packed floating point sum of $(u)(Q)_i$ and (v).

OPERATION SEQUENCE	
(Q) $\longrightarrow$ X Clear Q Unpack (X) $(X)_m \longrightarrow X$ if (Q) initial is negative, complement (X) (X) $\longrightarrow$ Q	
(u) $\longrightarrow$ X Clear A Unpack (X) $(X)_m \longrightarrow X$ if (Q) initial was negative, complement (X)	
Form the product of (Q) and (X) in A Shift (A) left eight places Pseudo-normalize mantissa product in A Clear X and Q If (A) $\neq 0$, round (optional*) bit A_{35} Normalize non-zero mantissa product in A_L	formation of normalized product
Characteristic too large, computer fault stop Characteristic too small, Clear A and X Omit next four lines.	
$(A_L) \longrightarrow X$ (X) $\longrightarrow$ Q Clear A Add D(X) to A	
(v) $\longrightarrow$ X Unpack (X) $(X)_m \longrightarrow X$	
If $(A_R)_c < (v)_c$ Clear A Add D(X) to A (Q) $\longrightarrow$ X	
Align mantissas Add D(X) to A Pseudo-normalize mantissa sum in A Clear X and Q If (A) $\neq 0$, round (optional*) bit A_{35} Normalize non-zero mantissa sum in A_L	formation of normalized sum
Pack non-zero operand sum in X Characteristic too small, Clear A and X Characteristic too large, Computer fault stop	

* The round option here is that both the product and sum are rounded, or neither the product nor the sum is rounded.

EXECUTION TIME	
NON-REPEATED	REPEATED

v / u	MC
MC	290
A	284

v / u	MC
MC	278n

each time, repeated or non-repeated, plus

$$8 \sum_{j=1}^{26} Q_j + 4Q_0 + 4s_1$$

where $Q_{0,j}$ is from complement of $(Q)_m$ initial if multiplier (Q) initial is negative

and s_1 = shift count necessary to pseudo-normalize the product of the mantissas

$$+ 2\left|\left((u)(Q)_i\right)_c - (v)_c\right| + 4s_2, \text{ and } +12 \text{ if } \left((u)(Q)_i\right)_c < (v)_c$$

microseconds for each execution,

where $\left((u)(Q)_{i\ c}\right)$ = biased characteristic of one addend

and $(v)_c$ = biased characteristic of another addend

and s_2 = shift count necessary to pseudo-normalize the sum of the mantissas

Range of s_1 is $0 \leq s_1 \leq 2$. If $s_1 = 2$, deduct 16μs
Range of s_2 is $0 \leq s_2 \leq 35$. If $s_2 = 35$, deduct 16μs

NOTE: if $\left|\left((u)(Q)_i\right)_c - (v)_c\right| \geq 2$, then $s_2 \leq 1$

2-152. The Floating Point Polynomial Multiply instruction produces results according to those equations mentioned previously for the formation of floating point products and sums. The remarks made in the discussion of the Floating Point Multiply instruction apply also to the multiplication part of the FPuv instruction. One difference is that (u) is not the multiplier for this instruction, but rather the operand initially in Q is treated as the multiplier.

2-153. The processes of multiplication, pseudo-normalization, rounding, and normalization of the product are the same as in the instruction FMuv. When the product has been formed, the normalized mantissa in A_L and the adjusted characteristic in a special floating point register, these quantities are treated as the first addend, (u), in the instruction FAuv. (If the biased characteristic of the product is less than zero, the Accumulator is cleared as is the special floating point register. The first addend is thus zero. If the biased characteristic is greater than 255, a computer fault stop occurs at this point.) The operand (v) is then unpacked as the second addend, and the remarks made in the discussion of the Floating Point Add instruction are applicable. The processes of alignment, addition, pseudo-normalization, rounding, normalization, and packing of the sum are now the same as in the FAuv instruction.

2-154. Function evaluations and the closely related problem of polynomial evaluation occur frequently in computation. An examination of a program to perform the evaluation of $a_n x^n + \ldots + a_1 x + a_0$ in floating point indicates the value of the Floating Point Polynomial Multiply instruction. The following instructions evaluate the expression

$$\{[(a_n \cdot x + a_{n-1})\ x + a_{n-2}]\ x + a_{n-3}\}x + \ldots$$

Location	Op code	u address	v address	Explanation
RO	TP	t0	Q	Transmit the quantity a_n to the Q Register
R1	RP	1 n-1	w	} — Form $\Sigma a_n x^n$ in Q
R2	FP	s0	t1	
s0			x	
t0			a_n	
t1			a_{n-1}	
t2			a_{n-2}	
t3			a_{n-3}	
.			.	
.			.	
.			.	

2-155. INSTRUCTION: Floating Point Inner Product FIuv 02uv

FUNCTION: Form the normalized, rounded (optional*), floating point product of (u) and (v). Then form in Q the normalized, rounded (optional*), and packed floating point sum of (u)(v) and $(Q)_i$. The location F_4, 00003, is used for temporary storage of $(Q)_i$.

OPERATION SEQUENCE	
$(Q) \longrightarrow X$	
$(X) \longrightarrow F_4$	
$(u) \longrightarrow X$	
Clear Q	
Unpack (X)	
$(X)_m \longrightarrow X$	
if (u) is negative, complement (X)	
$(X) \longrightarrow Q$	
$(v) \longrightarrow X$	
Clear A	
Unpack (X)	
$(X)_m \longrightarrow X$	
if (u) is negative, complement (X)	
Form the product of (Q) and (X) in A	formation of normalized product
Shift (A) left eight places	
Pseudo-normalize mantissa product in A	
Clear X and Q	
If (A) $\neq$ 0, round (optional*) bit A_{35}	
Normalize non-zero mantissa product in A_L	
Characteristic too large,	
Computer fault stop	
Characteristic too small,	
Clear A and X	
Omit next four lines	
$(A_L) \longrightarrow X$	
$(X) \longrightarrow Q$	
Clear A	
Add D(X) to A	
$(F_4) \longrightarrow X$	
Unpack (X)	
$(X)_m \longrightarrow X$	
If $(A_R)_c < (Q)_c$	
Clear A	
Add D(X) to A	
$(Q) \longrightarrow X$	

*The round option here is that both the product and sum are rounded, or neither the product nor the sum is rounded.

OPERATION SEQUENCE

Align mantissas
Add D(X) to A
Pseudo-normalize mantissa sum in A
Clear X and Q
If (A) $\neq$ 0, round (optional*) bit A_{35}
Normalize non-zero mantissa sum in A_L

} formation of normalized sum

Pack non-zero operand sum in X
    Characteristic too small,
        Clear A and X
    Characteristic too large,
        Computer fault stop
    (X) ⟶ Q

EXECUTION TIME

NON-REPEATED			REPEATED	
u \ v	MC	A	u \ v	MC
MC	312	306	MC	300n
A	306	300		
Q	308	302	Q	296n

each time, repeated or non-repeated, plus

$$8 \sum_{j=1}^{26} u_j + 4u_0 + 4s_1$$

where u_0, j is from complement of $(u)_m$ initial if multiplier (u) is negative

s_1 = shift count necessary to pseudo-normalize the product of the mantissas

$$+2 \left| \left((u)(v)\right)_c - \left((Q)_i\right)_c \right| + 4s_2, \text{ and}$$

$$+12 \text{ if } \left((u)(v)\right)_c < \left((Q)_i\right)_c$$

microseconds for each execution,

where $\left((u)(v)\right)_c$ = biased characteristic of one addend

$\left((Q)_i\right)_c$ = biased characteristic of another addend

s_2 = shift count necessary to pseudo-normalize the sum of the mantissa

Range of s_1 is $0 \leq s_1 \leq 2$ If $s_1 = 2$, deduct 16μs
Range of s_2 is $0 \leq s_2 \leq 35$ If $s_2 = 35$, deduct 16μs

NOTE: if $\left|\left((u)(v)\right)_c - \left((Q)_i\right)_c\right| \geq 2$, then $s_2 \leq 1$.

*The round option here is that both the product and sum are rounded or neither the product nor the sum is rounded.

2-156. The Floating Point Inner Product instruction produces results according to those equations mentioned previously for the formation of floating point sums and products.

2-157. The remarks made in discussion of the Floating Point Multiply instruction apply also to the multiplication part of the Floating Point Inner Product instruction. The process of unpacking the operands, multiplication, pseudo-normalizing, rounding, and normalizing are the same as in the instruction FMuv. (Before any of these processes, the content of Q is stored at location F_4.) When the product has been formed, the normalized mantissa in A_L and the adjusted characteristic in a special floating point register, these quantities are treated as the first addend, (u), in the instruction FAuv. (If the biased characteristic of the product is less than zero, the Accumulator is cleared as is the special floating point register. The first addend is thus zero. If the biased characteristic is greater than 255, a computer fault stop occurs at this point.) The initial content of Q is now acquired from its storage at F_4 and treated as the second addend. The remarks made in the discussion of the Floating Point Add instruction are now applicable since the operations of alignment, addition, pseudo-normalization, rounding, normalization, and packing are those of the FAuv instruction.

2-158. The Floating Point Inner Product instruction finds use in matrix multiplication, vector inner products, smoothing, the evaluation of bilinear forms, and the like. All of these functions are of the sort $a_1b_1 + a_2b_2 + \ldots + a_nb_n$.

Location	Op code	u address	v address	Explanation
R0	TP	L1	Q	Clear the Q Register
R1	RP	3 n-1	w	} Form $\sum a_n b_n$ in Q
R2	FI	M1	N1	
L1			0	
M1			a_1	
M2			a_2	
. . .			. . .	
. N1			. b_1	
N2 . .			b_2 . .	
.			.	

NPuv

2-159. INSTRUCTION: Floating Point Normalize Pack 04uv

FUNCTION: Replace (u) with the normalized, rounded (optional), and packed floating point number, the biased characteristic of which is $v_{34}...v_{27}$ and the mantissa of which is $(u)_i$. The word $(u)_i$ may be unnormalized, i.e., the most significant bit can be any (or none) of the bits $u_{34}...u_0$. The binary point is considered to be between u_{27} and u_{26}.

OPERATION SEQUENCE	EXECUTION TIME						
	NON-REPEATED				REPEATED		
(v) ⟶ X (X) $_c$ ⟶ special floating point register	v / u	MC	A	Q	v / u	MC	Q
	MC	142	136	138	MC	128n+2	122n+2
(u) ⟶ X	A	134	128	130			
Clear A Add D(X) to A	Q	132	126	128			
Clear X and Q							
Shift (A) left 27 places Pseudo-normalize (A)	each time, repeated and non-repeated, plus						
	4s						
If (A) ≠ 0, round (optional) bit A_{35}	microseconds for each execution,						
Normalize non-zero mantissa in A_L	where s = shift count necessary to pseudo-normalize the mantissa						
Pack non-zero operand in X Characteristic too small, Clear A and X Characteristic too large, Computer fault stop (X) ⟶ Q	The range of s is $0 \leq s \leq 35$. If s = 35, deduct 16 microseconds.						
(X) ⟶ u if u is A Clear A Add D(X) to A							

2-160. An example of the use of the NP instruction is as follows. Suppose that a program is to be run, with packed normalized (rounded) floating point constants in storage locations m_1 to m_n. Instead of normalizing, rounding, and packing these numbers manually, the program can put the characteristics in the characteristic positions of memory location m_{n+1} to m_{2n}, and the mantissas in locations m_1 to m_n. Then to pack these numbers, the programmer would use the two instructions:

RP 3 n
NP m_1 m_{n+1}

This results in packing all of the constants into m_1 to m_n.

UPuv
03uv

2-161. INSTRUCTION: Floating Point Unpack

FUNCTION: Unpack (u). Replace (u) with $(u)_m$, and replace (v) with $(u)_c$.

OPERATION SEQUENCE	EXECUTION TIME NON-REPEATED	REPEATED
(u) ⟶ X Unpack (X) (X)$_c$ ⟶ special floating point register (X)$_m$ ⟶ X (X) ⟶ u if u is A, Clear A Add D(X) to A Clear (X) Biased characteristic from special floating point register ⟶ $X_{34} \ldots X_{27}$ (X) ⟶ v if v is A Clear A Add D(X) to A	u \ v: MC / A / Q MC: 60 / 58 / 54 A: 54 / 52 / 48 Q: 52 / 50 / 48	

2-162. PROGRAM INTERRUPT

2-163. The Program Interrupt in the Univac Scientific Computing System provides a means of breaking into a running program and, with appropriate coding, retaining a record of the particular point at which the "interruption" occurred. The interrupt may be initiated either by manual selections at the computer control panel or by appropriate procedures during input and output of data.

2-164. The interrupt signal, from either a manual or programmed source, becomes effective on main pulse six of the first Normal Termination Sequence to be attempted after the interrupt option is chosen. The Normal Termination Sequence is the sequence of events which concludes all non-repeated instructions and acquires the next instruction to be executed. The effect of the interrupt signal on the Normal Termination Sequence is as follows.

NORMAL TERMINATION SEQUENCE

without interrupt signal	with interrupt signal
1. Transmit the address in PAK to SAR; advance the content of PAK	1. Transmit the address 00002 to SAR
	2. Read to the X Register the content of the location whose address is (SAR)
	3. Transmit the content of the X Register to PCR.

Note that when an interrupt is effected, the content of the Program Address Counter, PAK, is not disturbed.

2-165. The next instruction executed is the instruction now held in the Program Control Register, PCR. If the interrupt was effected, PCR contains the instruction stored at address 00002 (F_3). If the content of 00002 is a Return Jump instruction,

$$(00002) = RJ\ u_1\ v_1,$$

the address in PAK is transmitted to the v-address portion of the instruction at u_1, and effectively set aside for future reference. The execution of the Return Jump instruction is followed by the execution of the instruction stored at v_1.

2-166. The instruction at address u_1 might be a jump instruction ordering a jump to its v address, as

$$(u_1) = MJ\ -\ v,$$

or after the execution of the instruction RJ u_1 v_1, $(u_1) = MJ\ -\ (PAK)$.

Then, the execution of the instruction at u_1 effects a return back to the program which was "interrupted," at the point at which the interrupt occurred: the instruction at the address formerly held in PAK is now executed as it would have been if the interrupt had not occurred.

2-167. Coding such that the contents of address 00002 and u_1 are as mentioned above, probably provides the most obvious manner of using the interrupt feature.

2-168. The interrupt signal is not as immediately effective, if at the time the interrupt is initiated, a Repeat instruction is being executed, or an instruction is being repeated under the direction of a Repeat instruction. Remember that a repeated instruction, unless it is a jump instruction and the jump condition is satisfied, is not concluded by the Normal Termination Sequence. (The termination sequences concluding a repeated instruction are listed in the discussion of the Repeat instruction in the Instruction Repertoire section.)

2-169. If the repeated instruction is concluded by the Normal Repeat Termination Sequence, the interrupt is not effective until the termination of the instruction at F_1. In this case, the instruction at F_1 determines which address is left undisturbed in PAK. If (F_1) is a jump instruction ordering a jump to its v address, such as MJjv, this v address is left in PAK. In this case the v address would be the address w of the Repeat instruction, RPjnw.

2-170. If a jump instruction is being repeated, and a jump condition is satisfied, it is expected that the next instruction to be executed is specified by the u or v address of the repeated instruction. The u or v address of the jump instruction, then, is the address left undisturbed in PAK. A repeated jump instruction which has satisfied the jump condition is concluded either by the Normal Termination Sequence or by the Jump (Repeat) Termination Sequence which has as a part of it the Normal Termination Sequence.

section 3
OPERATING THE COMPUTER

3-1. SUPERVISORY CONTROL PANEL.

3-2. The computer control panel, termed the Supervisory Control Panel, is shown in Figure 3-1. Computer operations can be manually superintended from the Supervisory Control Panel. By following the proper manual operation procedures, instructions and data can be extracted individually from storage. Also, an instruction may be manually inserted in the computer at the control panel and then executed.

3-3. The Supervisory Control Panel provides a visual display of the internal conditions of the computer. Some of the computer registers which are represented on the control panel are the Accumulator, the Q Register, the X Register, Input-Output Registers A and B, the Program Control Register (MCR, UAK, and VAK), the Program Address Counter and the Storage Address Register. Registers are represented by a double row of lights. A light in the upper row indicates a "1" in a particular bit position; a light in the lower row indicates a "0" in a particular bit position. Depressing the small button below each pair of lights places a "1" in that position of the register. Depressing the small button at the lower right end of each register places a "0" in every bit position of the register.

3-4. A Monitoring Oscilloscope is located above the center section of the control panel. The oscilloscope displays a 64x64 array which provides visual indication of references to addresses in Magnetic Core Storage.

3-5. Most of the operating selections are made by depressing buttons in sets of indicators shown on the lower portion of the center section of the control panel. The indicators and buttons are grouped as follows, from left to right.

Operating Rate Group - includes indicators to show the selection of computer operating rate. Manual selections must be made in this group to provide other than high-speed operation.

Selective Jumps Group - the j selection for the instruction Manually Selective Jump MJjv is made here.

Selective Stops Group - the j selection for the instruction Manually Selective Stop MSjv is made here. All of the computer operation stops caused by the MSjv or Program Stop (final stop) instructions are indicated here.

Interrupt Control Group - provides the means of a manual program interrupt.

Operating Group - includes indicators to show if the computer is in operation and, if so, the mode of operation. The computer is set in operation by depressing the START button and possibly the STEP button in this group. Depression of the MASTER CLEAR button clears, among others, the computer registers A, Q, IOA, IOB, PCR, and SAR. In general, a Master Clear restores computer control components to their cleared state. Operating selections made manually before a Master Clear must be re-made before operation can be resumed.

B Fault Group and A Fault Group - provides indicators to show the occurrence of a computer fault.

3-6. Additional operating selections are sometimes made in the Test Switch Group on the right section of the panel and the MT Disconnect Switch Group on the left section of the panel. Most of these switches are used for maintenance purposes. If these switches are set to any but their normal position, computer operation must be in the Test mode. A complete discussion of the disconnect switches is not within the scope of this manual.

3-7. OPERATION OF THE COMPUTER.

3-8. GENERAL. - Computer operation is in either the Normal or Test mode. The selection of either of these modes is shown by the illumination of an indicator in the Operating Group. (See figure 3-2.) Automatic selection of the Test mode occurs when the TEST/NORMAL switch is set to TEST, or when the NORMAL/ABNORMAL DRUM switch is set to ABNORMAL, or when one of the buttons in the Operating Rate Group is depressed. (The TEST/NORMAL switch and NORMAL/ABNORMAL DRUM switch are both in the Test Switch Group.) Operation is at high speed unless one of the Operating Rate Group selections have been made.

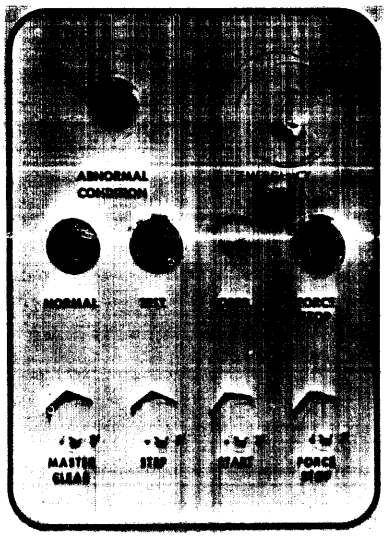


FIGURE 3-2. OPERATING GROUP

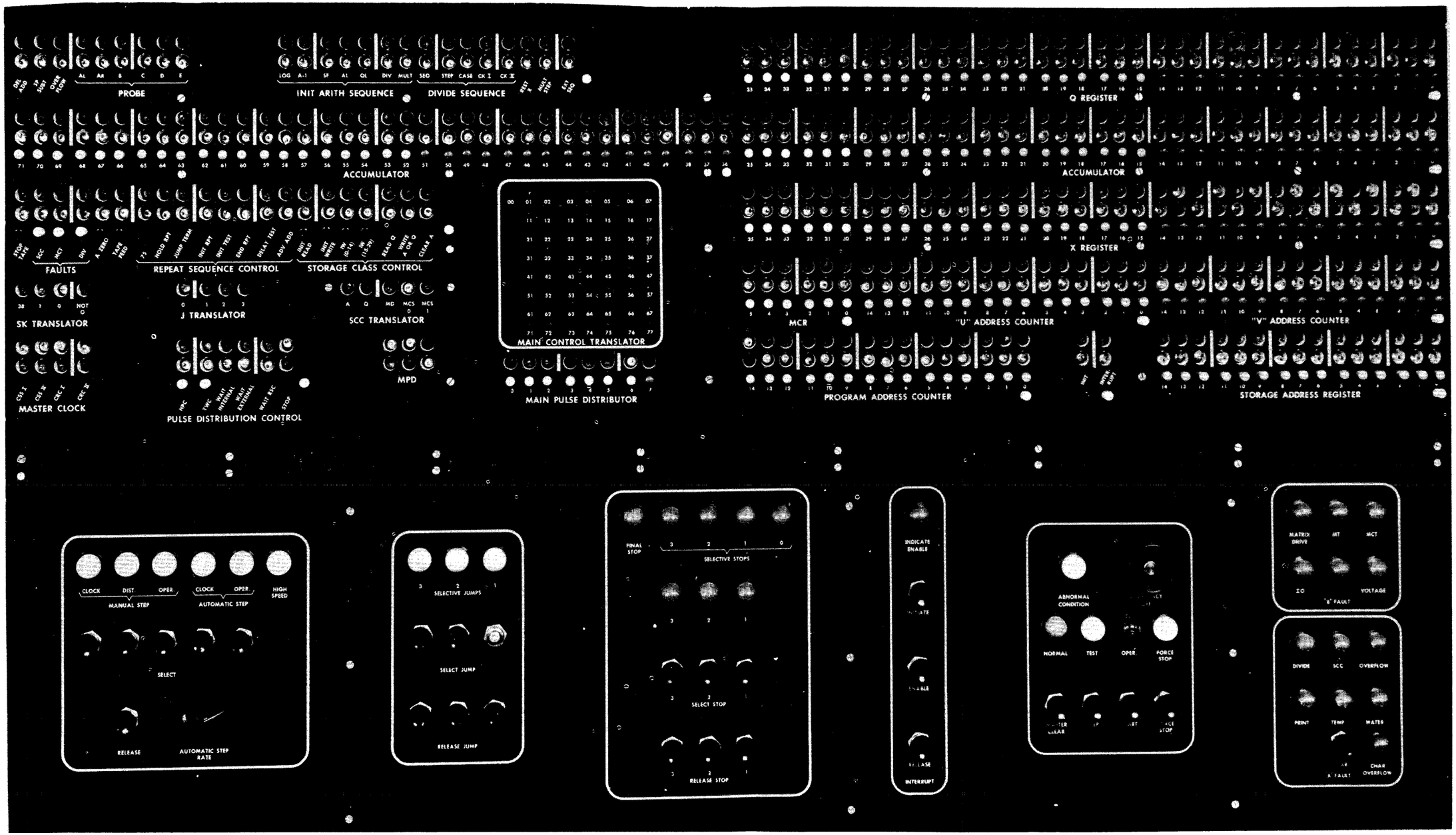
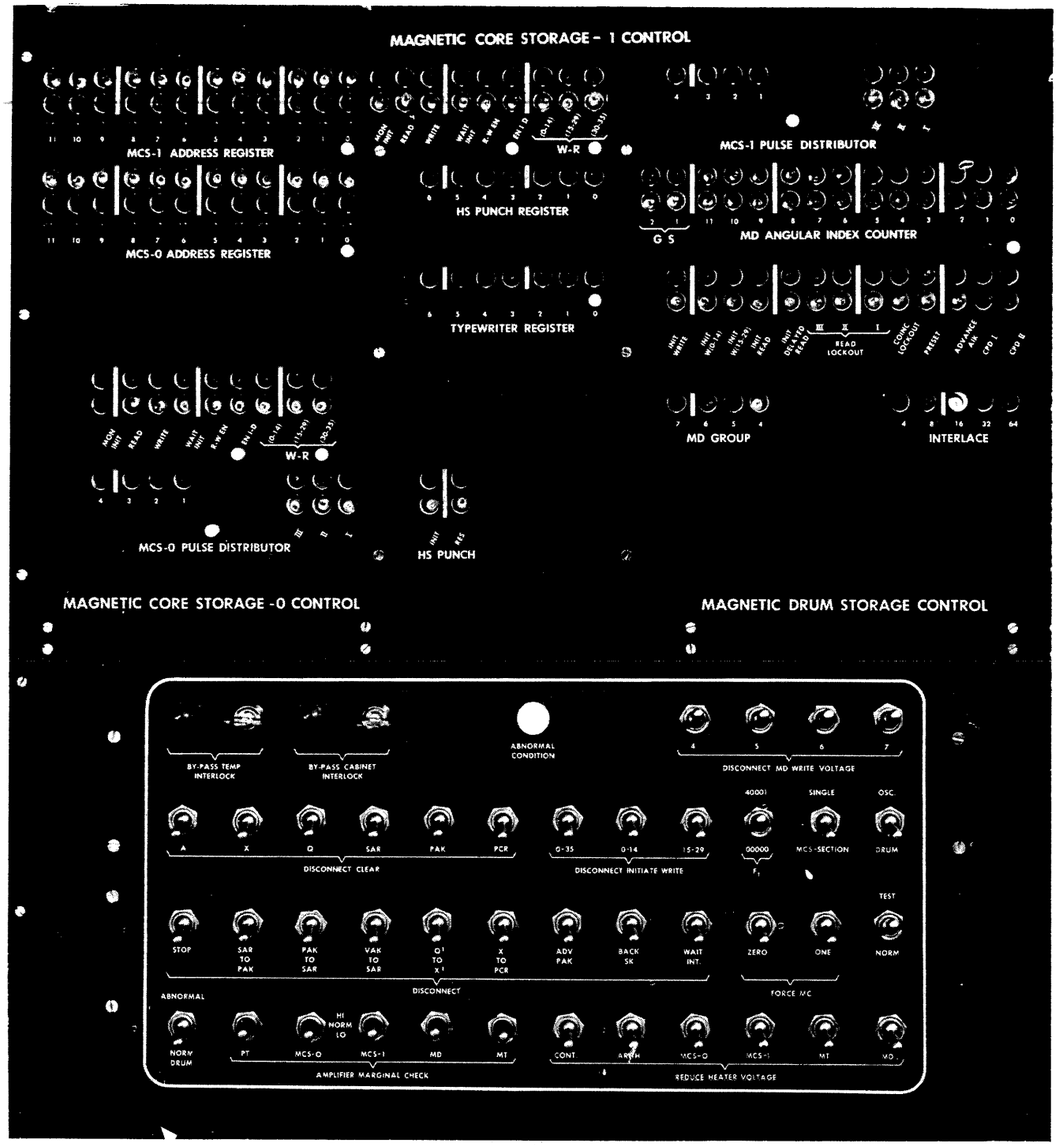


FIGURE 3-1
(part 1 of 2)

SUPERVISORY CONTROL PANEL



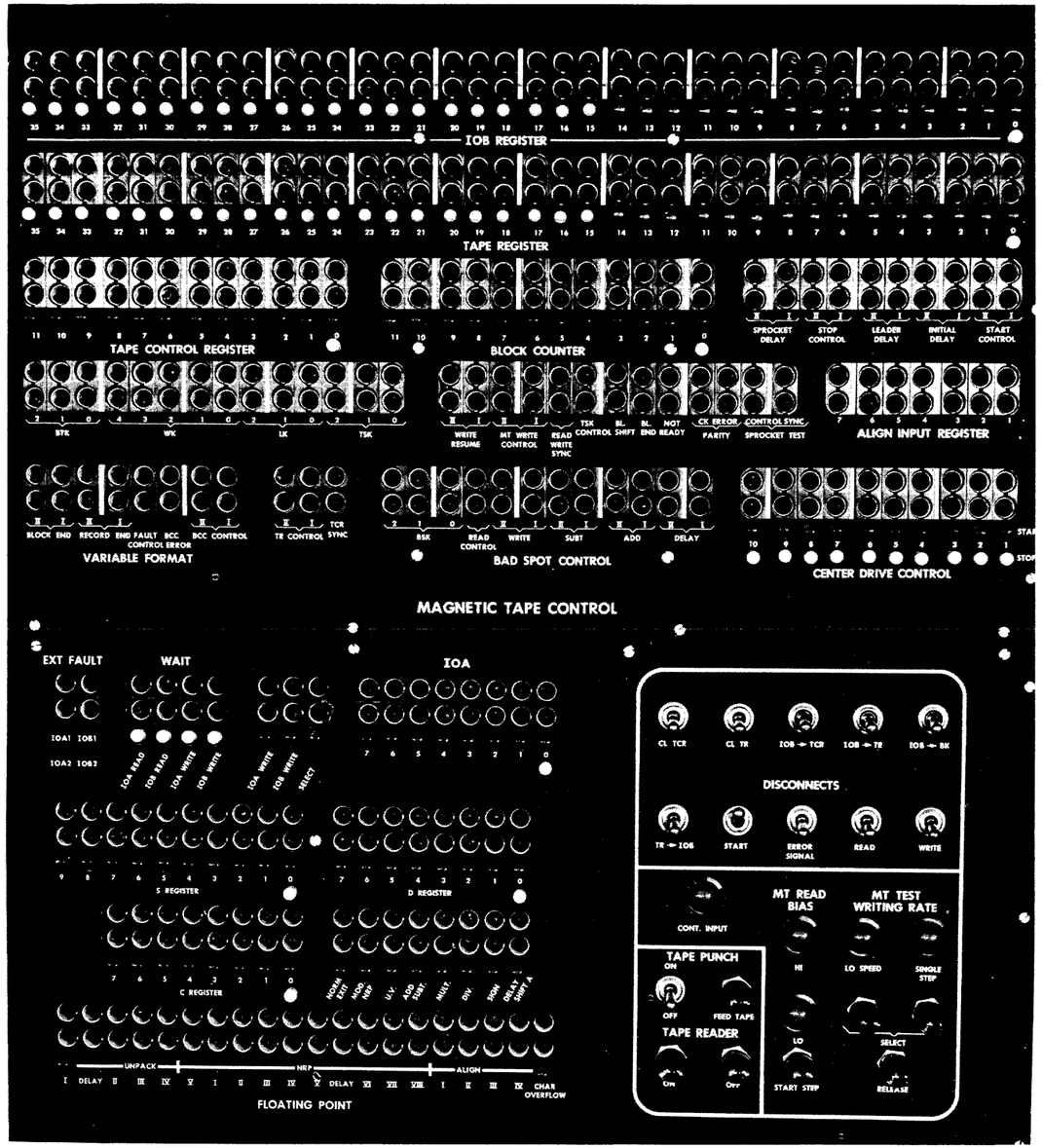


FIGURE 3-1
(part 2 of 2)

SUPERVISORY CONTROL PANEL

3-9. If the ABNORMAL CONDITION indicator in the Operating Group is illuminated, operation cannot be initiated unless the Test mode of operation has been selected and the TEST indicator is illuminated. The ABNORMAL CONDITION indicator is illuminated when any of the disconnect switches in the Test Switch Group or the MT Disconnect Switch Group are positioned to their abnormal condition setting.

3-10. Depressing the START button in the Operating Group illuminates the OPERATING indicator (assuming the condition above has been satisfied). The computer is said to be in operation as long as the OPERATING indicator is illuminated. The OPERATING indicator is extinguished by a computer fault, a programmed stop, or the depression of the FORCE STOP button. A Master Clear is not effected by depressing the button so labelled unless the OPERATING indicator is extinguished at the time. Depressing the MASTER CLEAR button automatically sets the Main Pulse Distributor (MPD) to six and the Program Address Counter, PAK, to 40000. Operation is begun according to the setting of MPD. The setting of MPD can be observed in the center portion of the center section of the control panel. If MPD is set to six, the first instruction is acquired from the address shown in PAK, and this instruction is placed in the Program Control Register, PCR. If the automatic setting of MPD and PAK are not desired, these registers and PCR can be altered manually before depressing the START button. If MPD is set to zero, the instruction currently in PCR is first to be executed when the START button is depressed.

3-11. If one of the Operating Rate Group selections has been made, the STEP button in the Operating Group must also be depressed, after the START button is depressed, before the computer actually initiates any operations.

3-12. NORMAL MODE OF OPERATION. - Only high-speed operation is possible in the Normal mode. In normal mode operation, manual selections can not be made at the control panel after the OPERATING indicator is illuminated. (An exception is the selection of a j for the Manually Selective Stop instruction.)

3-13. The computer is ready for operation in the Normal mode if both the TEST/NORMAL switch and the NORMAL/ABNORMAL DRUM switch are set to NORMAL, and the HIGH SPEED and NORMAL indicators are illuminated. If High Speed and Normal conditions are not indicated at this point, the RELEASE button in the Operating Rate Group must be depressed.

3-14. TEST MODE OF OPERATION. - Operation in the Test mode is at high speed unless one of the Step buttons in the Operating Rate Group is depressed. (See figure 3-3.) Timing during MANUAL STEP operation is controlled by the selection of CLOCK, DISTRIBUTOR, or OPERATION, and the depression of the STEP button in the Operating Group. Each depression of the STEP button releases respectively one CLOCK pulse, one main DISTRIBUTOR pulse, or the sequence of pulses necessary for the operation (execution) of one instruction, i.e., from MP6 to MP6 .

3-15. The AUTOMATIC STEP RATE potentiometer controls the rate at which Automatic Step Operation or Automatic Step Clock operations are performed. If AUTOMATIC STEP OPERATION is selected, the rate at which instructions are executed is controlled. If AUTOMATIC STEP CLOCK is selected, the rate at which clock pulses are released is controlled. Depression of the RELEASE button in the Operating Rate Group releases any selection made and prepares the computer for high speed operation.

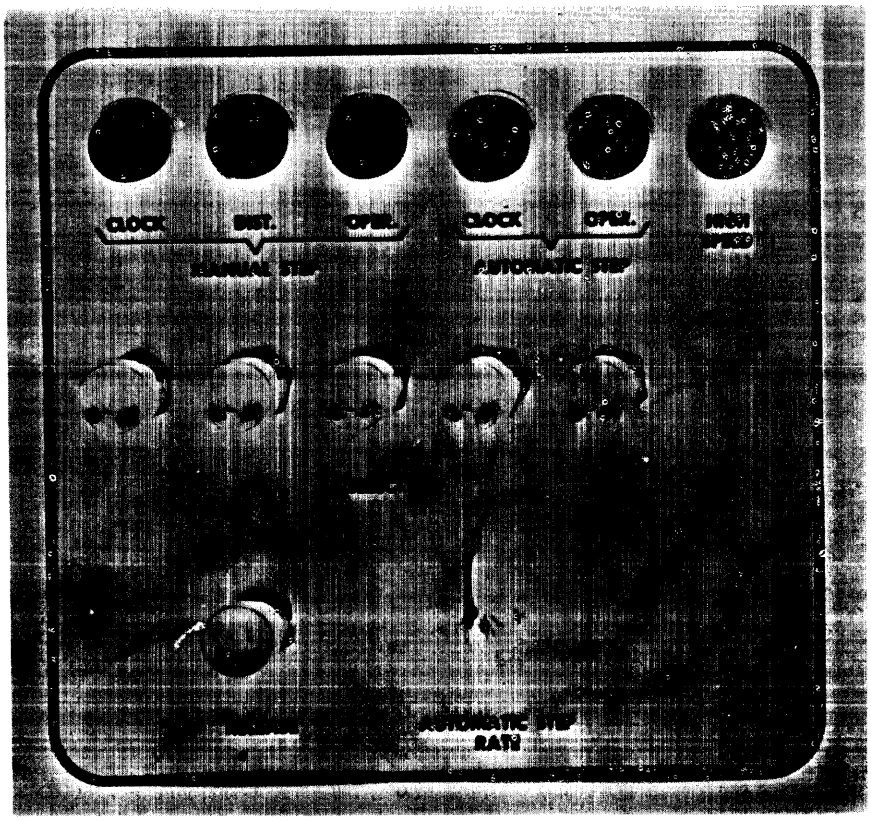


FIGURE 3-3. OPERATING RATE GROUP.

3-16. The computer is ready for operation in the Test mode if the TEST/NORMAL switch is set to TEST, or the NORMAL/ABNORMAL DRUM switch is set to ABNORMAL, or a Step rate in the Operating Rate Group is selected. Any one of these selections illuminates the TEST indicator in the Operating Group.

3-17. If the NORMAL/ABNORMAL DRUM switch is set to ABNORMAL, instructions from the Reserve Space on the drum are executed when operation is started.

3-18. Manual selections can be made from the computer control panel during actual operation in the Test mode. However, three exceptions which should be noted are as follows: when the OPERATING indicator is illuminated, PAK cannot be changed; a RELEASE or SELECT JUMP selection cannot be made; and a RELEASE or SELECT STEP selection cannot be made. The content of the Accumulator, Q Register, X Register, and the Program Control Register can be changed manually in the Test mode while the OPERATING indicator is illuminated.

3-19. STARTING OPERATION. - The manual procedure necessary to set the computer in operation, is as follows. (The mode of operation depends upon the Normal mode or Test mode conditions already established.)

a. Depress the MASTER CLEAR button in the Operating Group.

b. Depress any SELECT JUMP or SELECT STOP buttons called for by the program. The associated indicators will be illuminated. Make any optional selections desired, including any changes in MPD, PAK or PCR.

c. Depress the START button in the Operating Group. This illuminates the OPERATING indicator (unless the NORMAL mode and ABNORMAL CONDITION indicators are both illuminated). High speed operation is begun unless a Step selection was made. If this is the case, the STEP button in the Operating Group is depressed for each operation step.

3-20. Operation is stopped by an A Fault, B Fault, Force Stop, or a programmed stop. Indication of the operation halt is given by the extinguishment of the OPERATING indicator, and the illumination of a light indicating the cause of the halt.

3-21. JUMP AND STOP SELECTIONS. - The manual selections necessary to effect a programmed Manually Selective Jump instruction, MJjv with j = 1, 2, or 3, are made by depressing SELECT JUMP buttons in the Selective Jumps Group. (See figure 3-4.) To be effective, the selections must be made while the computer is not in actual operation (when the OPERATING indicator is extinguished), either Normal or Test mode. An effective manual selection is indicated by the illumination of the SELECTIVE JUMP 1, 2, and/or 3 indicator. To nullify a selection, the appropriate RELEASE JUMP button is depressed. This also must be done when the computer is not in actual operation.

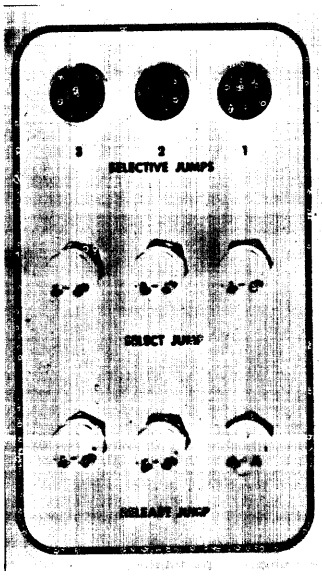


FIGURE 3-4. SELECTIVE JUMPS GROUP.

3-22. The manual selections necessary to effect a programmed Manually Selective Stop, instruction MSjv with j = 1, 2, or 3 are made by depressing SELECT STOP buttons in the Selective Stops Group. (See figure 3-5) A stop selection may be made during actual computer operation, normal or test mode, and is indicated by the illumination of the light immediately above the button depressed. When a stop occurs, it is indicated by the illumination of a SELECTIVE STOP light. (A Manually Selective Stop instruction with j = 0 requires no manual selection). To cancel a stop selection, the appropriate RELEASE STOP button is depressed. The release of a stop selection may also be made during actual computer operation, normal or test mode.

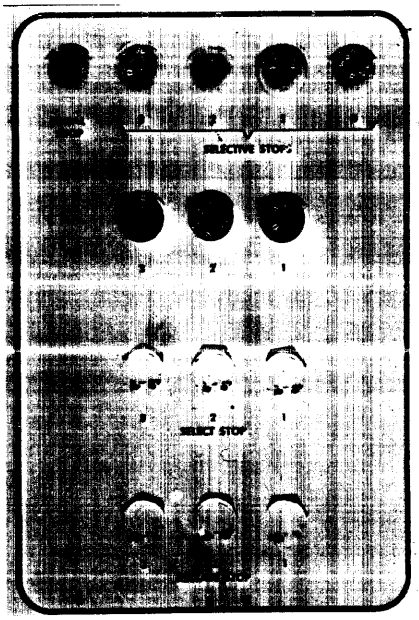


FIGURE 3-5. SELECTIVE STOPS GROUP.

3-23. MANUAL INTERRUPT SELECTION. - An interrupt may be initiated manually at any time by depressing buttons in the Program Interrupt Control Group. (See figure 3-6.) Two selections are necessary to activate a line to the interrupt control of the computer. Depressing the ENABLE button allows the line to be energized and illuminates the INDICATE ENABLE LIGHT. Depressing the INITIATE button when the enable is indicated by the light momentarily energizes the line. When the line is energized, a signal is sent to the Program Interrupt Control to effect the interrupt on MP6 of the first instruction which ordinarily would be concluded by the Normal Termination Sequence. Each time a manual interrupt is desired, the INITIATE button must be depressed and the INDICATE ENABLE indicator must be illuminated. Each time the line to the Program Interrupt Control is energized, the INDICATE ENABLE light is extinguished. Also, depressing the RELEASE button extinguishes the INDICATE ENABLE light. Thus, to effect each manual interrupt, both the ENABLE button and the INITIATE button must be depressed, and in that order.

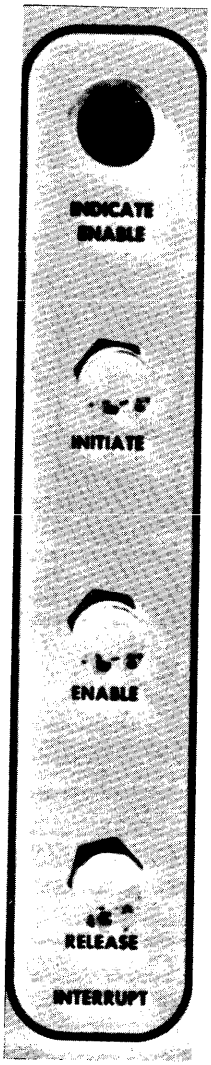


FIGURE 3-6

PROGRAM INTERRUPT CONTROL GROUP

3-24. RESTORATION OF OPERATION AFTER STOPS.

3-25. The computer ceases operation at the occurrence of a programmed STOP, a FORCE STOP, an EMERGENCY OFF, or a fault condition. The stops by classes are discussed subsequently, and the steps necessary to resume operation noted.

3-26. PROGRAMMED STOPS.

a. MANUALLY SELECTIVE STOP. - This stop is caused by a Manually Selective Stop instruction, MSjv, which is described elsewhere. When a stop occurs, the OPERATING indicator is extinguished and the appropriate SELECTIVE STOP indicator (red) is illuminated. To resume operation, depress the START button. The next instruction is taken from the v-address.

b. FINAL STOP. - The Program Stop instruction, PS--, indicates the end of the program. This instruction is described elsewhere. The SELECT STOP indicators and SELECTIVE JUMP indicators, if illuminated, will remain illuminated, and the FINAL STOP indicator (Selective Stops Group) is illuminated. To resume operation, it is necessary to depress the MASTER CLEAR button and follow the procedure for initiating computation in one of the two computer modes.

3-27. FORCE STOP. - An unscheduled immediate stop of computer operation can be effected by depressing the FORCE STOP button (Operating Group). The OPERATING indicator is extinguished and the FORCE STOP indicator (Operating Group) is illuminated. After the condition which prompted the stop has been altered, operation is resumed by pressing the START button. During operation if it is desired to change any selection in the Operating Rate Group or Selective Jumps Group, depressing the FORCE STOP button allows the RELEASE buttons in either of these groups to be depressed and a new selection to be made.

3-28. EMERGENCY STOPS. - An emergency stop removes <u>all</u> voltages from the equipment. Since all power is removed, a program in process is halted and cannot be immediately resumed. Since such a stop could destroy information, it may be well to reload the program and data into the system after proper operation is restored by maintenance procedures. Certain emergency conditions, such as high temperature beyond the allowable operating margin, cause an <u>automatic</u> emergency stop. An emergency stop can also be manually effected, for cases of unforeseen emergencies such as fire, by depressing the EMERGENCY OFF button in the Operating Group.

3-29. FAULT STOPS. - The resumption of operation after a computer fault stop is discussed later in the paragraphs discussing computer faults.

3-30. COMPUTER FAULTS.

3-31. GENERAL. - A computer fault is indicative of an erroneous condition serious enough to warrant the halt of computer operations. A fault condition, and the type of fault, is indicated on the computer control panel.

3-32. Errors leading to fault conditions can be classified broadly as follows:

a. Logical errors made during analysis and programming of the problem.
b. Errors made during the preparation of the coding.
c. Input errors, such as clerical errors during the preparation of the coding for input.
d. Operator intervention errors, such as improper procedures at the computer control panel.
e. Equipment errors - malfunctioning of the computer and the external equipment.

3-33. Computer faults are classified as A Faults and B Faults. A listing of these faults is as follows:

A Faults	B Faults
Divide	Matrix Drive
SCC (Storage Class Control)	MT (Magnetic Tape)
Overflow	MCT (Main Control Translator)
Print	I0 (Input Output)
Temp(erature)	Voltage
Water	
Char(acteristic) Overflow	

3-34. It is often difficult to diagnose the underlying cause of the generation of certain of the computer faults arising from improper programming. Hints which may aid the programmer in isolating the error are given in the subsequent discussion of the individual faults. The area of the program which generated this error is often difficult to locate, and for this task, the programmer must often resort to a "debugging" procedure.

3-35. Debugging is the term applied to the process of locating and correcting errors in a program. This process involves a series of trial runs of the program until the program is known to be without errors. Any computer faults incurred during the trial runs are diagnosed, and the program is corrected to eliminate the cause of the faults. Several types of service routines facilitate the debugging process. The cause of the fault can be eliminated temporarily by manually inserting corrected data into storage from the control panel. The programmer will most likely wish to prepare a new copy of the corrected program. Any corrective manual procedures at the control panel, however, are time-consuming and do not correct the erroneous programming steps which generated the incorrect data leading to the fault. This can be accomplished only by a careful review of the program.

3-36. "A" FAULT. - Computer operation is stopped upon detection of an A Fault. (See figure 3-7, "A" Fault Group.) The OPERATING indicator is extinguished and either one of the indicators in the A Fault Group is illuminated or the ABNORMAL CONDITION indicator in the Operating Group is illuminated.

3-37. An A Fault condition either stops computer operation immediately or allows it to continue momentarily if the continuing operation does not effect the erroneous generation of data. The procedure to resume operation after an A Fault is described for those particular faults where resumption of operation is possible. Resumption of operation is usually not desirable unless the fault was caused by equipment malfunction. The majority of the A Faults indicate an erroneous program.

3-38. OVERFLOW FAULT. - Imminent overflow of the sum into A_{71} is detected during the execution of the Multiply Add instruction. If this possibility exists, an Overflow fault is incurred. It is possible to resume computation after the fault stop by depressing the CLEAR A FAULT button and the START button. If this should be done, the programmer has no further indication as to whether or not the overflow actually occurred. Since this procedure is most likely undesirable, the programmer must try to isolate the reason for the imminent overflow condition. The following statements may aid in this task.

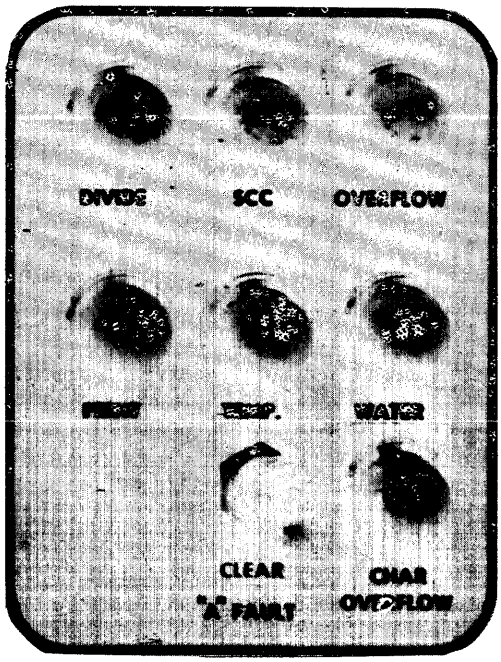


FIGURE 3-7. "A" FAULT GROUP.

3-39. The multiplier and multiplicand can be determined by noting the content of the Q Register and the X Register on the control panel. The magnitude of the number in A can be determined by observing the content of the Accumulator, remembering that the most significant bits of (A) are now in A_R. The addresses of the multiplier and multiplicand are found in UAK and VAK of the Program Control Register. If the Multiply Add instruction is not being repeated, the address of the Multiply Add instruction is found by subtracting one from the address in the Program Address Counter, PAK. If the programmer has some feeling as to what these quantities should be, it may be possible to locate the cause of the fault by making these observations.

3-40. DIVIDE FAULT. - A Divide fault is incurred during the execution of the Divide instruction if conditions are detected which point to the fact that the quotient which should be derived would exceed the capacity of the Q Register. It is possible to resume computation after the fault stop by depressing the CLEAR A FAULT button and then the START button. However, resumption of operation, without taking corrective action, allows the generation of a remainder and quotient which must be regarded as incorrect. The division

is at an undeterminable stage of completion at the time the fault is detected and computer operation is stopped. Thus, noting the content of the Accumulator and the Q Register is of no help to the programmer in his search for the cause of the fault. The divisor is found in the X Register. The address of the divisor is found in UAK of the Program Control Register. If the Divide instruction is not being repeated, its storage address is determined by subtracting one from the address in the Program Address Counter, PAK. These two addresses may enable the programmer to determine if the divisor was obtained from the correct location and to determine if the divisor was of the expected magnitude. However, operation cannot be resumed with the initiation of the Divide instruction since the dividend, which was originally located in the Accumulator before the Divide instruction was attempted, has been altered.

3-41. CHARACTERISTIC (CHAR) OVERFLOW FAULT. - The condition resulting in a Characteristic Overflow fault is detected during the execution of a Floating Point instruction with improper operands. Specifically, the Characteristic Overflow fault is caused by the detection of a derived floating point result with a normalized mantissa and a biased characteristic greater than 255. Computer operation is stopped immediately upon detection of this condition. The normalized mantissa can be noted in A_L, and the out-of-range biased characteristic can be noted in the 10-bit S Register, a special floating point storage register. This register is located on the lower half of the left section of the control panel. The Floating Point instruction being executed is in PCR, and the address of this instruction, unless it is being repeated, can be found by subtracting one from the content of PAK. The Master Clear button must be depressed before operation can be resumed.

3-42. PRINT FAULT. - The Print fault is discussed in the input and output section. Briefly, this fault is caused by an illegal typewriter code being sent to the typewriter by the execution of a Print instruction. This faulty Print instruction is cleared from PCR, and the illegal typewriter code is cleared from TWR, before the illegal code is detected. The computer stop is not immediate. An additional Print instruction if attempted before the fault stop, cannot be completed; and the computer is stalled on this instruction. Noting the content of PAK and PCR is of little or no help to the programmer in diagnosing the cause of the fault, except for the fact that it may enable him to locate the approximate vicinity of the guilty instructions in the program.

The programmer will probably choose to ignore the PRINT fault and resume operation by depressing the CLEAR A FAULT button and the START button. If the same print subroutine is used again, it may be necessary to correct any erroneous programming which led to the fault in order to prevent the re-occurrence of the fault.

3-43. SCC FAULT. - An SCC (Storage Class Control) fault results from a reference to an illegal address. This fault usually indicates an error in programming logic of which this illegal address is but a symptom.

The illegal address is found in SAR, UAK, or VAK. The address of the instruction being executed at the time of the fault is usually found by subtracting one from the content of PAK. This is true unless

a. the instruction is being repeated or

b. an attempt was made to obtain the next instruction from the Accumulator or non-existent storage. If this is the case, PAK contains the illegal address plus one, and MPD is at MP7. No indication is given of the location of the faulty instruction. The Master Clear button must be depressed to resume operation.

3-44. TEMP FAULT. - The temperature fault is an indication of a computer malfunction. A TEMP fault indication is the result of an air temperature above 100° F either in the computer or in some unit of external equipment. The high air temperature is indicated either by the illumination of one of the amber indicators mounted above the computer cabinet doors or by an indicator on the external equipment. Corrective maintenance procedures should be applied immediately. After correction of the high temperature condition, computer operation is re-started by depressing the START button. Operation is resumed from the point at which it was stopped. No data is lost and no incorrect data is generated as the result of a TEMP fault.

3-45. A maintenance decision can be made to resume operation despite the existence of the high temperature condition. The urgency for problem results may dictate that the program be continued despite the over-temperature condition. The maintenance procedure for continuation is to turn the BY-PASS TEMPERATURE INTERLOCK key (Test Switch Group), and depress the START button. A buzzer sounds continually when operating with this condition.

3-46. WATER FAULT. - The Water fault is an indication of a computer malfunction. The Water fault indication results from a water pressure fault condition (over-pressure or under-pressure) in the computer cooling system. Corrective maintenance measures must be applied to eliminate this fault. Operation is resumed after the correction of the Water fault by depressing the START button. Operation is resumed from the point at which it was stopped. No data is lost and no incorrect data is generated as the result of a Water fault.

3-47. ABNORMAL CONDITION FAULT. - An Abnormal Condition fault occurs if any of the disconnect switches in the Test Switch Group or MT Disconnect Switch group are accidentally or intentionally moved to their Abnormal setting position during normal mode operation. If this is done, the ABNORMAL CONDITION indicators in the Test Switch Group and the Operating Group are illuminated. Movement of the switch down to the normal position will eliminate the fault condition. (In the case of the Amplifier Marginal Check switches in the Test Switch group, the Abnormal position is either up or down, and the normal position is its center position.) Operation is resumed, after the switch setting is corrected, by depressing the START button. Operation is resumed from the point at which it was stopped. No data is lost and no incorrect data is generated as the result of this fault.

3-48. "B" FAULT. - Computer operation is stopped by the occurrence of a B Fault. The OPERATING indicator is extinguished and two indicators in the B Fault Group are illuminated - one of which indicates a B Fault in general, and the other of which designates the type of B Fault. Operation cannot be resumed after a B Fault without depressing the MASTER CLEAR button. Note that the MASTER CLEAR destroys the content of such computer registers as A, Q, PAK, PCR, etc. The MASTER CLEAR also extinguishes all of the B Fault indicators, unless the fault is caused by malfunctioning equipment. In this case, the external equipment fault must be corrected before the particular fault light is extinguished and before operation can be resumed. All of the B-Fault indicators must be extinguished before it is possible to resume operation.

3-49. MCT FAULT. - An MCT (Main Control Translator) fault is due to an illegal instruction operation code. (See figure 3-8.) It is detected on main pulse zero, MP0, when the attempt is made to execute the instruction. Computer operation is stopped immediately. The address of the incorrect instruction is determined by subtracting one from the content of PAK. (The address in PAK may indicate that an erroneous jump occurred to an area not containing a program instruction. This fault must then be traced to the instruction which caused the jump.) The incorrect instruction can be noted in PCR; the illegal operation code is found in MCR. Assuming that the correct version of the operation code is known, it is possible to resume operation after the Master Clear by replacing the content of PCR with the correct operation code in MCR, setting MPD to zero, replacing the content of PAK, and depressing the START button. Or, if the instruction is corrected in storage, the address of the instruction can be set in PAK after the Master Clear. Then, depressing the START button resumes operation with this instruction. Since the content of the Accumulator and the Q Register are lost by the Master Clear, this data should be noted previous to the Master Clear and re-inserted in A and Q before resuming operation. Both of these methods of correction should be considered temporary as the occurrence of the fault is indicative of an error in programming logic.

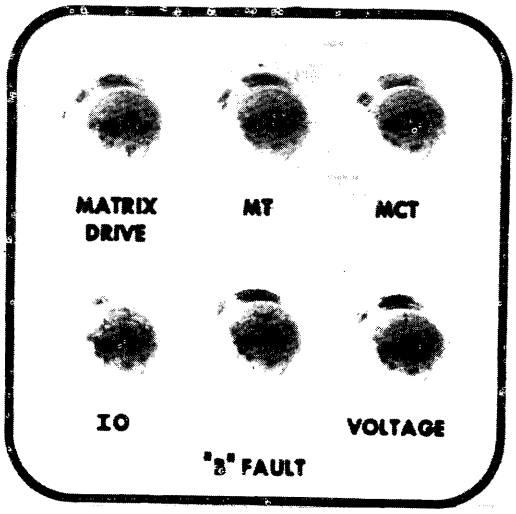


FIGURE 3-8. "B" FAULT GROUP.

3-50. VOLTAGE AND MATRIX DRIVE FAULTS. - The Voltage and Matrix Drive faults are both indicative of a computer malfunction. (The Voltage fault indicates a malfunction in power supply system, and the Matrix Drive fault indicates a malfunction in the Magnetic Core Storage system.) Both of these faults require corrective maintenance. Since erroneous computer operation may result from these faults before the computer is stopped, the program in process at the time should be re-loaded after the correction of the fault and a Master Clear.

3-51. MT FAULT. - A MT (Magnetic Tape) fault indicates a fault generated in using the Magnetic Tape System. The fault could be caused by an equipment error or a programming error. The MT faults are discussed in detail in the input and output section under the discussion of the Magnetic Tape System. The Master Clear button must be depressed to allow resumption of operation.

3-52. IO FAULT. - The IO fault indicates the programming of incorrect input or output procedure or faulty operation of external equipment. (See figure 3-9.) The illumination of the IO fault indicator and one of the EXTERNAL FAULT indicators located on the left section of the control panel indicates a fault involving the use of IOA or IOB These External Faults are discussed in the input and output section. The IO fault, in conjunction with one of the External Faults, generally indicates a program timing error. The nature of an IO Fault <u>not</u> resulting from an External Fault can be determined by checking the fault indicator lights on the external equipment currently in use. Such indicator lights and the associated faults are discussed in the input and output section in the discussions of the particular pieces of external equipment. The fault indicators resulting from programming errors are extinguished by depressing the MASTER CLEAR button. The fault indicators resulting from the improper operation of external equipment are extinguished by correcting the condition causing the fault, and then depressing the MASTER CLEAR button. The occurrence of an IO fault caused by incorrect program timing may necessitate a major revision of the program.

FIGURE 3-9. EXTERNAL FAULT.

3-53. MANUAL READING AND WRITING.

3-54. Two examples of manual operations at the control panel are given to illustrate the type of operations possible. These procedures can be utilized in the diagnosis and correction of certain of the computer faults. However, use of these procedures is not recommended for purposes of debugging because of the amount of time required for manual computer operation.

3-55. MANUAL WRITING FROM THE Q REGISTER. - The procedure below is used to manually insert a word into storage, and then display in the Program Control Register the word which was inserted.

a. Depress MASTER CLEAR (Operating Group)

b. Select MANUAL STEP DISTRIBUTOR (Operating Rate Group)

c. Set Main Pulse Distributor, MPD, to 0

d. Set MCR to 11 (Transmit Positive instruction)

e. Set UAK to 31000 (Q address)

f. Set VAK to the desired storage address

g. Set PAK equal to VAK

h. Place in Q Register word to be written

i. Depress START button (Operating Group)

j. Depress STEP button (Operating Group) only until PCR displays the word just written into storage.

3-56. MANUAL READING TO THE Q REGISTER. - The procedure below is used to extract a word from a chosen storage location and display it in the Program Control Register.

a. Depress MASTER CLEAR (Operating Group)

b. Select MANUAL STEP DISTRIBUTOR (Operating Rate Group)

c. Set PAK to address of word to be extracted

d. Depress START button (Operating Group)

e. Depress STEP button (Operating Group). Word appears in PCR.

To extract the word at the next address, and at successive addresses, continue as follows.

f. Set Main Pulse Distributor, MPD, to 6

g. Depress STEP button

h. Repeat steps 6 and 7.

section 4

INPUT - OUTPUT

4-1. GENERAL.

4-2. The input output section of the Univac Scientific Computing System incorporates three input output sub-systems: one which provides for the use of an electric typewriter; one which provides for the use of a high speed paper tape punch; and one which provides for the use of a variety of general input output equipments. The input output section is the link between external equipments and the computer storage. Data transmissions, as directed by computer instructions, proceed in general as shown in figure 4-1. The transmission of data occurs necessarily in accordance with the operating rate of the external equipment.

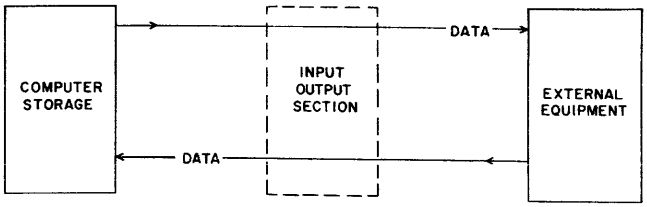


FIGURE 4-1

4-3. The input and output section might be loosely defined as being comprised of those computer elements whose basic function is input and/or output, and the control circuitry associated with the external equipments. In the following discussion, the three sub-systems of the input output section are referred to as the Typewriter sector, the High Speed Punch sector, and the general Input-Output sector.

4-4. The Typewriter sector facilitates the output of on-line typewritten information at a rate of approximately nine characters per second. Print instructions are programmed to direct the operation of the typewriter. The typewriter is particularly effective for monitoring the progress of complex or critical programs. By appropriate use of Print instructions, a programmer can cause the typewriter to print out various symbols (numbers and/or characters) which, in effect, give the programmer significant information about the program being run. (Because of its relatively slow rate of printing, the typewriter is not recommended for use when a large quantity of output is desired.) The typewriter can also provide a punched tape copy (in typewriter code) of all the information it prints.

4-5. The High Speed Punch sector facilitates punched paper tape output at a rate of 60 frames of tape per second. Punch instructions are programmed to direct the operation of the paper tape punch. The High Speed Punch is a convenient means for getting information out of the computer at a medium rate of speed. Output paper tape data, depending upon the code in which it is punched, can be tabulated off-line, thus freeing the computer for further computation, or can be used as a subsequent input for the computer.

4-6. The Input-Output sector is of general-purpose design and the only subsystem which provides for both in and out data transmission. The Univac Scientific can communicate with a wide variety of external equipments, with other computers, and with control systems; in fact with any external unit capable of delivering and/or receiving binary digital information. In this manual, the following external equipments, and their relationship to the Input Output sector, are discussed: the Photoelectric Tape Reader, the Univac Card Equipment and the Univac Magnetic Tape Units. Only the input-output aspects of the Univac Card Equipment and the Univac Magnetic Tape Units are considered. Since these equipments are capable of communicating with the computer as both an input and output device, they can be considered also as auxiliary storage for the computer.

The Input-Output sector is used when any of the following instructions are executed.

a. External Function - selects an external equipment for operation and, in many cases, the mode of operation (as start, stop, free run, etc.) for that equipment. This instruction is necessary for an effective External Read or External Write instruction.

b. External Read - achieves the storage of data, which has been transmitted from previously selected external equipment, in the computer high speed storage.

c. External Write - extracts data from the computer high speed storage for transmission to previously selected external equipment.

4-7. Table - presents a synopsis of the make-up of the input output section.

SECTORS	INPUT OUTPUT INSTRUCTIONS	EXTERNAL EQUIPMENT	EQUIPMENT RATE
Input-Output	External Function External Read External Write	Photoelectric Paper Tape Reader Univac Card Equipment Univac Magnetic Tape Units Any other device capable of delivering and/or receiving binary digital information	200 frames/sec. 120 cards/min. 1800 words/sec. (fixed block length) 2100 words/sec. (var. block length)
Typewriter	Print	Electric Typewriter	9 characters/sec.
High Speed Punch	Punch	High Speed Paper Tape Punch	60 frames/sec.

The input output section is elaborated on in the following pages, and the operation of the external equipment(s) associated with each sector is described in detail. Although the objective of this portion of the manual is to formulate proper programming procedures for in-out data transmissions, an emphasis is also placed on the actual sequence of events that occur when each external equipment is called upon to receive and/or supply data. Also discussed are the various faults that can occur if (1) the safe-programming rules are not observed, or (2) the required manual preparations prior to in-out operations are not made, or (3) the equipment fails. Since certain equipments have off-line functions, manual controls for operating these equipments apart from the computer are discussed where such a discussion is appropriate.

4-8. INPUT OUTPUT SECTOR

4-9. GENERAL. The Input-Output sector links the central computer with various external equipments. The principal central computer components are the Input-Output Registers (IOA and IOB), their associated lockout circuitry, the Interrupt control circuits, and the In-Out Fault detection circuit. The external portion in general consists of the control circuits associated with the external equipments used.

4-10. THE INPUT-OUTPUT REGISTERS. – IOA (8 bits) and IOB (36 bits) are basically buffer storage registers. For input transmissions, IOA (or IOB) receives information from a previously selected external equipment and temporarily retains that data. During the execution of an External Read instruction, the data is taken from IOA (or IOB) and sent to the computer memory, via the X Register. In all output transmissions (except those for the typewriter and high speed punch, IOA (or IOB) receives information from the X Register during the execution of an External Write instruction, and temporarily retains that data until it is accepted by an external equipment. IOB is also used to store a code word during the execution of an External Function instruction. This code word determines which external device associated with the Input-Output sector is to function, as well as the manner in which it is to operate.

4-11. INPUT-OUTPUT LOCKOUT CIRCUITS. - The control of external equipment provided by the execution of appropriate External Function instructions, is merely a basic initial control. External Function instructions are used to start and define how an equipment is to operate, but other circuitry associated with the equipment itself then takes over, interprets what is to be done, and actually controls the equipment while the required operation is carried out. As soon as the External Function instruction is executed, the computer is free for internal operations (computing); but with time restrictions. At specific times in the initiated operation of the external equipment, the program must process data, i.e., the computer must execute External Read or External Write instructions, as required by the in-out operation in progress. When the in-out operation specified by the External Function instruction is completed, the external equipment either stops automatically, as is the case in "step" mode of operation, or is stopped by the execution of another External Function instruction, as is the case in "free run mode" of operation.

4-12. Because the activity of the external equipment is electromechanical, the rates at which the computer and the external equipment operate can be different. Lockout circuits are provided to perform two functions: (1) to prevent the central computer from running ahead of external equipment and (2) to enable the central computer to know when it can function in an in-out operation. Lockout circuitry is thus a brake on computer activity to prevent the computer from executing External Function, Read, or Write instructions too soon.

4-13. If at the time the computer initiates the execution of an External Read instruction, the referenced input-output register has not received data from some external equipment, the computer is stalled by the lockout condition. In transferring input data to the referenced input-output register, the external equipment removes the lockout for that register. The computer then proceeds to complete the External Read instruction, transferring the input data to storage. At the same time, the computer also re-establishes the lockout condition for the referenced input-output register.

4-14. The lockout circuit has a corresponding function during output operations. When the computer executes either an External Function or External Write instruction, the action is that of an output operation. The execution of these instructions sends data to the referenced input-output register. (The External Function instruction references the IOB register. The External Write instruction references either IOB or IOA.) To insure that the register referenced is cleared and not involved in any previous output operation, the computer tests the output portion of the lockout circuitry when it begins the execution of an External Function or External Write instruction. If the referenced register is involved in an output operation, computer operation is stalled until the completion of the previous operation. (The completion of the previous operation implies that the external equipment has accepted, for its use, the contents of the referenced register.) At that time, the input-output register is cleared, and the lockout condition for that register is removed by the external equipment. The computer then proceeds with the execution of the External Function or External Write instruction. With the transmission of data to the referenced register, the computer re-establishes the lockout condition.

4-15. There is no provision made to lock out the computer in the following situation: IOA or IOB is sent <u>input</u> data by an external equipment. This data is not disposed of by an External Read instruction before the execution of an External Write instruction (in the case of IOB). The logical sum of the input data and the output data is formed in the register involved. External Read instructions must therefore be properly programmed to eliminate the possibility of this error for which no detection is provided.

4-16. INTERRUPT FEATURE. - The interrupt feature can be used in relationship with external equipment as follows:

a. An interrupt signal can be generated within the external equipment, and sent to the computer (where such a provision is made in the external equipment), at the times the external equipment is ready to process data. The

option of sending such an interrupt signal to the computer may be provided by either a manual switch selection on the external equipment or by appropriately coding an External Function instruction with an Interrupt bit.

b. An interrupt signal can be manually initiated at the computer control panel at such times when equipment, not on-line with the computer, has data ready for computer processing.

Thus, the interrupt feature allows the programmer to use the computation time available between successive data transmissions more freely and safely.

4-17. Both of the above uses of the interrupt feature imply that (1) an appropriate data processing routine is stored in the computer, (2) this routine is referenced by an instruction at F_3, or by an instruction following the execution of (F_3), and (3) the address in PAK, at the time of interruption, is saved if a return to the program in process at the time of the interruption is desired. A Return Jump instruction stored at F_3 can accomplish both (2) and (3). A return to the program in process at the time of interruption suggests that certain storage locations (such as A and Q), used by both the interrupted program and the data processing routine, should have their contents prestored before, and restored after, the execution of the data processing routine. Hence, the data processing routine is usually not initiated immediately after the interrupt signal is effective. With most external equipments only a limited amount of time is allowable, after the equipment is ready to process data, before the data must be processed. This time is known as the "receptive time" of the equipment. Failure to process data within the times specified for each equipment causes the loss of data and usually a computer fault. Therefore, the programmer must heed the receptive time for the equipment receiving or transmitting data during preparation of any routine prefacing the actual data processing routine. Computation time in excess of the receptive time might also occur if the program in process at the time of the interrupt is currently executing a Repeat instruction, or is making a Magnetic Drum reference.

4-18. IN-OUT FAULT DETECTION CIRCUITS. Circuits are provided to detect both programming errors and external equipment failures. When a fault is detected, the computer stops, and indication of the fault is given on the computer control panel. The type of fault which occurred is indicated either on the computer control panel or on an external equipment.

4-19. The occurrence of any of the IO faults listed below indicates that a logical sum of data has been formed in one of the Input-Output Registers. These faults can be caused by programming errors or by equipment failures. If any of these faults occur, the specific fault is indicated on the computer control panel on the left section, and the IO Fault is indicated on the center section.

a. IOB (IOA) READ FAULT (CLASS I) - This occurs if the computer did not dispose of previous input data sent to IOB (IOA) before the external equipment sent more input data to IOB (IOA). A programming error resulting in this fault could be the failure to program a sufficient number of External

Read instructions to process the input data, or the failure to program these External Read instructions sufficiently close together, timewise, so as to "keep up" with the external equipment.

b. IOB (IOA) READ FAULT (CLASS II) - This occurs if information sent to IOB (IOA) by an External Write instruction or External Function instruction if IOB is involved was not disposed of by an external equipment before an external equipment sent input to IOB (IOA). This fault can result from executing an External Function instruction during, (rather than before and after) data processing periods of an in-out operation.

4-20. EXTERNAL INSTRUCTIONS. - The execution of an External Function instruction, EF-v, transmits a 36-bit code word from the v address of EF-v to IOB. The purpose of this code word is to select which of the external equipments is to function and to specify the manner in which that equipment will operate.

4-21. The particular external equipment selected is determined by the placement of a "1" in a specific master bit position of IOB. A "1" in a particular master bit position is known as the "IOB select bit" for the specified equipment. IOB select bits for the equipments discussed in this manual are as follows:

IOB_{35} - Univac Card Equipment

IOB_{33} - Photoelectric Paper Tape Reader

IOB_{31} - Univac Magnetic Tape Units

The function of the selected external equipment is specified by the combination of "1's" placed in other bit positions of IOB. A list of IOB select bits and code words for the above-mentioned equipments is given in a table in the appendix.

4-22. The v-address for an External Function instruction starting an external equipment from rest can be any addressable computer location. In all other uses of EF-v, an MD address should not be referenced because of the random access time to the drum, and the restrictions on timing required by the external equipments.

4-23. The purpose of the External Read instruction, ERjv, is to store input data, from some external equipment, in the computer high speed memory. If j = 0, the eight bits in IOA are transmitted to the v-addressed location. If j = 1, the 36 bits in IOB are transmitted to the v-addressed location. Either IOA or IOB, not both, can be involved in the execution of each External Read instruction.

4-24. The v address of ERjv can specify Magnetic Core, the Q Register, or the Accumulator, but not a Magnetic Drum address. Provision is made for a computer fault to occur if v is an MD address. If a fault occurs, it does so after the input data has been stored in MD. The computer fault causes a halt

of computer operation. As a result, other faults occur which stop external equipment and in-out operations. If Magnetic Core or the Q Register are referenced by ERjv and IOA is involved, the rightmost eight bits of (v) are (IOA) and the 28 leftmost bits are zeros. If A is referenced, the eight rightmost bits of the Accumulator are (IOA) and the remaining 64 bits are zeros.

4-25. The purpose of the External Write instruction, EWjv, is to supply data from the computer high speed memory to the external equipment. The right-hand eight bits of (v) are transmitted to IOA if j = 0. The 36 bits of (v) are transmitted to IOB if j = 1. Notification is also given to the previously selected external equipment that data has been delivered to IOA or IOB. Either IOA or IOB, not both, can be involved in the execution of each External Write instruction.

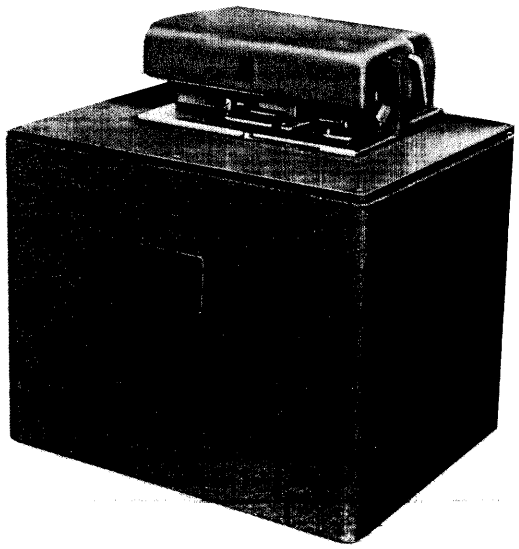
The v-address of EWjv can specify Magnetic Core, the Q Register, or the Accumulator, but not a Magnetic Drum address. If v is an MD address, a computer fault occurs and the computer stops. As a result, other faults occur which stop the external equipment and the in-out operations.

4-26. In the execution of an External Write instruction, the contents of the v address is first sent to the X Register. If j = 0, only the eight rightmost bits of X are sent to IOA; if j = 1 the 36 bits of (X) are transmitted to IOB.

4-27. PHOTOELECTRIC PAPER TAPE READER.

4-28. GENERAL. The execution of External Function instructions control the passage of punched paper tape through the tape reader (see figure 4-2). Input data from the tape is stored by the execution of External Read (j = 0) instructions. The sensing mechanism of the tape reader is binary in nature (a hole in the tape is regarded as a "1"; the absence of a hole, a "0"). Input data can be punched on the tape in any binary code. The loading routine, which controls the punched tape input operations, includes instruction to interpret input data after it is stored.

Figure 4-2.



4-29. PUNCHED PAPER TAPE. Figure 4-3 shows a diagram of a portion of seven level punched paper tape. This diagram illustrates the following terms used in discussing punched paper tape: frame, level, and feed holes.

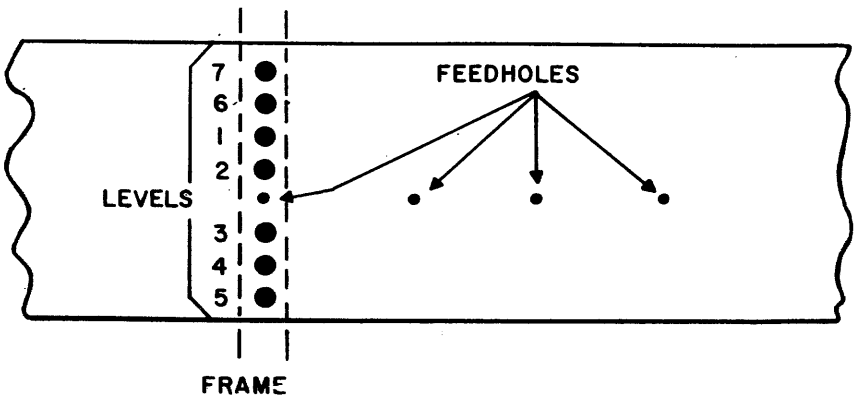


FIGURE 4-3

Levels are positions across the width of the tape in which data holes can be punched. A frame is a column of levels whose position along the length of the tape is defined by the feed hole.

4-30. As the tape moves under the reading head of the tape reader, punched tape is read one frame at a time. Each level contributes its bit of input data, and the feed-holes are used, among other things, to effect simultaneous transmission of the contents of the frame to IOA. In addition to the seven level tape shown in figure 4-3, five level tape may also be employed. Such tape is narrower and does not have tape levels 6 and 7. No discussion is made of 5-level tape in this manual.

4-31. The tape can be punched in typewriter code, bioctal code (two octal digits per frame), binary coded decimal, or any arbitrary code. In general, tape levels 6, 1, 2, 3, 4, and 5 store the actual input data; the tape level 7, if used, stores special loading codes interpreted by the loading routine.

4-32. PHOTOELECTRIC TAPE READER. The Photoelectric Tape Reader is capable of reading paper tape at the rate of 200 frames per second. The principal parts of the tape reader are the tape drive mechanism and the photoelectric reading station. The reading station contains eight photocells which are associated with the seven tape levels and the feed hole in each frame. As the tape moves past the reading station, data is read frame by frame, and a binary image of the data in each frame is generated. When the feed hole for each frame is read, the data from that frame is transmitted to the seven lower order bit positions of IOA. (Actually only the holes are sensed and "1's" transmitted. IOA therefore contains zeros prior to the receipt of each frame of data.) The computer is notified each time IOA receives a frame of data from the reading station. The computer can then execute an External Read instruction (j = 0) to store (IOA) in the computer high speed memory. The transmission from the tape to computer storage is shown in figure 4-4.

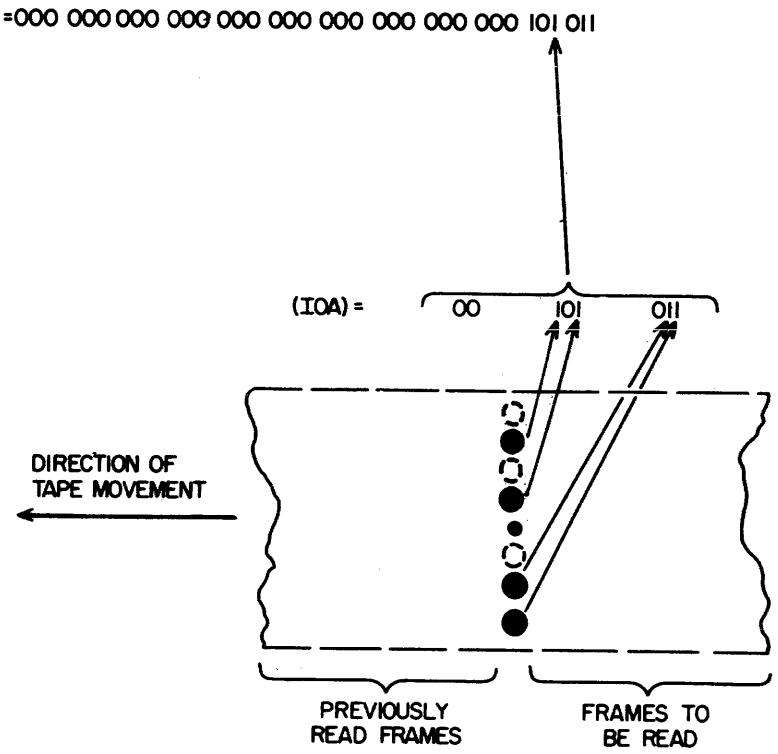


Figure 4-4.

4-33. The tape drive mechanism is an electromagnetic clutch-brake device whose operation is controlled by the execution of appropriate External Function instructions. These instructions specify the type of tape movement through the reader.

4-34. TAPE READER MODES OF OPERATION. The tape reader has two modes of operation, STEP and FREE RUN. In the STEP mode of operation, the tape drive mechanism moves the tape one frame and stops automatically. In the FREE RUN mode of operation, the tape drive mechanism moves the tape continuously through the tape reader.

NOTE

In the following discussion for each mode of operation, when tape motion is indicated, it is assumed that an input tape has been properly positioned in the tape reader, that power has been supplied, that the START button on the tape reader has been depressed, and that the tape reader is starting from an initial "at rest" condition.

4-35. The STEP mode of operation is initiated by the execution of an EF-v instruction, where (v) is 100000300000 (octal notation). When the tape reader control circuitry detects that $IOB_{33} = 1$, $IOB_{16} = 1$, and $IOB_{15} = 1$, the following events occur:

a. In the tape drive mechanism, the brake is released and the clutch engaged, initiating tape movement.

b. As a frame of tape passes under the reading station, the feed hole in that frame is sensed to simultaneously effect the following:

   (1). Clear IOB and remove the IOB lockout for the computer.

   (2). Engage the brake and disengage the clutch; i.e., begin stopping tape drive.

   (3). Initiate a delay to make it impossible to subsequently start the reader for approximately three milliseconds.

   (4). Transmit the data read from the frame to IOA and notify the computer that this data can now be processed by an External Read instruction (j = 0).

The reader thus starts, transmits one frame to IOA, and stops. Because the start-stop mechanism is mechanical, the delay of approximately three milliseconds is introduced so that if another External Function instruction is executed, it will have no effect in the reader until the delay interval passes.

4-36. The FREE RUN mode of operation is initiated by the execution of an EF-v instruction, where (v) = 100000200000 (octal). When the tape reader control circuitry detects that IOB_{33}= 1 and IOB_{16}= 1, the following events occur:

a. In the tape drive mechanism, the brake is released and the clutch engaged, initiating tape movement.

b. As the first frame of tape passes under the reading station, the feed hole in that frame is sensed to simultaneously effect the following:

   (1). Clear IOB and remove the IOB lockout for the computer.

   (2). Transmit the data read from the first frame to IOA and notify the computer that this data can now be processed by an External Read instruction (j = 0).

NOTE

The condition of the brake and clutch are not disturbed; tape movement is continuous.

4-37. IOB is cleared as the first frame of data is sensed. As each succeeding frame passes under the reading station, the feed hole in that frame is used to:

a. Transmit the data read from that frame to IOA and notify the computer that this data can now be processed by an External Read instruction (j = 0).

b. Determine, since the last feed hole was sensed, if an External Function instruction was executed with (v) = 100000100000. If this instruction was executed since the data of the last frame was sent to IOA, the current feed hole is then used to transmit the data of the current frame to IOA, to stop tape drive, to initiate a delay which makes it impossible to subsequently start the tape reader for three milliseconds, to clear IOB, and to remove the lockout condition set up for IOB during execution of the (STOP FREE RUN) External Function instruction.

4-38. In the FREE RUN mode of operation then, tape movement is begun and is continuous until the computer program stops it. At any time that the tape drive is stopped in this mode, the data from the last frame to pass under the reading station has been sent to IOA and must be disposed of by an External Read instruction (j = 0). It is also important to execute the (STOP FREE RUN) External Function instruction in the interval between the times at which the last two desired frames are sent to IOA.

4-39. PROGRAMMING AND TIMING. Punched tape input operations require the execution of a sequence of instructions called a loading routine. Each routine provides the following.

a. Appropriately coded External Function instructions which are timed for execution to effect the desired passage of tape through the tape reader.

b. One External Read instruction (j =0) for each frame of input data, with each ERjv programmed timewise for execution as determined by the timing restrictions of the reader.

c. Instructions which provide for any necessary assembly and interpretation of input data. This is necessary because of the particular coding of the input data.

d. Instructions which differentiate authentic information in each frame of tape from meaningless input.

4-40. Safe programming procedures for the use of External Function and External Read instructions are best illustrated by timing diagrams for the two modes of operation (see figures 4-5 and 4-6).

4-41. TIMING. In the STEP mode, at least nine milliseconds of computation time are available between successive starts of the tape drive mechanism (see figure 4-5). Assume the tape reader is ready for operation when the EF-v, (v) = 100000300000 is executed. If the tape reader is experiencing the three millisecond stopping delay from a previous operation, up to three milliseconds is introduced between ① and ②, shown in figure 4-5 to occur simultaneously.

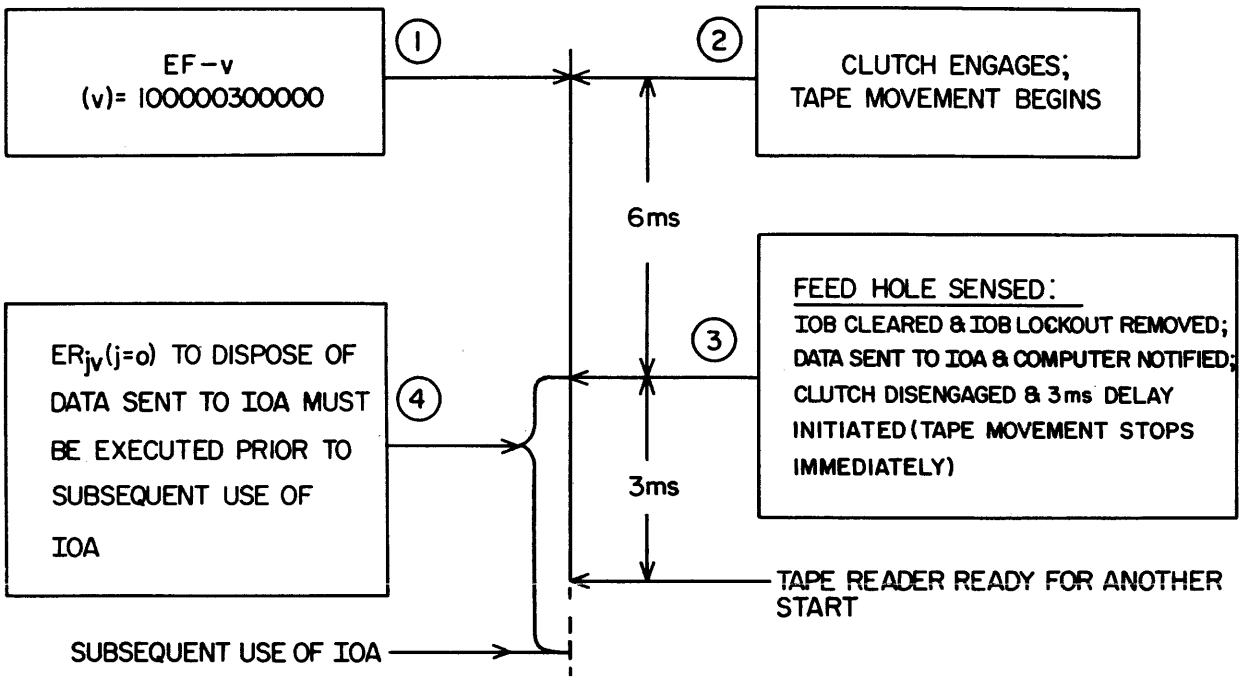


Figure 4-5

4-42. Since the tape reader stops automatically in this mode, the computer is free for any amount of computation between successive EF-v, (v) = 100000300000 instructions. However, to dispose of the data sent to IOA, an External Read instruction (j = 0) must be executed sometime after each EF-v (STEP) instruction and prior to any other EF-v which initiates input to IOA. Failure to program in this manner results in an IOA Read Fault, (Class I).

4-43. Timing for the FREE RUN mode is shown in figure 4-6. In this diagram, six frames are read and transmitted to IOA. Five External Reads are shown, and the execution of the sixth ER is mentioned. This example can, of course, be expanded to include the reading of any number of frames. It is assumed that the tape reader is ready for operation at the time the EF-v, (v) = 100000200000 is executed. If the tape reader is experiencing the three milliseconds stopping delay from a previous operation, up to three milliseconds is introduced between points (1) and (2), shown in figure 4-6 to occur simultaneously.

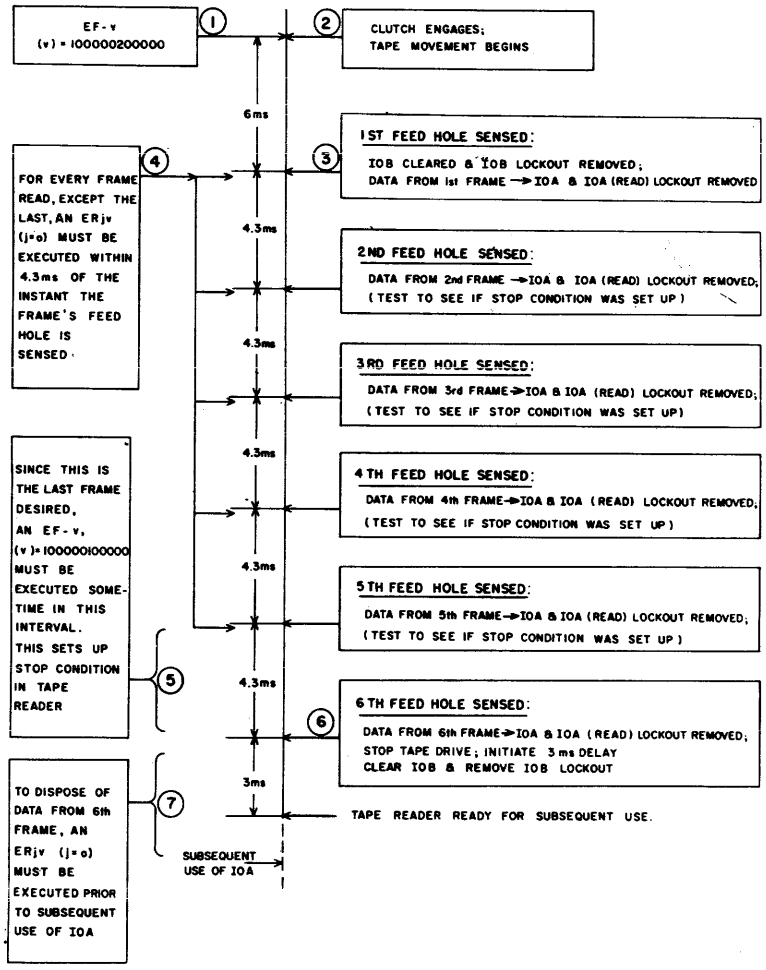


Figure 4-6

Figure 4-6 illustrates each of the following.

a. At least six milliseconds, but less than 10.3 milliseconds, computing time is available between EF-v, (v) = 100000200000, and ERjv (j = 0) required to read the first frame.

b. Data from each succeeding frame is safely disposed of if External Read instructions (j = 0) are programmed at intervals less than 4.3 milliseconds.

c. To read exactly n frames, an EF-v, (v) = 100000100000, must be programmed for execution between the time the $(n - 1)^{th}$ and the n^{th} feed hole is sensed. However, the n^{th} frame has been sent to IOA and must be disposed of by an External Read instruction (j = 0) before any subsequent use is made of IOA. If an ERjv (j = 0) is not programmed to dispose of the last frame sent to IOA, and a subsequent input is sent to IOA from any external equipment, an IOA READ FAULT (Class I) occurs. Also, the possibility exists that an External Write instruction (j = 0) might be attempted before the External Read (j = 0) is executed. If this should occur, the logical sum of data is formed in IOA. This error is not detected by the computer.

4-44. SAMPLE SEVENTH LEVEL LOADING CODES. Seventh level holes may be used to direct the storage of input data and check the loading operations. A loading routine which interprets the seventh level holes for this purpose is commonly known as a bioctal loading routine. The arrangement of seventh level holes is arbitrary. Figure 4-7 is cited as only one example.

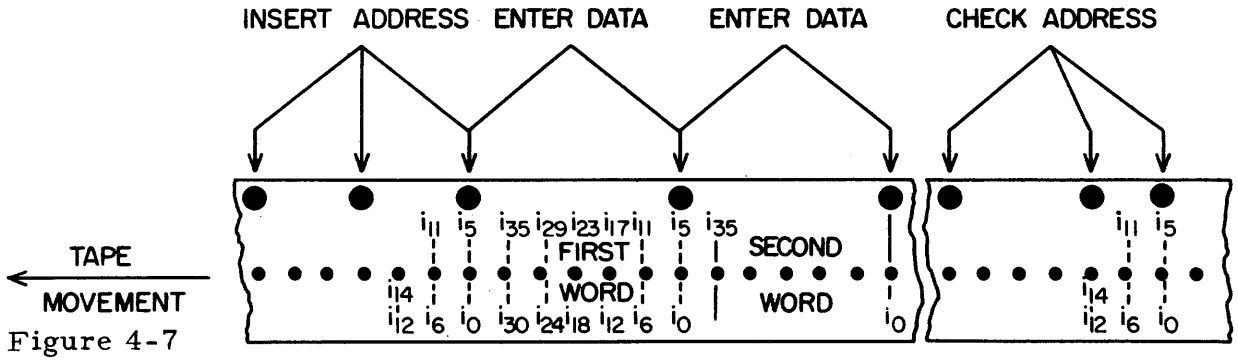


Figure 4-7

4-45. The INSERT ADDRESS (figure 4-7) indicates 15 bits to be used as a "dummy" Program Address Counter to effect the desired storage of the words on the tape. The ENTER DATA indicates that the word just assembled is to be placed at the address held in the "dummy" Program Address Counter. (The "dummy" PAK should be advanced after each "enter data" operation.) The CHECK ADDRESS indicates 15 bits which should check with the address held in the "dummy" Program Address Counter.

4-46. The loading routine for this tape should accomplish the following. As each frame of tape is read,

a. isolate the data from levels 6, 1, 2, 3, 4, and 5 from the 7th level bit.

b. assemble the data of six adjacent frames.

c. combine the 7th level bits to form INSERT ADDRESS, ENTER DATA, or CHECK ADDRESS configurations.

As each sixth frame of tape is read,

a. determine the configuration formed by the combination of the seventh level bits.

b. effect the function indicated by the configuration.

4-47. FAULT DETECTION. The IOA Read Fault (Class I) caused by improper use of the tape reader, as discussed previously, is detected by the computer. If power to the tape reader fails, the motor stops and any tape movement ceases. This situation results in a voltage fault which is indicated on the computer control panel. A voltage fault is a computer B Fault which stops computer operation.

4-48. MANUAL PREPARATION. Manual operations involved in using the tape reader are:

a. Release the clamp over the tape reader passage by turning the lever on the end of the reader hood. Insert the paper tape in the tape reader passage, with the 7^{th} level to the outside (open side). The tape should move freely within the tape guides. Lock the clamp over the tape reader passage by lowering the lever on the end of the hood.

CAUTION
POWER MUST BE OFF WHEN POSITIONING TAPES IN READER.

b. Turn the reader power ON by depressing either the START button on the tape reader or the TAPE READER ON button on the computer control panel. The tape reader cannot be started if an A Fault, B Fault, or PROGRAM (FINAL) STOP condition exists in the computer. If any of these conditions exist, as indicated on the computer control panel, the tape reader may be started after depressing the Master Clear button.

c. The power to the reader is turned OFF by depressing either the STOP button on the tape reader or the TAPE READER OFF button on the computer control panel. The reader should be OFF when it is not in use. When the reader is ON, a transmission to IOA occurs when the reader is turned OFF, and when tape or any other object is moved past the photocell which senses the feed holes in the tape. Also, the computer is notified of this transmission. Depressing the Master Clear button on the computer control panel will return IOA and the read control circuitry to their cleared states. If IOA is in use by other external equipment at the time the reader light is lighted, the reader must be left on until IOA is not in use.

4-49. HIGH SPEED PUNCH

4-50. GENERAL. (See figure 4-8.) The High Speed Punch produces output information on seven level punched paper tape as directed by the execution of appropriate Punch instructions, PUjv. The principal computer components of the High Speed Punch sub-system are the High Speed Punch Register, and the associated lockout circuitry. The external portion of this sub-system is the High Speed Punch, and the associated control ciruitry. A brief discussion is given below for these principal parts.

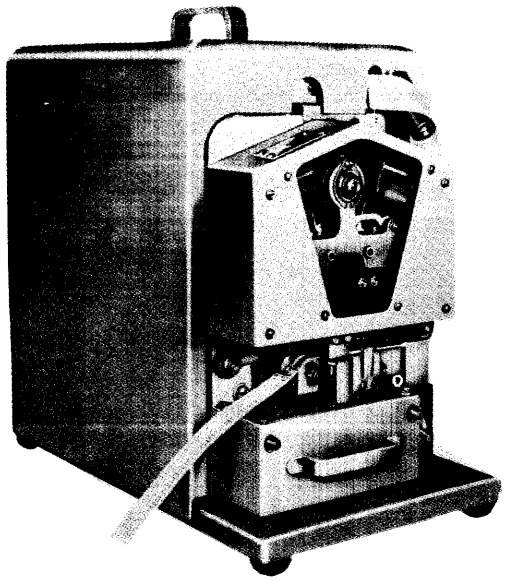


Figure 4-8

4-51. HIGH SPEED PUNCH. The High Speed Punch operates at a rate of 60 frames punched per second. The majority of moving parts are kept continuously in motion when power is supplied to the punch. The punch operates, therefore, in continuous cycles. The punch has a set of eight punch pins, seven of which punch data holes in a frame of tape, and one of which punches the feed hole to identify the frame.

4-52. During each punch cycle, two physical events may occur: (1) tape punching and (2) tape movement. Once during each punch cycle the control circuitry references the High Speed Punch Register for data. If no information has been sent to this register, the punch merely completes its cycle, and no punching or tape movement occurs. If data has been sent to the High Speed Punch Register by the execution of a Punch instruction, preparation for punching is initiated. After punching has occurred, the tape is advanced one frame.

4-53. Tape may be advanced through the punch apart from program control by depressing the TAPE FEED button on the computer control panel. (There is also a lever on the top of the punch which can be operated for this purpose.)

4-54. HIGH SPEED PUNCH REGISTER. The High Speed Punch Register, HPR, is a seven bit register. During the execution of a Punch instruction, "1's" from the six lower order bit positions of the v-addressed location are sent to this register. Bits of "1" from $v_5 \ldots v_0$ are transmitted to $HPR_5 \ldots HPR_0$ via the X Register. Also, a j of 1 in the instruction PUjv is transmitted to HPR_6. The High Speed Punch Register temporarily retains the information until it is needed in a punch cycle. When, during the punch cycle, preparation for punching has been initiated, the High Speed Punch Register is cleared by the punch control circuitry. This provision for clearing HPR is necessary since only "1's" are transmitted to the register. The combination of "1's" and "0's" in HPR determines where holes are punched in the tape, i.e., a hole is punched for each "1" in HPR.

4-55. Each Punch instruction provides for one frame of tape to be punched. Data transmitted to HPR by the Punch instruction PUlv (j = 1) is represented on the tape as shown in figure 4-9.

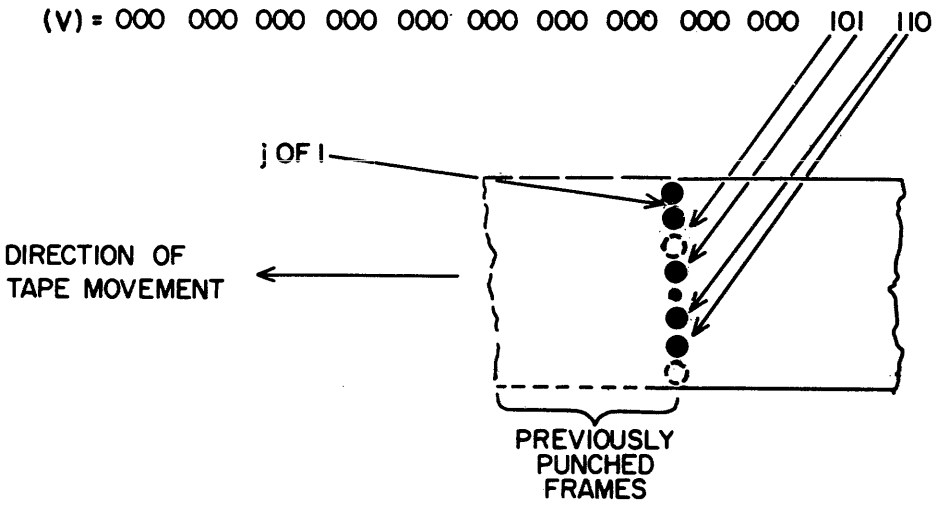


Figure 4-9

4-56. The HPR lockout circuitry functions essentially the same as each of the other output lockout circuits. During the execution of a Punch instruction, but before the transmission of data to HPR, the computer determines whether the punching has been completed for a previously initiated punch cycle. If it has not, an HPR lockout exists for the computer and the computer stalls. When punching is completed for the punch cycle then in progress, the lockout is removed and the Punch instruction is completed. The HPR lockout condition is re-established with the transmission of data to HPR. The computer is free for other computation at this point, but it cannot send any further information to HPR until the current data in HPR is punched and the HPR lockout is removed.

4-57. PROGRAMMING AND TIMING. The Punch instruction directs the transmission of the 36-bit word at the v-addressed location to the X Register. If there is no HPR lockout condition, the "1's" from the six lower order bit positions of X are transmitted to HPR, as is a j of 1. When the punch mechanism is ready to begin a punch cycle, the High Speed Punch Register is checked for data. If HPR has received data, the punch pins are set to punch the contents of HPR, a condition is established to provide for later tape movement, and HPR is cleared. Punching now occurs, after which the HPR lockout is removed from the computer, and the tape is advanced one frame. Any time after the HPR lockout is removed, another Punch instruction may be executed, and more data can be sent to HPR. However, this data remains ineffective in HPR until the punch control circuitry signals that the current cycle is completed and another cycle is to begin. This sequence of events is illustrated in figure 4-10.

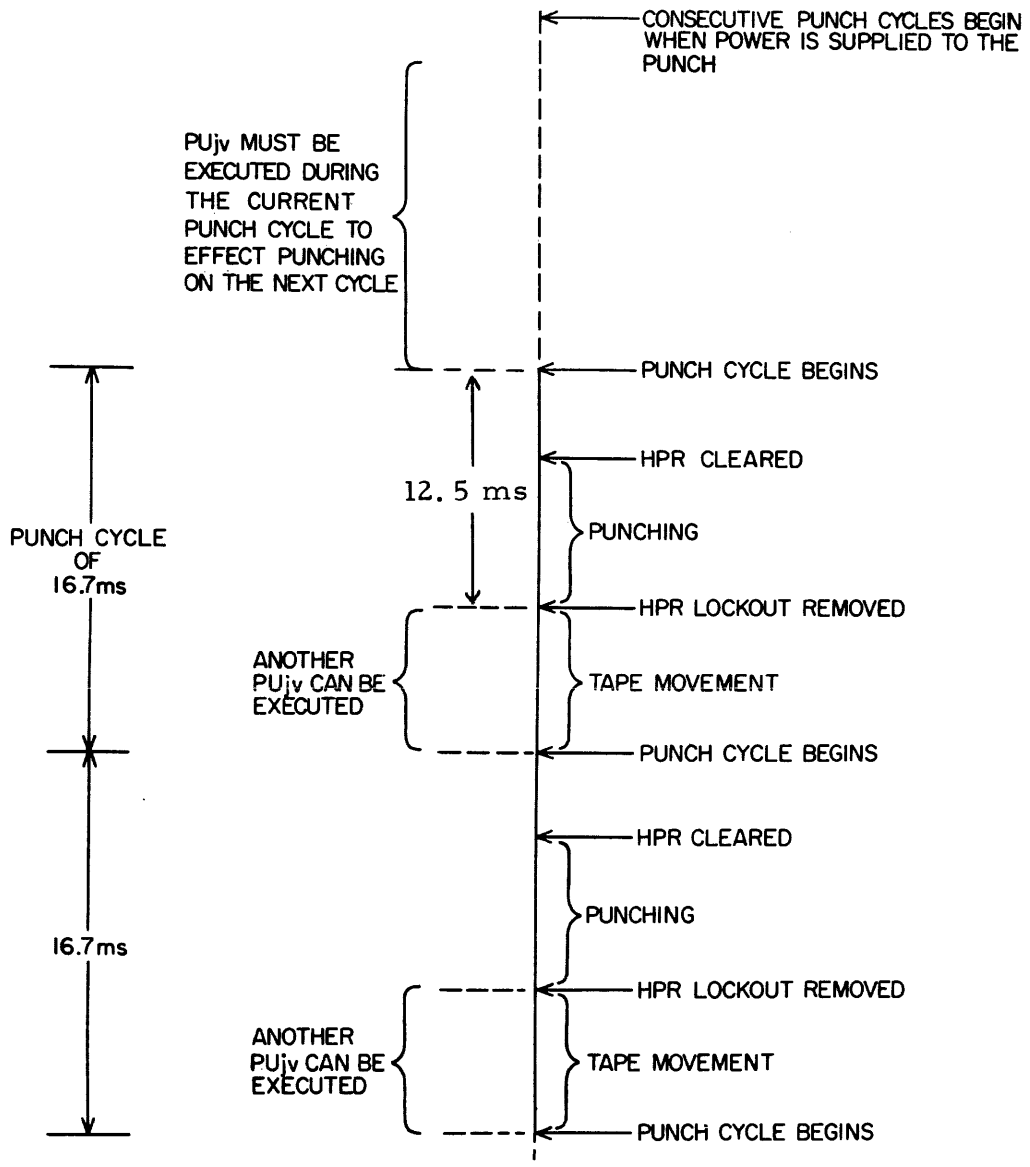


Figure 4-10

4-58. Figure 4-10 illustrates the following:

a. Any amount of computation time is available between Punch instructions if it is <u>not</u> desired that punching occur in <u>consecutive</u> punch cycles.

b. Punch instructions should be programmed for execution at less than 16.7ms intervals if punching is to occur during consecutive punch cycles.

4-59. The following instructions provide for punching one word.

Location	Op code	u-address	v-address	Explanation
a0	TP	c0	c1	Set working index
a1	TP	b0	Q	Transmit the word to be punched to the Q Register.
a2	LQ	Q	6	Shift the six bits to be punched to $Q_5 \ldots Q_0$
a3	PU	0	Q	Punch six bits
a4	IJ	c1	a2	Test for six frames punched
a5				Next instruction
b0				Word to be punched
c0	0	0	5	Index for six punches
c1				Working index

4-60. FAULT DETECTION. - No programming errors are detected by the High Speed Punch output sub-system.

4-61. MANUAL PREPARATION. The punch is ready for operation when the tape is properly positioned in the punch with a "leader" of blank tape preceding the first punching position. This leader may be fed through the punch by depressing either the TAPE FEED button on the computer control panel or the lever on the top of the punch. The punch is started by setting both the toggle switch on the punch and the TAPE PUNCH toggle switch on the lower left section of the computer control panel to their ON positions. Either switch in its OFF position stops the power supplied to the punch.

4-62. ELECTRIC TYPEWRITER.

4-63. GENERAL. (See figure 4-11.) The electric typewriter produces on-line typewritten manuscript as directed by the execution of appropriate Print instructions, PR-v. The principal computer components of this output sub-system are the Typewriter Register, (TWR) and its associated lockout circuitry. The external portion of this sub-system is an electric typewriter and its associated control circuitry. A brief description is given below for these principal parts.

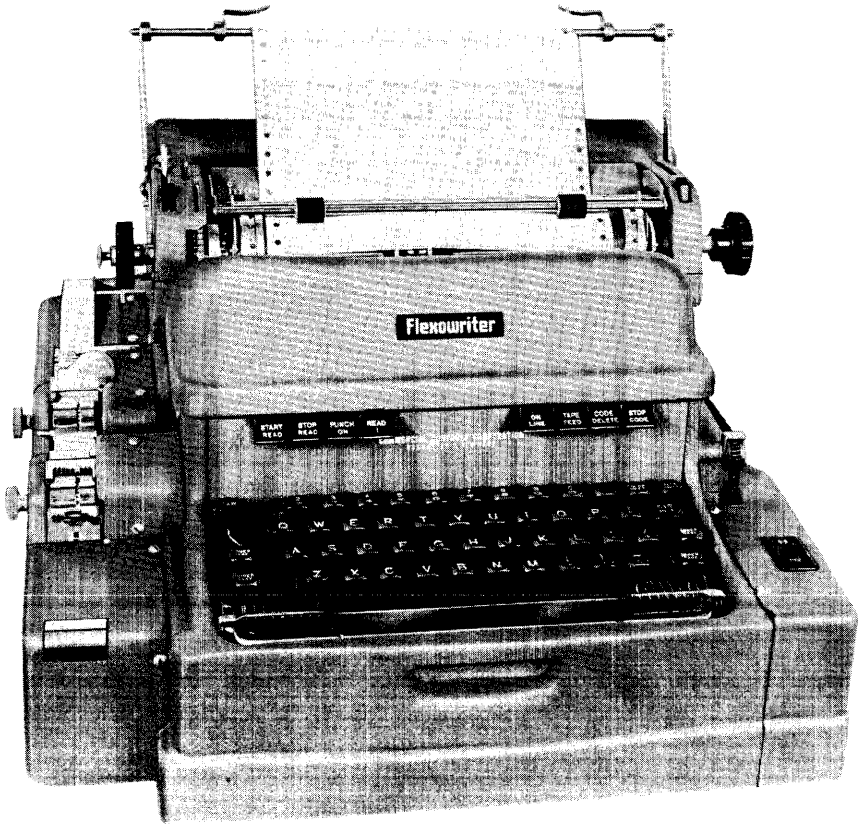


Figure 4-11

4-64. ELECTRIC TYPEWRITER. - The electric typewriter can be operated either manually from the keyboard or automatically by the computer. Manual operation might be utilized for typing column headings and manuscript identification.

4-65. The electric typewriter can perform 51 different operations. By the appropriate execution of Print instructions, 49 of these operations can be performed on-line at a rate of approximately nine per second. Standard 8½ by 11 inch paper can be employed. On the 10 inch roll paper commonly used, approximately 100 character positions are available across the width of the paper. Each instruction can cause the typing of a single letter, number or character, or the performance of a physical function as space, tab, carriage return, etc. A complete list of typewriter operations, and the codes to effect such typewriter operations, is given in Figure 4-12. It is possible to obtain a punched tape (typewriter code) copy of typewriter operations by properly setting a manual control on the typewriter.

TYPE LETTER UC	TYPE LETTER LC	OCTAL
A	a	30
B	b	23
C	c	16
D	d	22
E	e	20
F	f	26
G	g	13
H	h	05
I	i	14
J	j	32
K	k	36
L	l	11
M	m	07
N	n	06
O	o	03
P	p	15
Q	q	35
R	r	12
S	s	24
T	t	01
U	u	34
V	v	17
W	w	31
X	x	27
Y	y	25
Z	z	21

TYPE NUMBER UC	TYPE NUMBER LC	OCTAL
1	1	52
2	2	74
3	3	70
4	4	64
5	5	62
6	6	66
7	7	72
8	8	60
9	9	33
0	0	37

PERFORM TYPEWRITER OPERATION	OCTAL
SPACE	04
SHIFT UP	47
SHIFT DOWN	57
BACK SPACE	61
CAR. RETURN	45
TABULATOR	51
COLOR SHIFT	02
CODE DELETE*	77
STOP*	43

*FOR OFF-LINE OPERATION ONLY.

TYPE SYMBOL UC	TYPE SYMBOL LC	OCTAL
- (SUPERSCRIPT MINUS)	- (HYPHEN OR MINUS)	56
. (MULTIPLY)	= (EQUALS)	44
/ (VIRGULE)	+ (PLUS)	54
((OPEN PARENS)	, (COMMA)	46
) (CLOSE PARENS)	. (PERIOD)	42
_ (UNDERLINE)	↑ (ABSOLUTE	50

THE UPPER CASE, UC, OR LOWER CASE, LC, CHARACTER IS TYPED ACCORDING TO THE POSITION OF THE TYPE BARS.

Figure 4-12

4-66. TYPEWRITER REGISTER. - The Typewriter Register functions as a six-bit buffer storage register. The execution of a Print instruction sends "1's" to this register, via the X Register. The particular combination transmitted results in a six-bit code which specifies the typewriter operation to be performed. The Typewriter Register temporarily retains this code until the typewriter control circuitry has interpreted the code and initiated a typewriter cycle. The Typewriter Register is then cleared early in each typewriter cycle. Since only "1's" are transmitted to the Typewriter Register, TWR must be cleared prior to its receipt of each code. TWR lockout circuitry is provided for the computer to insure that a Print instruction does not cause data transmission to the Typewriter Register before (1) the typewriter operation initiated by a previous Print instruction is underway, and (2) TWR is cleared.

4-67. TWR lockout circuitry functions basically the same as each of the other output lockout circuits. During the execution of a Print instruction, before the "1's" held in the six lower order stages of X are transmitted to the Typewriter Register, the computer tests to see if the typewriter control ciruitry has actually initiated a previously specified typewriter operation; if it has not,

a TWR lockout exists for the computer, and the computer stalls. When the previous typewriter operation has been initiated, TWR is cleared and the typewriter control circuitry removes the TWR lockout condition from the computer. The execution of the current Print instruction is then completed: the computer transmits the "1's" held in X to the Typewriter Register, notifies the typewriter control circuitry that this data has been sent, and re-establishes the TWR lockout condition. Note that the computer is free for internal operations when it completes the execution of the Print instruction, but more data cannot be sent to the Typewriter Register until the typewriter control circuitry has cleared the Typewriter Register and removed the TWR lockout condition.

4-68. PROGRAMMING AND TIMING. - A Print instruction, PR-v, first directs the transmission of the contents of the v-addressed location to the X Register. If there is no TWR lockout condition, the computer transmits the "1's" stored in the six lower order bit positions of X to the Typewriter Register, notifies the typewriter control that this data has been sent, and re-establishes a TWR lockout condition for itself. When the typewriter control circuitry is notified that data has been sent to the Typewriter Register, a typewriter cycle is initiated and the typewriter is prepared to perform the operation specified by the code. If the code in the Typewriter Register is a legal code, the corresponding typewriter operation is begun. (If the code sent is not a legal code, no typewriter function occurs and the typewriter control circuitry is conditioned to indicate a Print fault.) After the TWR lockout for the computer is removed, a period of time is necessary to complete the current typewriter cycle. If another Print instruction is executed during this period, another typewriter operation will follow the first without delay. (This is so unless an illegal operation was detected during the last cycle. If this is the case, the TWR lockout for the computer is not removed during the cycle caused by a Print instruction with an illegal code.) Figure 4-13 illustrates the basic timing involved in typewriter output operations.

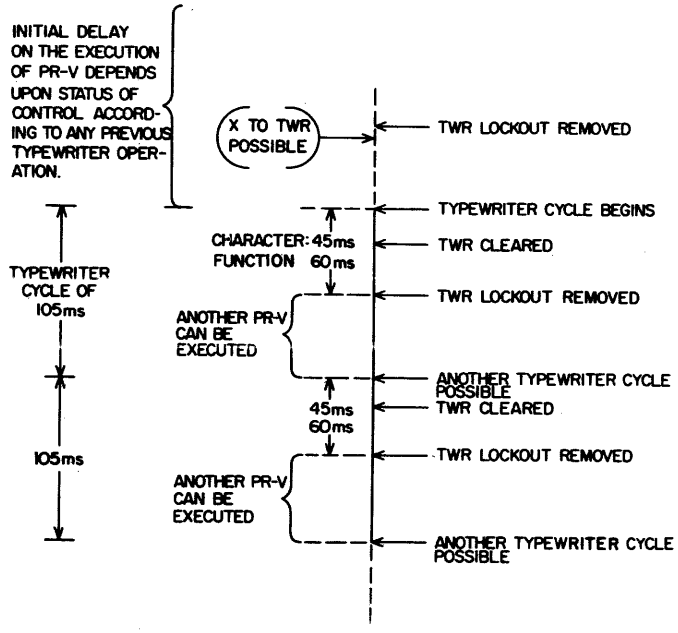


Figure 4-13

4-69. Figure 4-13 illustrates the following:

a. At least 105 ms internal computing time is available between successive typewriter operations. Since the typewriter tends to do one cycle and stop, any amount of computation can be done between PR-v instructions.

b. If PR-v instructions are programmed at intervals of less than 105 ms, typewriter cycles are continuous.

4-70. The following conventions should be observed when programming for typewriter output.

a. A PR-v instruction is required for each typewriter operation. Care should be taken that only legal codes are programmed in v_5 - v_0 when this instruction is executed.

b. Note that all the typewriter codes for letters, numbers, and signs have upper and lower case characters. The character typed is determined by the location of the type bars in their upper and lower case position. To produce the upper case character, precede the character code by a SHIFT UP code. To resume typing in lower case, a SHIFT DOWN code must precede the first code to be typed in lower case. For example, to type the single upper case letter M, the following two instructions would be programmed:

PR-v, where $v_5 \ldots v_0$ is 100 111 SHIFT UP

PR-v, where $v_5 \ldots v_0$ is 000 111 Type M

Subsequent print instructions will continue upper case typing until a

PR-v, where $v_5 \ldots v_0$ is 101 111 (SHIFT DOWN) appears.

c. A movement left of the carriage by one space to position it for the next typing operation occurs automatically after each printing of a letter, number, or character.

d. The format for the typewritten copy is programmed by proper use in an output routine of such typewriter functions as space, tab, carriage return, etc. The typewriter has been known to fail to print the first character following a carriage return. This condition can be corrected by maintenance procedures. A typewriter function such as a space could be programmed as the first character on the line to eliminate the possibility of losing a printed character.

e. Associated with each carriage return operation is a line spacing operation. This determines the number of spaces between consecutive lines in the typewritten copy. This function is not programmed. The line spacer is set manually prior to initiation of output typing.

f. The tabulator operation also requires a manual setting, since the interpretation of the TAB code merely releases the carriage to move to a predetermined manual tab setting.

g. As long as power is supplied to the typewriter, the typewriter keys are operative. Care should be taken to do any manual typing required for format headings or manuscript identification prior to the time output from the computer is to occur. The keyboard should not be touched inadvertently at any other time or erroneous data will be introduced into the manuscript.

h. The CODE DELETE and STOP codes have no meaning during on-line use of the typewriter. If these codes are programmed in a Print instruction, a Print fault occurs. The CODE DELETE code is used to cover an error in the punched tape during off-line preparation of the paper tape. The STOP code in a punched tape stops off-line generation of a duplicate tape, or off-line creation of a printed copy.

4-71. FAULT DETECTION.

a. PRINT FAULT. - This fault is detected when an illegal code is held in the Typewriter Register. This fault automatically stops typewriter activity and stops the computer in an A Fault condition. Since the Typewriter Register is cleared, no indication of the illegal code is given. The computer stop is not immediate. Hence, an additional Print instruction may be attempted before the fault stop. If this is the case, the computer is stalled since the TWR lockout established during the last cycle has not been removed.

b. POWER LOSS. - If the power supply to the typewriter fails, the computer stops in an A-fault condition. The typewriter stops immediately. If voltage fails within the typewriter, the typewriter finishes the current cycle and stops. The computer continues in operation. Another Print instruction attempted will be executed but no typewriter operation occurs and the TWR lockout for the computer is not removed. When and if another Print instruction is attempted, the computer is locked out and stalls.

4-72. MANUAL PREPARATION.

a. Insert paper.

b. Set line spacer and tabs as required.

c. Turn typewriter power on by setting OFF-ON switch to the right of the typewriter to ON position.

d. Manually type any desired titles or column heads not provided for by the program.

e. Depress ON-LINE tab on the front of the typewriter.

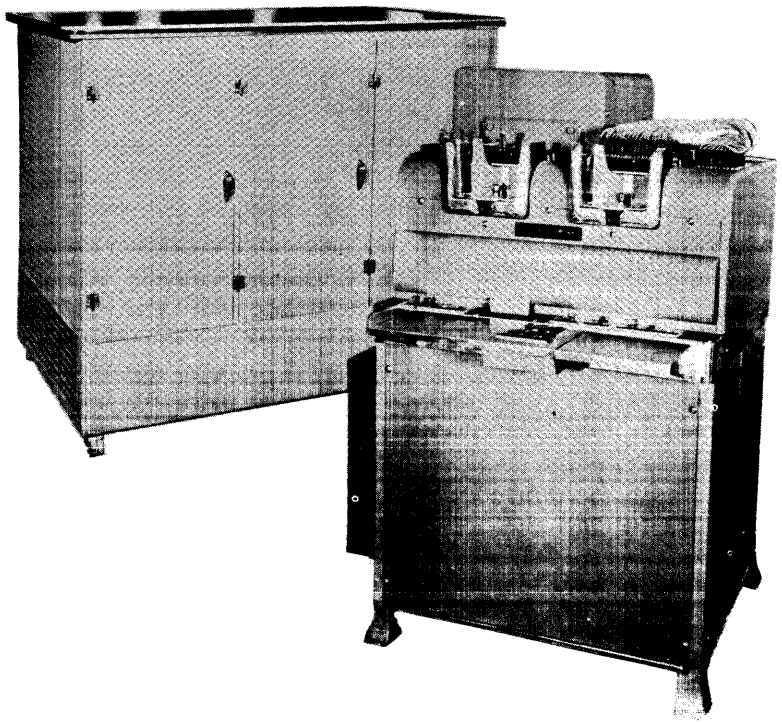
f. If a punched paper tape copy of printed information is desired, depress PUNCH ON tab on the front of the typewriter. Place paper tape in the punching mechanism.

g. When finished using the typewriter, turn typewriter power off by setting OFF-ON switch to OFF position.

4-73. UNIVAC CARD EQUIPMENT.

4-74. GENERAL. (See figure 4-14.) The Univac Card Equipment provides the Univac Scientific with punched card input and output. External Function instructions control the start-stop action of the card unit and the passage of cards through the equipment. External Read and External Write instructions are appropriately executed to process data for this equipment. The card unit equipment has two modes of operation: SINGLE CARD and FREE RUN. In both modes the Interrupt feature can be employed to automatically notify the computer when data transmissions are to occur. The Univac Card Equipment is composed of the punched card unit and the punched card unit control cabinet. This equipment is capable of punching or reading standard 12 row, 80 column tabulating cards at a maximum rate of 120 cards per minute. Cards can be punched and read simultaneously at this rate so that a total of 240 cards per minute can be processed.

Figure 4-14



4-75. PUNCHED CARDS. - Rectangular indices or holes punched in the card in a predetermined configuration represent the input or output data. In general this data is alphabetical as well as numerical. Each card provides a space for indices in 80 vertical columns and 12 horizontal rows. A typical card is shown in figure 4-15. The columns are numbered 1 through 80 from left to right. Individual index positions in the lower ten rows of each column are numbered and clearly defined. Corresponding index positions are available in the top two rows, 12 and 11, although they are not printed on the card. A hole punched in an index position is regarded by the machine as a binary "1"; the absence of a hole, as a binary "0".

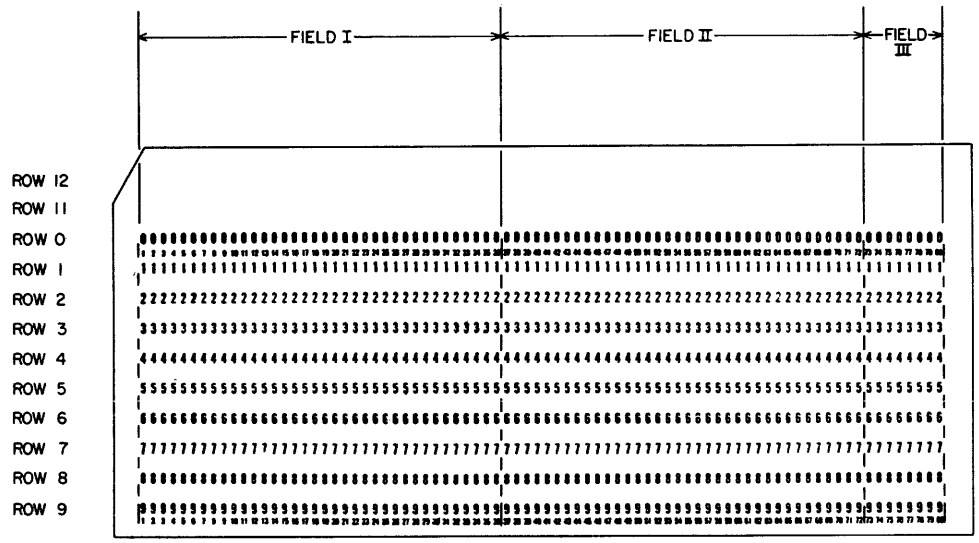


Figure 4-15

4-76. In both input and output operations data is processed on a row by row basis. Each of the 12 rows of a card is logically divided into 3 groups of columns, called "fields". Field I consists of columns 1 through 36; Field II consists of columns 37 through 72; and Field III consists of columns 73 through 80. A switch in the card unit control cabinet can be set so that during any card run the programmer has the option of processing all 80 columns on each card (Fields I, II, and III), or columns 1 through 72 (Fields I and II). The reading and punching mechanisms interpret the contents of Fields I and II as 36 bit words and the contents of Field III as an eight bit "word". A decoding or encoding routine may be necessary before punching, or after reading.

4-77. Card formating is done by columns. Usually a decimal or octal digit is represented in a column by a single index, positioned so that the punched row number corresponds to the value of the digit. Codes for alphabetical information are more arbitrary. An alphabetical character is represented in a column by a one or two index combination which usually consists of a maximum of one index in rows 0, 11, and 12 and a minimum of one index in rows 1 through 9. Figure 4-16 is an example of one representation of alphabetical and numerical characters on a card. Since cards are punched and read by rows, and data is represented usually by columns, computer programs involving card input and output must provide the necessary interpretation. A mechanical restriction limits the number of punches per card. A safe limit to observe is 60 punches per row and 160 punches per card.

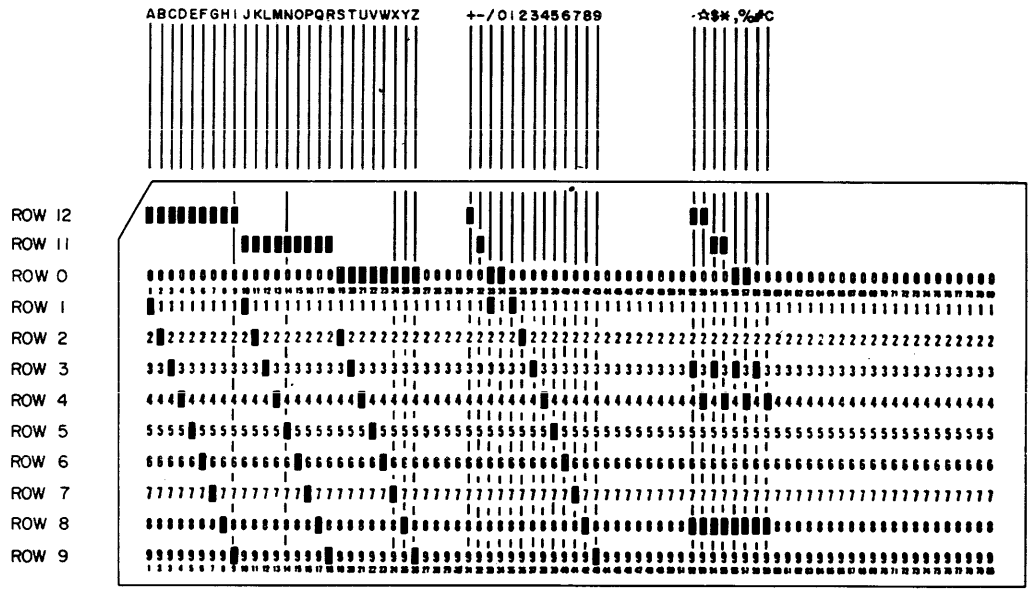


Figure 4-16

4-78. OPERATING THE CARD EQUIPMENT. The card equipment is set in operation under computer control by the execution of programmed External Function instructions. These EF-v instructions may be coded to

a. select the card unit for operation

b. control its start-stop mechanisms;

c. provide for all card movement through the card unit; and

d. specify punching and/or reading operations as required.

If merely card movement is desired, only the appropriate External Function instructions need be executed. When punching and/or reading operations are also to occur, External Write and/or External Read instructions must also be executed by the computer at the proper times to achieve the necessary data transmissions. Since the card unit can utilize the Interrupt feature, the computer can be notified each time a row is to be processed.

4-79. To reference the card equipment, the code word sent to IOB by an External Function instruction must always provide a "1" for IOB_{35}. The code word may also provide "1" for IOB_7, and IOB_5 through IOB_0. The "1" in IOB_{35} is used to start and prepare the equipment for the function specified by the combination of "1's" in IOB_7, and IOB_5 through IOB_0. When the card unit is started, an electromagnetic clutch engages and causes an 18 point sequence of operations called a card cycle. If IOB_5=0, this clutch causes a single card cycle and then disengages. The IOB selections made are then dropped, and the card unit stops automatically. Such operation is the SINGLE CARD mode of operation. If IOB_5=1, the clutch causes continuous card cycles, and all selections made for operation are retained until an External Function instruction to disengage the clutch is executed. Such operation is the FREE RUN mode of operation.

4-80. Manually operated switches and buttons are provided to simulate some of the operations provided by the External Function instructions. These manual controls thus enable an operator to run the card unit apart from the computer. Manual operation is convenient at times for positioning cards, emptying cards out of the card unit, etc.

4-81. CARD MOVEMENT. - The card unit (figure 4-17) is a device with two card channels: a right-hand (reading) channel for cards that supply data to the computer; and a left-hand (punching) channel for cards that receive output data from the computer. A top view sketch of each of these channels is given in Figure 4-18. Note that each card channel has five card positions between feed hopper and receiving stacker. Card movement is controlled by a knife edge under each feed hopper and sets of rollers in each card channel. A set of rollers is associated with each station in each channel. Depending upon the IOB selections, various roller and knife-edge actions are effected. Hence all the rollers in a channel can operate, or only some, depending on the required card movement. A set of rollers operates to move a card one station. During every card cycle, rollers in stations 2, 3, 4, and 5 of both the punching and reading channels move any card in their station to the next station. During card cycles initiated by an External Function instruction which places a "1" in IOB_2 (PICK READ CARD bit), the knife edge under the read feed hopper and the roller in station 1 of the reading channel <u>also</u> operate. During card cycles initiated by an External Function instruction which places a "1" in IOB_3 (PICK PUNCH CARD bit), the knife edge under the punch feed hopper and the rollers in punch station 1 <u>also</u> operate.

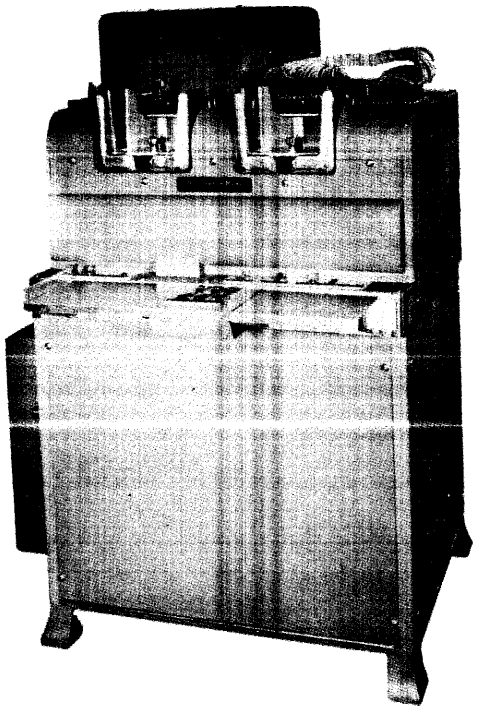
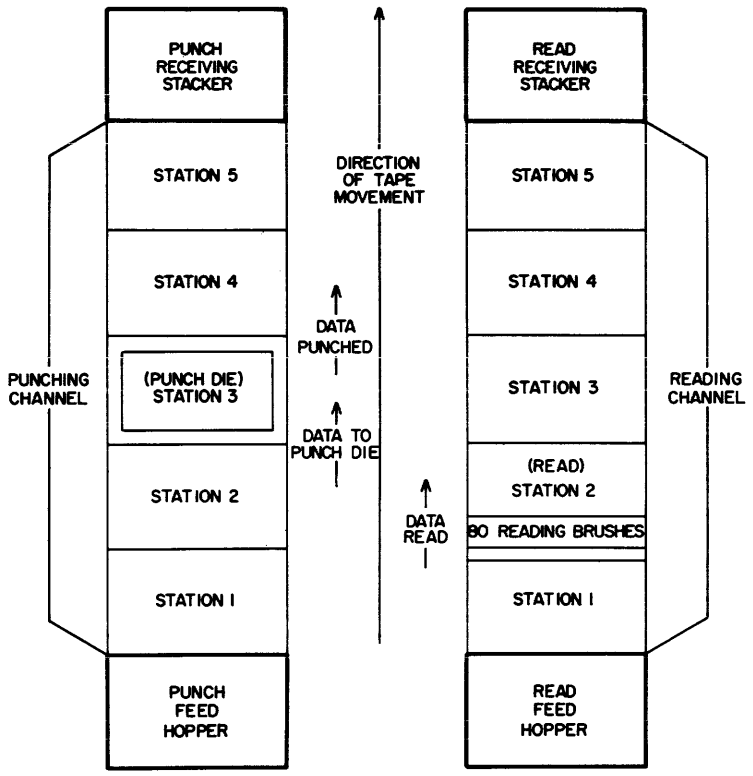


Figure 4-17

Figure 4-18



4-82. SELECTIONS FOR CARD CYCLE OPERATIONS. - The 18 points of a card cycle are assigned identifying numbers as shown in figure 4-19.

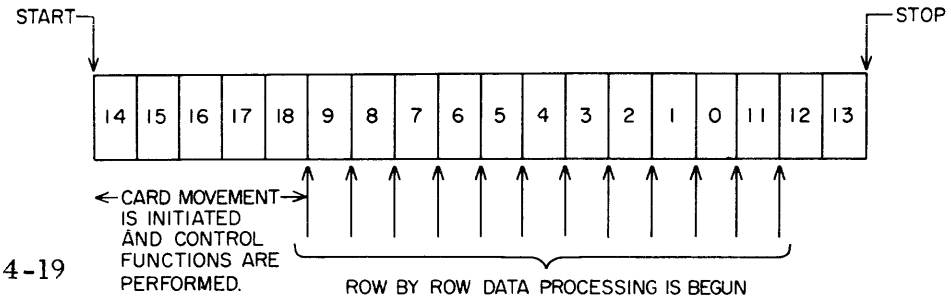


Figure 4-19

At the start of a cycle, the cards are at rest and the card unit is at point 14. Points 14 through 18 establish selections for particular card unit functions, initiate card movement as required, and perform other control functions. During each of the next 12 points a row of information can be processed; these points are numbered in correspondence to the card rows 9, 8, 7, 6, 5, 4, 3, 2, 1, 0, 11, 12. Toward the end of point 12 in the SINGLE CARD mode, selections are dropped, the clutch disengages*, and the card unit stops at the completion of point 13. It is thus necessary when operating in the SINGLE CARD mode to supply all selections for each card cycle. In the FREE RUN mode of opera-

* Perhaps "tends to disengage" is a better phrase, since the SINGLE CARD mode of operation can be made to simulate the FREE RUN mode by appropriately introducing the START SINGLE CARD mode for a next card cycle during point 12 of the current cycle.

tion, selections are not dropped and the clutch does not disengage. Point 14 for another cycle follows point 13, without interruption. In the FREE RUN mode, then, selections need only be programmed once when starting the first card cycle.

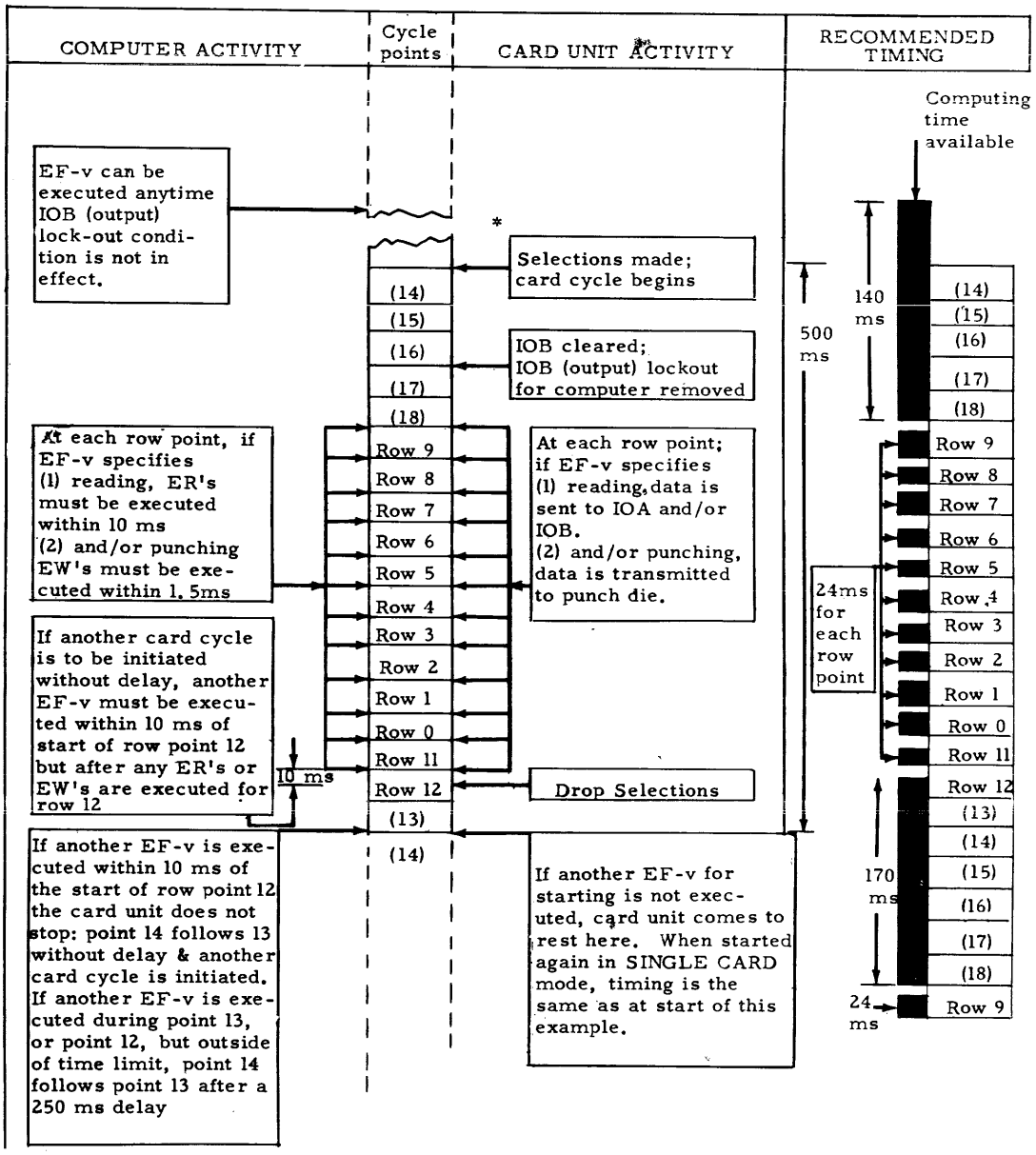
4-83. The PUNCH CARD bit, IOB_1=1, provides for setting up the punch die with the information in IOB (and IOA), as placed there by the execution of External Write instructions. The transmission of data occurs on cycle points 9, 8, ..., 1, 0, 11, 12. The card to be punched is moved automatically into punch station 3. This card is actually punched on cycle point 15 of the next cycle (after which the punched card moves into punch station 4).

4-84. The READ CARD bit, IOB_0=1, provides for reading a card as it moves into read station 2. A PICK READ CARD bit provides the card movement from read station 1 to read station 2. As the card is moved past the reading brushes, the data is sensed on cycle points 9, 8, ..., 1, 0, 11, 12, and sent to IOB (and IOA). External Read instructions remove data from IOB (and IOA).

4-85. The INTERRUPT bit, IOB_7=1, provides for a signal to be sent to the computer interrupt control at the outset of the processing of each row in a reading, punching, or punching/reading operation. The interrupt signal is also provided, without a programmed selection, if the Interrupt switch on the card equipment is set to its "Locked In" position. Thus, if the interrupt signal is only desired as programmed, the Interrupt switch must be set to its "By Command" position.

4-86. Selections for the FREE RUN mode are established during the first card cycle and then maintained. To stop a Free Run operation, a second External Function instruction must be executed. The code word for this last External Function instruction must provide a STOP FREE RUN bit, IOB_4=1. This EF instruction must be executed just before the last card (or set of cards in combined punching/reading) is to be processed; i.e., before the last card cycle of Free Run operation begins. The card unit will then stop during the last card cycle after data processing is completed.

4-87. PROGRAMMING AND TIMING. - Timing for the SINGLE CARD and FREE RUN modes of operation is illustrated in the diagrams following (figure 4-20 and 4-21). In regard to these figures, the following comment is made. The card unit is adjusted to perform 120 card cycles per minute. Theoretically then, each of the 18 points in the card cycle consumes 500/18 or 27.8ms. Because of the mechanical nature of the card unit, the theoretical time for each card cycle point is not always realized. The times listed in the column labelled TIMING show realistic and recommended timing for safe programming. Differences existing between these times and the theoretical times are deliberate and are for the purpose of insuring synchronization of the electromechanical, mechanical, and electronic components involved.

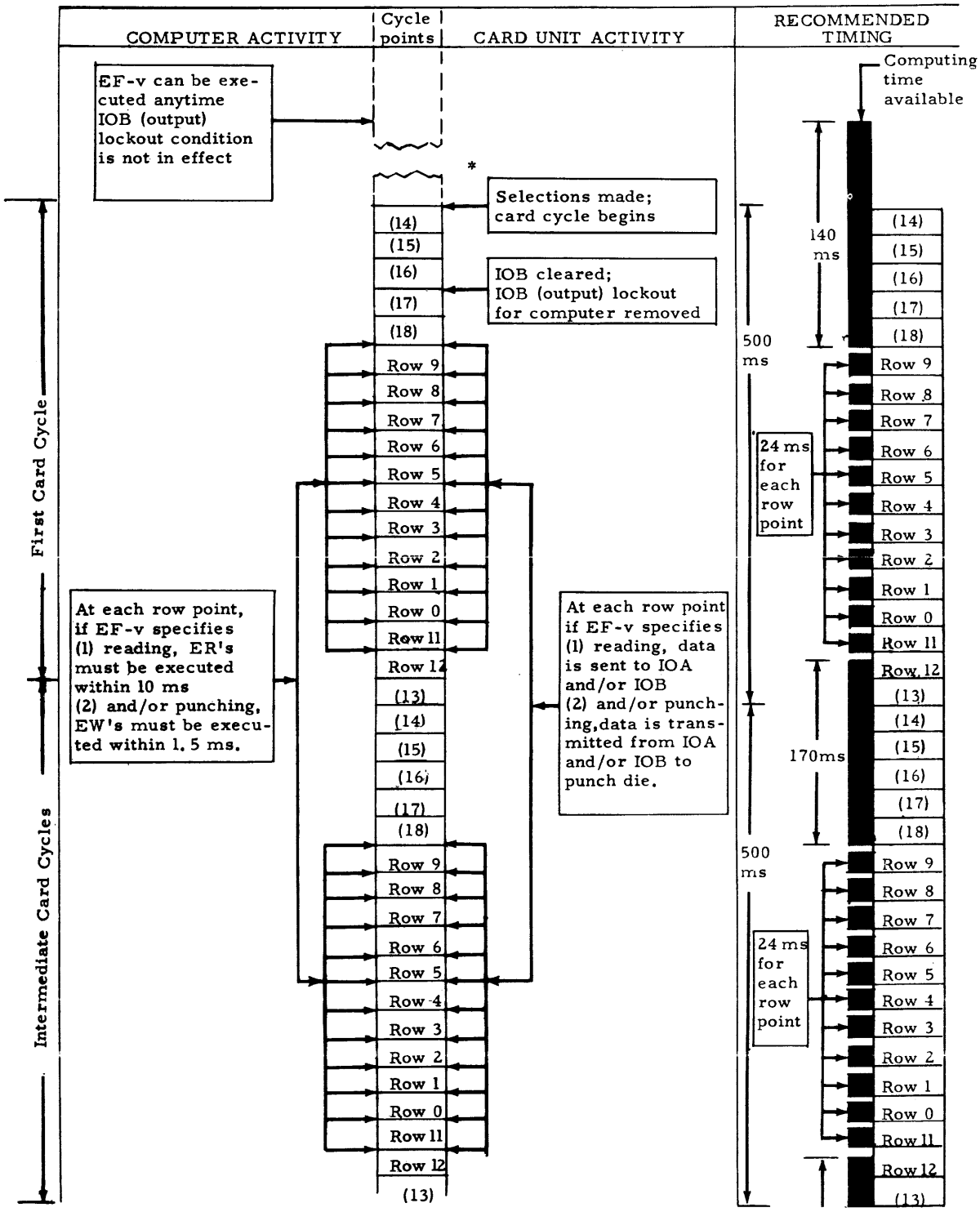


* The maximum time required to engage the clutch after the execution of an EF-v is 250 ms. Therefore a delay of from 0 to 250 ms between the execution of the EF-v instruction and the beginning of point 14 occurs here.

Figure 4-20

4-88. In any of the following discussion of card equipment operation, it is assumed that the necessary manual preparation for card unit operation is completed. Thus, it is assumed that power is supplied to the equipment, that the Field III switch is set for the desired number of columns (72 or 80) to be processed on each card, that cards are properly positioned in the feed hoppers, and that all switches on the card unit control panel are appropriately set for operation under computer control.

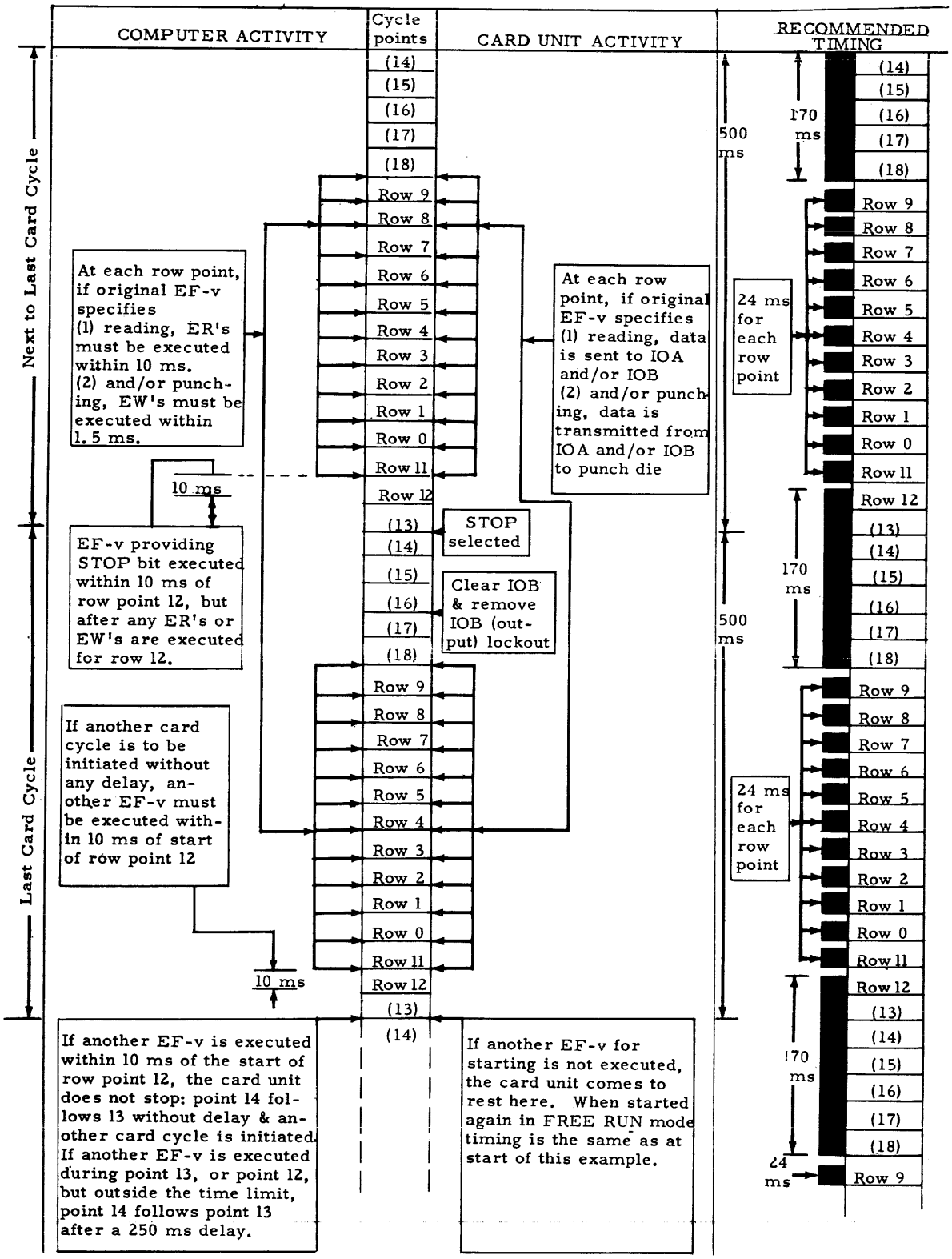
Figure 4-21 TIMING FOR FREE RUN MODE



* The maximum time required to engage the clutch after the execution of an EF-v is 250 ms. Therefore a delay of from 0 to 250 ms between the execution of an EF-v and the beginning of point 14 occurs here.

(continued on next page)

TIMING FOR FREE RUN MODE
(continued from previous page)



4-89. When reading data from cards, three External Read instructions must be executed for each row if all three fields of a card are to be read. Data is read from each row on the cards in the order Field I and Field III, then Field II. The External Read instructions are usually programmed consecutively in the order ERjv (j=0) for Field III, ERjv (j=1) for Field I, and ERjv (j=1) for Field II. At least the first two instructions must be executed within 10 ms from the time when the row point begins. If the instruction for reading Field I data from IOB is not executed within 10ms, a No Information fault occurs. The data in Field III in the row is sensed if the Field III switch in the card unit control cabinet is set to its NORMAL position. If this switch is set to its OUT position, IOA receives no information from Field III, and the IOA read lockout is not removed. If an attempt is made to execute an ERjv with j=0, the computer will stall. This stall will lead eventually to a No Information fault. Thus, if Field III is not to be read, two External Read instructions with j=1 must be programmed for each row. At least the first ERjv must be executed within 10 ms after the row point begins.

4-90. When punching cards, three External Write instructions must be executed for each row if all three fields of the card are to be punched. Data is punched in each row in the cards in the order Field I and Field III, then Field II. The External Write instructions are usually programmed consecutively in the order EWjv (j=0) for Field III, EWjv (j=1) for Field I, and EWjv (j=1) for Field II. At least the first two instructions must be executed within 1.5 ms from the time when the row point begins. If data for Field I is not placed in IOB by EWjv (j=1) within 1.5 ms, a No Information fault occurs. Field III is punched only if the Field III switch within the card unit control cabinet is set to its NORMAL position. If this switch is set to its OUT position, the punch elements for Field III are not positioned regardless of the execution of EWjv with j=0. It is well to remember that actual punching of a card occurs at the beginning of the cycle following the cycle during which the External Write instructions provided the data to be punched. The punch elements for all rows of the card are set up before any actual punching occurs.

4-91. In a combined punching and reading operation, either all three fields are both punched and read, or Fields I and II are both punched and read. This is according to the setting of the Field III switch in the card unit control cabinet to either its NORMAL or OUT position. If Field III is to be punched and read, External Write and External Read instructions should be programmed consecutively for each row in the order EWjv (j=0), EWjv (j=1), EWjv (j=1), ERjv (j=0), ERjv (j=1), ERjv (j=1). The first EWjv with j=1 must be executed within 1.5 ms after the beginning of the row point, or a No Information fault will occur. No provision is made to detect whether the succeeding instructions are executed in time to correctly process the data. Therefore, all External Write and External Read instructions for each row should be executed consecutively, with the External Write instructions preceeding the External Read instructions. These same statements apply to simultaneous punching and reading of Field I and II except that only two EWjv's and two ERjv's with j=1 need be executed.

4-92. If the interrupt option is chosen for use with the card equipment, either by programmed or manual selection, the interrupt signal is sent to the interrupt control in the computer at the beginning of each row point. From this moment, program time is restricted to the time limits given in the preceeding paragraphs for (1) the interruption of the program in process and (2) whatever computation is necessary up to and including the execution of the instruction to process Field I.

4-93. SAMPLE CARD READ PROGRAM. -See page 4-36 for the program for reading consecutive cards, in the single card mode of operation.

To transmit data from n consecutive cards, with continuous card unit operation, a routine such as the following can be used. At the conclusion of this program, the n^{th} card (last card read) is found in the read receiving stacker; an $(n+1)^{th}$ card is in the first reading station.

4-94. A program to read n consecutive cards, in the free run mode of operation, executes the same number of External Read instructions, with the same timing restrictions on their execution. However, only one External Function instruction need be programmed to provide for reading n cards. The code word at v for such an EF-v is (v) = 400000000045 which provides Start, Free Run, Pick Read Card, and Read bits. This instruction would replace the second EF-v of the following program. To stop reading, an External Function instruction must be executed to provide a Stop bit. Such an EF-v would have (v) = 400000000020. This instruction must be executed within 10ms of the beginning of row point 12 of the next to the last card desired to be read. No additional External Functions are necessary between the EF providing the Free Run bit and the EF providing the Stop bit.

Location	Op Code	u-address	v-address	Explanation	
	EF	00000	v	where (v) =400000000004 Start, pick read card	1 cycle
	. . .			. . .	
	EF	00000	v	where (v) =400000000005 Start, pick read card, and read	1 cycle
	. . .			Other instructions requiring execution time less than 140ms.	
	ER	00000	v	Read Field III, row 9	
	ER	10000	v	Read Field I, row 9	
	ER	10000	v	Read Field II, row 9	
	. . .			Repeat ER's for rows 8, 7, . . . 1, 0, 11, 12, each repetition being initiated in less than 10ms after the beginning of the corresponding row point.	
	EF	00000	v	where (v) =400000000005 Start, pick read card, and read This instruction must be executed within 10ms of the beginning of row point 12 of the previous cycle.	n-1 cycle
	. . .			Other instructions requiring execution time less than 170ms	
	ER	00000	v	Read Field III, row 9	
	ER	10000	v	Read Field I, row 9	
	ER	10000	v	Read Field II, row 9	
	. . .			Repeat ER's for rows 8, 7, . . . 1, 0, 11, 12, each repetition being initiated in less than 10ms after the beginning of the corresponding row point.	
	. . .			Execute instructions between set of arrows above, n-2 more times, observing all time limits quoted.	
	EF . . .	00000	v	where (v) =400000000000 Start Execute this instruction four times, each execution occurring within 10ms of row point 12 of the previous cycle.	4 cycles

4-95. SAMPLE CARD PUNCH PROGRAM. -See page 4-38 for the program for for punching consecutive cards, in the single card mode of operation.

To punch data in n consecutive cards, with continuous card unit operation, a routine such as the following can be used. At the conclusion of this program the n^{th} card (last card punched) is in the punch receiving stacker; an $(n+1)^{th}$ card is in Station 5, and an $(n+2)^{th}$ is in Station 1.

4-96. A program to punch n consecutive cards, in the free run mode of operation, executes the same number of External Write instructions, with the same timing restrictions on their execution. However, only one External Function instruction need be programmed to provide for punching n cards. The code word at v for such an EF-v is (v) = 400000000052 which provides Start, Free Run, Pick Punch Card, and Punch bits. This instruction would replace the third EF-v of the following program. To stop punching, an External Function instruction must be executed to provide a Stop bit. Such an EF-v would have (v) = 400000000020. This instruction must be executed within 10ms of the beginning of row point 12 of the next to the last card desired to be punched. No additional External Functions are necessary between the EF providing the Free Run bit and the EF providing the Stop bit.

Location	Op Code	u-address	v-address	Explanation	
	EF	00000	v	where (v) =400000000010 Start, pick punch card	1 cycle
	. . .			. . .	
	EF	00000	v	where (v) =400000000010 Start, pick punch card	1 cycle
	. . .			. . .	
	EF	00000	v	where (v) =400000000012 Start, pick punch card, and punch	
	. . .			Other instructions requiring execution time less than 140ms	
	EW	00000	v	Punch Field III, row 9	
	EW	10000	v	Punch Field I, row 9	1 cycle
	EW	10000	v	Punch Field II, row 9	
	. . .			Repeat EW's for rows 8, 7, . . . , 1, 0, 11, 12, each repetition being initiated in less than 1.5ms after the beginning of the corresponding row point.	
	EF	00000	v	where (v) = 400000000012 Start, pick punch card, and punch . This instruction must be executed within 10ms of the beginning of row point 12 of the previous cycle.	
	. . .			Other instructions requiring execution time less than 170ms	
	EW	00000	v	Punch Field III, row 9	
	EW	10000	v	Punch Field I, row 9	
	EW	10000	v	Punch Field II, row 9	
	. . .			Repeat EW's for rows 8, 7, . . . , 1, 0, 11, 12, each repetition being initiated in less than 1.5ms after the beginning of the corresponding row point.	n-1 cycles
	. . .			Execute instructions between set of arrows above, n-2 more times, observing all times limits quoted.	
	EF	00000	v	where (v) = 400000000000 Start (n^{th} card is punched during this cycle)	
	. . .			Execute this instruction three times, each execution occurring within 10ms of row point 12 of previous cycle.	3 cycles

4-97. SAMPLE PUNCH AND READ CARDS PROGRAM. -See page 4-4 for the program for simultaneous punching and reading consecutive cards, in the single card mode of operation.

To punch data in n consecutive cards, and simultaneously read data from n consecutive cards, with continuous card unit operation, a routine such as the following can be used. At the conclusion of this program, the last card punched, $n^{\underline{th}}$ card, is in the punch receiving stacker; an $(n+1)^{\underline{th}}$ card is in punch station five; and an $(n+2)^{\underline{th}}$ card is in punch station one. Also, the last card read, $n^{\underline{th}}$ card, is in the read receiving stacker, and an $(n+1)^{\underline{th}}$ card is in read station one.

4-98. A program to punch and read simultaneously n consecutive cards, in the free run mode of operation, executes the same number of External Write and External Read instructions, with the same timing restrictions on their execution. However, only one External Function instruction need be programmed to provide for punching and reading n cards. The code word at v for such an EF-v is (v) = 400000000057. This code word provides Start, Free Run, Pick Punch Card, Punch, Pick Read Card, and Read bits. This instruction would replace the third EF-v of the following program. To stop simultaneous punching and reading, an External Function instruction must be executed to provide a Stop bit. Such an EF-v would have (v) = 400000000020. This instruction must be executed within 10ms of the beginning of row point 12 of the next to the last card desired to be punched. No additional External Functions are necessary between the EF providing the Free Run bit and the EF providing the Stop bit.

Location	Op Code	u-address	v-address	Explanation	
	EF	00000	v	where (v) =400000000010 Start, pick punch card	1 cycle
	. . .			. . .	
	EF	00000	v	where (v) =400000000014 Start, pick punch card, pick read card	1 cycle
	. . .			. . .	
	EF	00000	v	where (v) =400000000017 Start, pick punch card, punch, pick read card, read	
	. . .			Other instructions requiring execution time less than 140ms	
	EW	00000	v	Punch Field III, row 9	
	EW	10000	v	Punch Field I, row 9	
	EW	10000	v	Punch Field II, row 9	1 cycle
	ER	00000	v	Read Field III, row 9	
	ER	10000	v	Read Field I, row 9	
	ER	10000	v	Read Field II, row 9	
	. . .			Repeat EW's and ER's for rows 8, 7, . . . , 1, 0, 11, 12, each repetition being initiated in less than 1.5ms after the beginning of the corresponding row point	
	EF	00000	v	where (v) = 400000000017 Start, pick punch card, punch pick read card, read This instruction must be executed within 10ms of the beginning of row point 12 of the previous cycle	
	. . .			Other instructions requiring execution time less than 170 ms	
	EW	00000	v	Punch Field III, row 9	
	EW	10000	v	Punch Field I, row 9	
	EW	10000	v	Punch Field II, row 9	n-1 cycles
	ER	00000	v	Read Field III, row 9	
	ER	10000	v	Read Field I, row 9	
	ER	10000	v	Read Field II, row 9	
	. . .			Repeat EW's and ER's for rows 8, 7, . . . , 1, 0, 11, 12, each repetition being initiated in less than 1.5ms after the beginning of the corresponding row point.	
	. . .			Execute instructions between set of arrows above, n-2 more times, observing all time limits quoted.	
	EF	00000	v	where (v) = 400000000000 Start Execute this instruction four times, each execution occurring within 10ms of row point 12 of the previous cycle.	4 cycles

4-99. FAULT DETECTION. - Any stop of computer operation during actual card unit operation, whether the computer is stopped by a fault condition or manually, will probably lead to the generation of a card equipment fault. In general, a computer stop does not stop card unit operation unless the card equipment is operating in the Step mode. If the card equipment is in the Step mode operation when a computer stop occurs, card equipment operation is stopped at the end of the current cycle. If the card equipment is in Free Run mode operation when a computer stop occurs, the card unit operation usually continues until either the MASTER CLEAR button on the computer console is depressed, or the STOP button on the card unit control panel is depressed.

4-100. Card equipment faults which are the direct cause of computer faults and the stop of computer operation, are listed subsequently. The type of fault is indicated by the illumination of the appropriate light on the card unit control panel shown in figure 4-22.

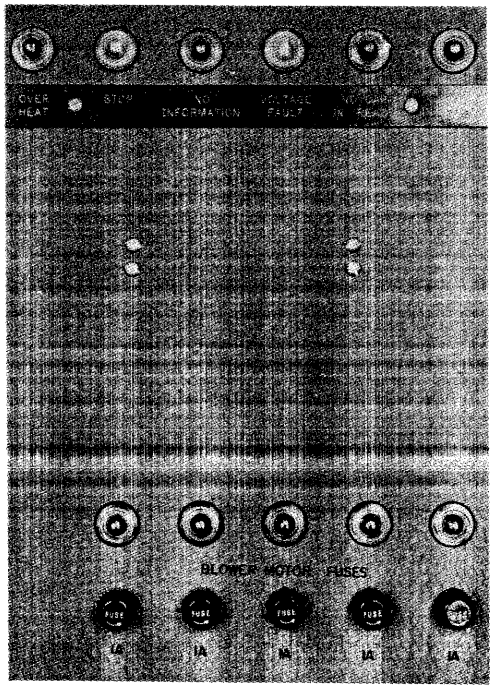


Figure 4-22

4-101. NO INFORMATION FAULT. - A computer B Fault results from one of the following situations.

a. Reading operation: External Read instruction (j=1) for processing Field I input data not executed within 10ms of the beginning of the row point;

b. Punching operation: External Write instruction (j=1) to supply data for Field I not executed within 1.5ms of the beginning of the row point;

c. Punching/Reading operation: External Write instruction (j=1) to supply data for Field I not executed within 1.5 ms of the row point.

4-102. The computer is stopped immediately upon detection of the faults; i.e., as soon as the time limit is exceeded, the computer is notified and stopped in a fault condition. Note that no fault is incurred by the failure to process Field II in time.

4-103. NO CARD IN READER. - A cycle programmed to read information from a card is initiated by an EF-v instruction where the contents of v includes the read bit. During this cycle External Read instructions are executed to read information from IOB, as transmitted there from the punched card. If there is no card moving through the read station during the cycle when the card is programmed to be read, a computer B Fault is incurred. The lack of a card is detected midway of the card cycle (point 5). After some necessary delay time internal to the card control unit, the computer is notified of the fault and is stopped. As long as the card cycle continues, the read brushes sense all 1's as if a card were being read with a hole in each index position. Rows of 1's are transmitted to the computer memory by the External Read instructions which are executed before the computer stop. It is unlikely, however, that a complete "card image" of 1's will be stored and processed before the computer stop.

4-104. NO CARD IN PUNCH. - A cycle programmed to set up the punch die is initiated by an EF-v instruction where the contents of v include the punch bit. During this cycle External Write instructions are executed to transmit the information to the punch. If there is no card in punch station two to be moved to station three for punching at the beginning of the next cycle, a computer B Fault is incurred. The lack of a card is detected early in the cycle (point 15). After some necessary delay time internal to the card control unit, the computer is notified of the fault and is stopped. If the time available between the start of the cycle and the first EW instruction is utilized for other computation, the computer is stopped before the first EW instruction is executed. If this computation time is not utilized, the computer is stalled before the stop by the attempt to execute EW instructions before the card unit is ready to accept information. The card unit is stopped in both step and free run modes of operation at the end of the cycle during which the punch die normally would have been set up with information.

4-105. VOLTAGE FAULT. - A computer B Fault is incurred by the failure of power to the card equipment drive motor. This fault may be produced by faulty wiring or by blown fuses. The card equipment and the computer stop immediately. The card equipment needs the attention of maintenance personnel before it can be used.

4-106. OVERHEAT FAULT. - A computer A Fault is incurred by a temperature rise above 100^{o} F in any portion of the card unit control cabinet. If data transmissions to or from cards is in progress, this fault will probably generate a NO INFORMATION fault. The card equipment should be given the attention of maintenance personnel before it is again used.

4-107. The illumination of the STOP light on the card unit control cabinet indicates one or more of the following conditions.

a. The Stop button is depressed.

b. The read receiving stacker is full.

c. The punch receiving stacker is full.

d. The read feed hopper is empty.

e. The punch feed hopper is empty.

f. The Standby Switch is thrown to its forward position (away from the operator).

g. A Punch Jam has occurred. If a card jams under the edge of the punching die, as it enters Station 3 of the punching channel, the JAM and STOP indicators on the card unit control cabinet light, and the card equipment stops. Power to the card unit should be shut off immediately. Maintenance procedures are necessary to resume operation.

These conditions do not cause a computer fault stop; however, no more card unit activity is possible, and the computer may stall, until the condition is eliminated. All of these conditions, except (g), permit resumption of operation when the condition is removed. When the stop button is depressed as in (a), the Stop light is illuminated; when this button is released, the Stop condition is removed. For stops caused by (b) and (c), simply remove the cards from the filled stacker(s). For stops caused by (d) or (e), move the Standby Switch to the forward position during refill of input hoppers. (During the manual operation which clears the channels, the depression of the Clear button causes a bypass of the circuitry which normally causes an equipment stop when the feed hoppers are empty.) Note that during card reading, or card punching operations, the hopper not being used must contain at least one card to prevent the occurrence of the stop caused by (d) or (e) above. To remove condition (f), place the Standby Switch towards the operator.

4-108. PREPARATION FOR OPERATION. - The equipment is prepared for operation under program control after power is applied at the card unit control cabinet, and the following steps are carried out (see figure 4-23):

a. Step 1 - If only 72 card columns are to be read or punched, make sure that FIELD III switch is set to the OUT position. If 80 columns are to be read or punched, set this switch to NORMAL. (The Field III switch is located within the card unit control cabinet.)

b. Step 2 - If card reading is to be performed, place the deck of cards to be read into the right-hand feed hopper of the card unit cabinet. If card punching is to be performed, place a deck of cards in the left-hand feed hopper. The cards are placed for feeding face down so that row 9 is the first row processed in both punching and reading operations. Place the metal weights on top of the decks. At least one blank card should be placed in either hopper when the corresponding channel is not to be used in the program.

c. Step 3 - Set the toggle switches on the card unit panel in the following manner:

DUPL.	away from operator
PUNCH	away from operator
MOTOR	left
DC	left
READ	away from operator
PICK READ	towards operator
PICK PUNCH	towards operator
STANDBY	towards operator

After these steps have been performed, the punched card program may be started in the computer.

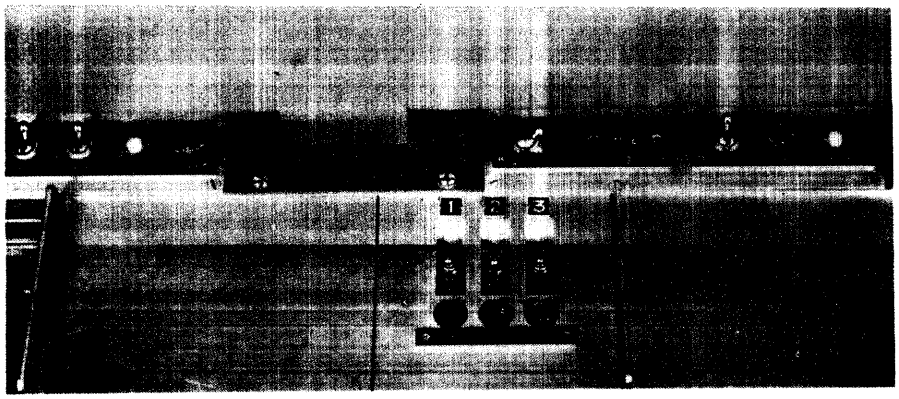


Figure 4-23

4-109. Certain operations of the punched card system may be undertaken by manual manipulation of switches and pushbuttons on the card unit panel. The procedures are as follows:

4-110. TO START. - If cards are in the feed hoppers and it is desired to drive the card equipment through one or more card cycles, press the START pushbutton. If the START button is pressed and released immediately, the card equipment will drive through one complete card cycle. If the button is held down, the equipment will run continuously.

4-111. TO PICK PUNCH CARDS. - If it is desired to feed cards into the punching channel, set toggle switch PICK PUNCH CARDS to the ON position (away from operator), then press and hold down the START pushbutton until the desired number of cards have fed.

4-112. TO PICK READ CARDS. - If it is desired to feed cards into the reading channel, set toggle switch PICK READ CARDS to the ON position (away from the operator), then press and hold down the START pushbutton.

4-113. TO STOP. - To stop the card equipment during a controlled run so that cards may be added to the feed hoppers, etc., either set toggle switch STANDBY to the ON position (away from the operator) or press and hold down the red STOP pushbutton. This causes the card equipment to stop at the end of the current card cycles so that the computer program is temporarily halted. If the toggle switch is reset, or the red STOP pushbutton is released, the system resumes normal operation.

4-114. To clear cards from either channel, remove the cards from the hopper, set the PICK READ CARD or PICK PUNCH CARD switch to its ON position if a card is in Station 1, and depress the CLEAR and START buttons until the channel is clear.

section 5

UNIVAC SCIENTIFIC MAGNETIC TAPE UNITS

5-1. GENERAL.

5-2. The magnetic tape sector of the Univac Scientific input-output section comprises a number of tape handling mechanisms and an electronic control section. Each tape unit, or Uniservo, handles and processes a metallic tape on which data can be stored as magnetized spots. The Uniservos are located external to the computer and are individually operated by the commonly shared magnetic tape control circuits internal to the computer. The number of tape units used is optional up to a maximum of ten functional units. By means of manual selections, the unit designations may be assigned in any manner to the functional units.

5-3. Magnetic tape data processing is under program control, in accordance with appropriately programmed External Function, External Read, and External Write instructions.

5-4. The Univac magnetic tape units are both an auxiliary memory for the Univac Scientific and an input-output equipment. As an auxiliary storage system, they provide a flexible, non-volatile memory of unlimited capacity for the computer. As input-output devices they give the Univac Scientific fast in-out data handling properties. Also, use of the Uniservos makes possible both on-line communication with the Univac series of computers, and off-line processing of information by a variety of Univac peripheral equipments. For input to the Univac Scientific, information may be recorded on tape according to three recording formats: fixed block length, variable block length, and continuous data input. A block of recorded information on the tape comprises a group of computer words. A block has unrecorded space preceding and following it. These spaces separate this block from other recorded blocks.

5-5. Output information from the Univac Scientific may be recorded on tape in fixed block length and variable block length. The fixed block length mode of operation is standard with the computer. Optional control circuits may be included to provide both the variable block length mode and the continuous data input mode.

5-6. The fixed block length mode reads and records information on the tape in blocks of fixed length. The variable block length mode reads and records information on tape in blocks of variable length. The length of the block is limited by the capacity of high speed storage. The continuous data input mode reads information recorded continuously on the tape with the only limitation on the length of a "block" of information being the length of the tape. This form of recording is useful for real time observations which will not permit interruptions to format the information in fixed or variable block lengths. Data input from tape recorded in this manner would need be interrupted when the capacity of high speed storage is reached.

5-7. UNITAPE.

5-8. The recording tape with the Uniservos is known as Unitape. The recording tape is a thin metallic strip of phosphor-bronze about 0.002 inch thick and 1/2 inch wide. The metallic tape is plated with a special nickel-cobalt film that can be magnetized. Information is stored on Unitape by magnetizing tiny spots on this film surface. Areas on the tape have either one of two polarities, representing either a "1" or a "0."

5-9. Figure 5-1 illustrates the physical arrangement of the spots on Unitape, and also the conventional representation of the arrangement. In figure 5-1, the darkened rectangles represent "1's"; the remaining area on the tape is polarized in the "0" direction.

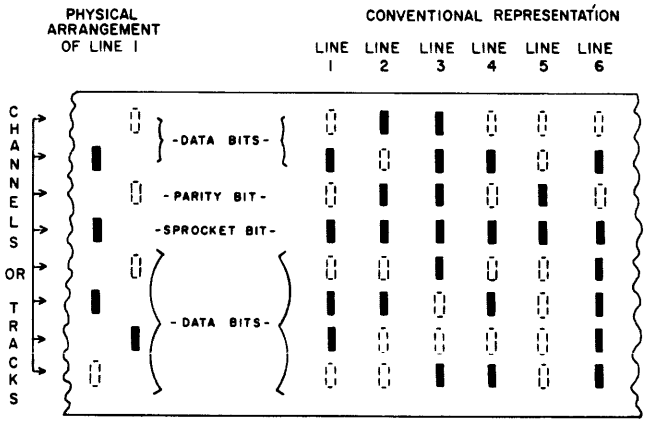


FIGURE 5-1.
UNITAPE TERMINOLOGY

5-10. Unitape terminology is discussed below.

Line: A line consists of the eight bit positions across the width of the tape, in which "1's" and "0's" can be represented.

Sprocket bit: The position of each line along the length of the tape is defined by its sprocket bit. The sprocket bit is recorded automatically with the recording of each line of data. Sprocket bits are always recorded as "1's." When a Unitape is read, the sprocket bits indicate that data (not a blank space) is being sensed. The sprocket bits also provide the basic timing for all reading as well as most positioning operations.

Parity bit: A parity bit or check bit is also recorded automatically with the recording of each line of data. Parity bits enable the equipment which reads the Unitape to check the accuracy with which it reads each line. The parity bit is a "1" if the six data bits are an even number of "1's"; and the parity bit is "0" if the six data bits are an odd number of "1's."

Hexabit character: The six data bits on each line are sometimes referred to as a hexabit character.

Pulse code: The combination of the parity bit and hexabit character is generally called a pulse code. The number of "1's" in each pulse code is always odd.

Tape density: The number of lines written per inch is the tape density. Unitapes processed by the Univac Scientific are recorded at 50 or 128 lines per inch. Unitapes are read and recorded one line at a time by all equipments that employ them.

5-11. All Unitapes are carefully tested, prior to their use, to insure perfect recording surfaces. Some areas of the tape may not be useable for recording: the noise level may be too high, the playback of recorded signals may be too weak, a splice may be present, etc. Holes are punched before, through, and after every imperfect area on Unitapes intended for fixed block and variable block length recording. These punched holes are detected, and the area between passed over, by the Univac Scientific and by all auxiliary Univac equipments on which the Unitape is used. Provision is made for unuseable tape areas in specifying the "length" of a Unitape. Approximately one foot of tape is provided for each error detected in testing. When a length of Unitape is specified, this is guaranteed recording length, free from imperfection. Four lengths are currently used: 1500' (standard with the Univac Scientific), 500', 200', and 100'. The latter two lengths, 200' and 100' are available as perfect tapes. Improper handling and prolonged use of the tapes will tend to develop new bad spots. Such bad spots, when detected, must be marked by punched holes.

5-12. Bad spot detection is not active when reading continuous data input Unitapes, since either perfect tapes are used or imperfections on the tape are not marked by punched holes. A Unitape with limited imperfections can be used for continuous data input in certain applications where many observations are recorded, and erroneous data can be "smoothed out" by the computer program. If it is essential, however, that perfect data be recorded during continuous data input recording, only perfect Unitapes should be used.

5-13. The unit of fixed block length Unitape data processed by Univac off-line peripheral equipment is the pulse code (combination of the hexabit character and parity bit). Each pulse code represents an individual alpha-numeric character, special symbol, or physical function of that equipment. All Univac off-line peripheral equipments either record or read and transcribe fixed block length Unitapes. No off-line equipments read or record variable block length Unitapes. The Univac I, and Univac II and Univac File Computer must be programmed to process magnetic tape.

5-14. UNIVAC SCIENTIFIC UNISERVO.

5-15. The Univac Scientific Uniservo is the tape handling device for reading, writing, and variously positioning Unitapes. All operations of the Uniservo can be controlled by the execution of appropriate computer instructions. Uniservo operations not involving data transmissions, such as rewinding and moving without reading or writing, can be manually controlled from the computer control panel, with the computer operating in the Test mode.

5-16. The following principal parts of the Uniservo are illustrated in Figure 5-2 and discussed below: read/write head, tape transport mechanism, erase head, and bad spot detectors. Operations indicators are discussed in the section on OPERATION.

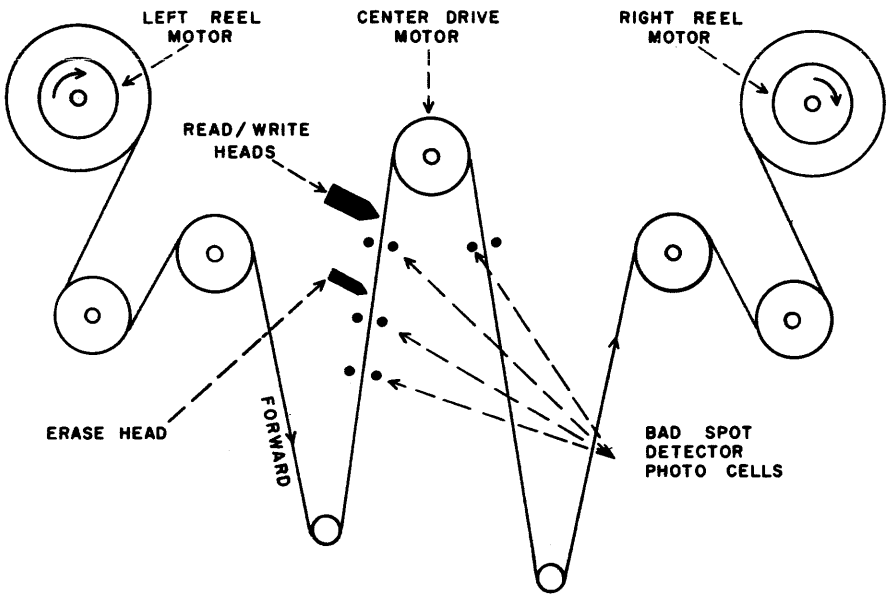


FIGURE 5-2. UNIVAC SCIENTIFIC UNISERVO.

5-17. TAPE TRANSPORT MECHANISM. - The tape-transport mechanism controls the tape drive and the movement of tape past the read/write head. Tapes can be moved either forward or backward as desired. Forward tape movement is defined as tape movement from the left reel to the right reel, with the right reel rotating in a clockwise direction.

The principal components of the tape transport mechanism are illustrated in Figure 5-2. They are as follows:

a. the left reel motor, center drive motor, and right reel motor: these motors effect tape drive.

b. two tape reels. In forward tape movement, the left reel is the supply reel and the right reel is the take-up reel. In backward movement the right reel is the supply reel; the left, the tape-up reel.

5-18. The tape transport mechanism accelerates tape from rest to 100 inches per second, standard tape speed in Univac Scientific Uniservos, in seven or less milliseconds, using approximately 0.25 inches of tape. It decelerates tape at the same rate. The center drive motor starts when magnetic tape control signals that tape movement is to start. The left and right reel motors are instrumental in adjusting the tension of the tape during tape starting and stopping, and tape movement. In general, when tape movement is started in a direction opposite to the previous movement, the adjustment by the tape transport mechanism for the reversal of direction causes a reversal delay. This reversal delay creates an interval between initiating a tape operation and actual reading or writing of the tape.

Figure 5-3 illustrates how Unitapes are attached to the two tape reels on each Uniservo.

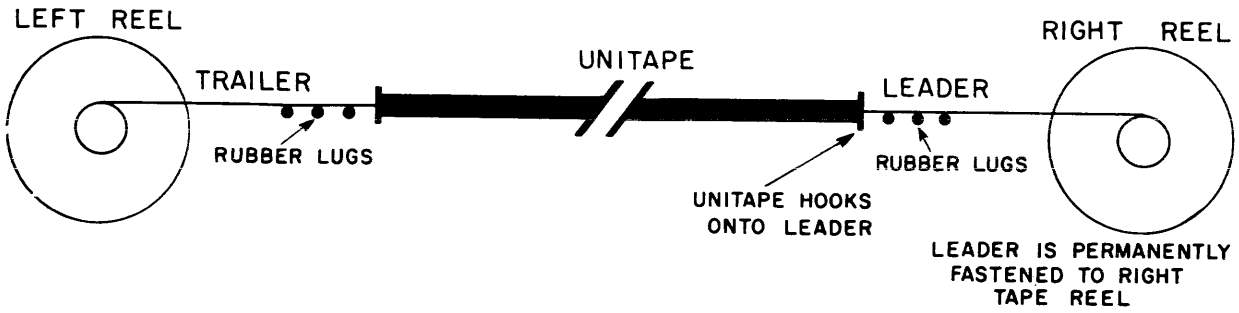


FIGURE 5-3. ATTACHMENT OF UNITAPES TO THE UNISERVO.

5-19. A reel of tape to be processed is manually mounted on the left reel motor. A leader of blank plastic tape, already threaded through the tape transport devices, and permanently fastened to the right reel, is then hooked to the free end of the Unitape on the left reel. A tape so positioned is said to be rewound. The first tape movement must therefore be in the forward direction. When the tape drive starts, the leader moves past the read/write head and onto the right reel. The first recorded block arrives at the read/write head no sooner than one second after the tape movement is initiated. The time required to move the tape from its rewound position up to the point where information is first recorded is known as the leader delay.

5-20. In order to prevent physical damage to Unitapes, the tape leader and tape trailer have rubber lugs attached. If a faulty program tries to move tape beyond either of its ends, the lugs trip a microswitch which automatically stops tape drive. An attempt to move onto the tape trailer results in a computer fault; an attempt to move onto the tape leader results in a computer stall.

5-21. ERASE HEAD. - In a writing operation, the tape is erased before new information is recorded on it. This function is performed by the erase head, which leaves the tape polarized in the zero direction. This erasure guarantees that the entire width of the tape is "clean." Since the erase head is located approximately four inches in advance of the read/write head, information that distance ahead of the read/write head is always erased. When a pre-recorded tape is rewritten, then, information in the early part of the block following the last block written is destroyed. Hence, writing operations should be started at the point where a last writing operation was stopped, or on a rewound Unitape.

5-22. READ/WRITE HEAD.

5-23. The read/write assembly consists of eight separate read/write heads. The strip of tape read or written by a read/write head is known as a channel. The exact vertical placement of each channel on the tape is established by each read/write head during recording. One head senses and records the parity bit channel; another senses and records the sprocket bit channel; and the remaining six, sense and record the six data bits channels. In reading operations, as each line on the tape passes beneath the read/write head, "1's" that are stored in the channels are sensed. The sprocket bit is always a "1"; when it is

sensed, any "1's" read from the six data channels and from the parity bit channel are simultaneously delivered to a register in magnetic tape control. The sprocket bit is sent to other control circuits which in general are concerned with checking bad spot detection and with timing the read operation. In writing operations, the channel beneath each read/write head is polarized in the zero direction unless a "1" is to be recorded. A "1" is always recorded in the sprocket channel, and each read/write head associated with the six data channels is pulsed if that channel is to store a "1." Also, depending on whether the number of "1's" recorded in the six data channels is even or odd, a parity bit of "1" is respectively recorded or not. The read/write head reads information in both forward and backward movement of the tape, but records information only in the forward direction. The read/write head is effectively inoperative in reading or writing operations while a bad-spot area is passing beneath it.

A plastic shim, or buffer tape, is inserted between the read/write head and the Unitape. This plastic tape serves to reduce friction and wear on both the tape and read/write head. (If this tape is broken or the supply is exhausted, tape movement stops automatically in the Uniservo.)

5-24. BAD SPOT DETECTION.

5-25. The bad spot detection circuits function during read, write, and move operations only when bad spot areas on the tape are marked by punched holes. These circuits enable the magnetic tape control circuits to disregard information received from unreliable tape areas. Holes punched in the tape at 2.5" intervals are detected by photoelectric cells which automatically interrupt reading, or writing operation until the bad area is passed over.

5-26. Bad spots are anticipated during writing so that the bad area never occurs between blocks. This ensures block spaces free of imperfections which could otherwise be picked up as extraneous signals during reading operations. In fixed block length mode operation, recording is interrupted for a bad area at any time except during the writing of the first or last two words of the block. In variable block length mode operation, recording is such that a bad area is kept between the third and fourth line of a word.

5-27. The bad spot detection circuits do not function in rewind operations, or in continuous data input operations if a tape with unmarked bad spots is being processed.

5-28. PROGRAMMED MAGNETIC TAPE PROCESSING

5-29. SELECTION OF A TAPE OPERATION

5-30. Three modes of tape operation are possible with the Univac Scientific magnetic tape system: fixed block length mode, variable block length mode, and continuous data input mode. Unitape for all three types of operation has lines arranged in groups called blocks. Blank spaces on the tape separate successive blocks. The number of lines in each block on fixed block length Unitapes is a constant, 720. Provision is made during recording to leave

spaces between blocks, i.e., inter-block spacing, of either 1.0 inch or 2.4 inches.* The number of lines in each block in variable block length and continuous data input modes is not necessarily constant. On variable block length Unitapes, only one block spacing of 1.4 inch is used. Various spacings (of 3 inches or less) can be employed to distinguish groups of data on continuous data input Unitapes. On fixed block length Unitapes only, blocks are subdivided into six blockettes, groups of 120 consecutive lines; blockettes may or may not be physically separated on the tape. When recording, spaces between blockettes, i.e., blockette spacings, of 0.1 inch, or 1.0 inch, are possible.

5-31. All tape processing by the Univac Scientific magnetic tape units must be programmed. Appropriate External Function instructions are programmed to start, stop, and completely define each tape operation. Coded information provided by the External Function instruction includes the following:

a. specification of optional recording mode, if option is available;

b. type of tape operation to be performed;

c. selection of a uniservo, if tape operation requires a selection;

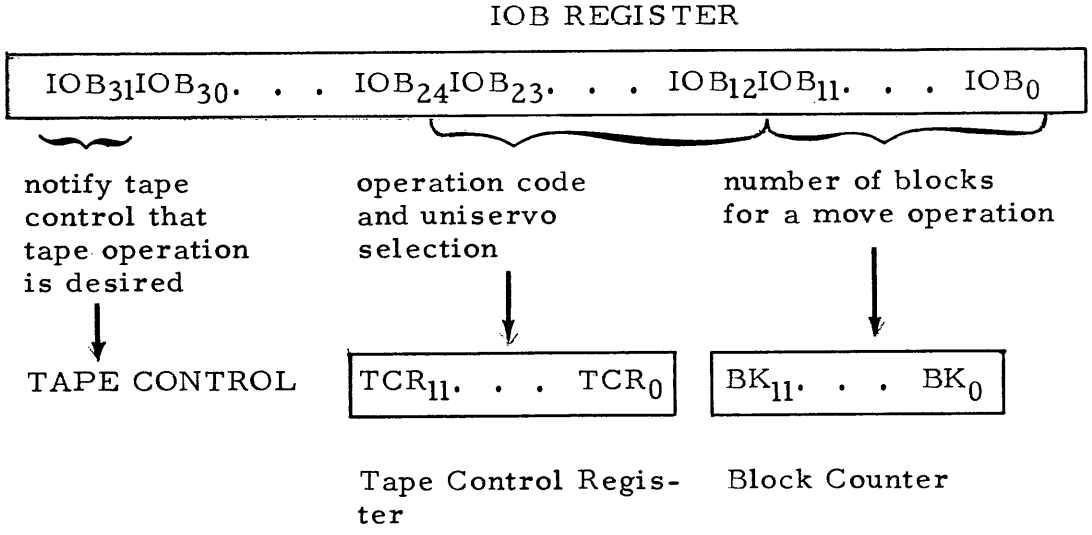
d. type of recording format for writing, i.e., inter-block and blockette spacing and recording pulse density (for fixed block length mode only);

e. number of blocks to be moved <u>without</u> reading or writing (not applicable in continuous data input mode).

5-32. The External Function instruction transfers the content of its v-address to the Input/Output Register, IOB. The 36-bit word thus introduced into IOB designates the magnetic tape operation to be performed. The "Select Magnetic Tape" bit, $IOB_{31}=1$, causes a transmission of the content of IOB to registers in the tape control system. The presence of the proper tape operation codes in the Tape Control Register, TCR, provides the control system with the information needed to carry on the particular tape operation designated. If the tape operation designated is a move operation, the number of blocks to be moved is transferred to the Block Counter, BK. The transmission from IOB to TCR and BK is shown in the following diagram.

---

* The inter-block space of 1.0 inch is nominal. If the computer is provided with the option of variable block length operation and continuous data input operation, the space left between blocks during a recording operation, in either fixed or variable modes, is approximately 1.4 inches. If the computer is not equipped for these optional recording modes, the space left between blocks during writing blocks of fixed length is approximately 1.2 inches. Tape recorded with any of the inter-block spaces, 1.0, 1.2, and 1.4 inches, can be read by the Univac Scientific. In this text, the space between blocks of fixed length is referred to as being one inch.



5-33. Following is a brief description of the different tape operations and their IOB selection bits. These operations are described in detail in the sections on the different modes of operation. The details should be well understood before a program for tape operation is prepared.

a. Read Forward - Read data from tape on the specified uniservo, assemble into 36-bit computer words, and transfer the words to IOB. Reading is possible in all modes of operation. The Read Forward bits are $IOB_{18,17,16,}$ =001.

b. Read Backward - Identical to read forward except that the tape is moved in the reverse direction. The bits of each computer word are assembled in the same order as in a read forward operation. The Read Backward bits are $IOB_{18,17,16,}$ =101.

c. Write Forward - Transfer words from IOB, and record on tape on the specified uniservo. Writing is possible in fixed and variable block length modes of operation. The recording format (density and spacing) can be specified in fixed block length writing. Bits $IOB_{18,17,16}$=011 specify writing at 128 lines per inch. Bits $IOB_{18,17,16}$=111 specify writing at 50 lines per inch. Bit IOB_{21}=1 specifies 2.4 inch interblock spacing. Bits $IOB_{20,19}$=01 specify 0.1 inch spacing between blockettes. Bits $IOB_{20,19}$=10 specify 1.0 inch spacing between blockettes.

d. Stop - Stop tape movement and tape reading or writing. Only interblock stops are possible in fixed block length mode. In variable block and continuous data input modes, intra-block stops are possible. The Stop bits are $IOB_{23,22}$=11.

e. Move Forward (n blocks) - Move tape forward n blocks on the specified uniservo (without a read or write operation), where $0 \leq n \leq 2^{12}-1$ Continuous data input mode has no move operation without simultaneous reading. The Move Forward bits are $IOB_{18,17,16}=010$. Bits $IOB_{11...0}$ specify the number n of blocks to be moved.

f. Move Backward (n blocks) - Identical to move forward except that the tape is moved in the reverse direction. The Move Backward bits are $IOB_{18,17,16}=110$.

g. Rewind - Rewind tape on the specified uniservo to the leader position. The mode of operation is of no consequence in either of the rewinding operations. The Rewind bits are $IOB_{23,22}=01$.

h. Rewind Interlock - Rewind tape on the specified uniservo to the leader position and provide an interlock which prevents further effective references to that tape unit until appropriate steps are taken to remove the interlock. The Rewind Interlock bits are $IOB_{23,22}=10$.

i. Change Bias - Change the read bias level to higher or lower than normal; or return bias level to normal if high or low bias level was chosen by a previous instruction. No uniservo specification is necessary or possible in the same instruction. The mode of operation is of no consequence in the change bias operation. Bits $IOB_{15...12}=1101$ specify normal read bias. Bits $IOB_{15...12}=1110$ specify low read bias. Bits $IOB_{15...12}=1111$ specify high read bias.

5-34. Except for the change bias and stop operations, a tape unit selection must be included in the EF code word for each of the above operations. Bits $IOB_{15...12}=0001, 0010, ...1010$ specify Uniservo 1, 2, ...10. A complete list of IOB code words to choose the above operations in the mode desired, is given in the tables on page 5-14.

5-35. SELECTION OF VARIABLE MODE, OR CONTINUOUS DATA INPUT MODE, OPERATION.

5-36. The tape operation code words inform magnetic tape control which type of the three Unitapes is to be processed. One of three modes of operation, fixed block length mode, variable block length mode, or continuous data input mode, is selected by appropriate programming; this selection is determined by the kind of Unitape to be processed.

5-37. If bits $IOB_{20,19}=11$ are included in a tape operation code, the specified operation (if it is legal) is in <u>either</u> the variable block length mode operation or continuous data input mode operation. These bits, the Select Variable/Continuous bits, cause operation in whichever mode was previously established for selection by one or more EF Change Mode instructions and/or a preceding computer Master Clear.

5-38. The EF Change Mode instruction places a "1" in IOB_{31} (magnetic tape master select bit) and bits "100" in $IOB_{18...16}$ (Change Mode bits.) The EF Change Mode instruction does not effect operation in any mode; it merely prepares tape control for the selection of the mode. Selection of the mode is by subsequent EF tape instructions with the Select Var/Cont bits. Which mode will be selected by such EF tape instructions depends upon the sequence of preceding Change Mode instructions and/or a Master Clear. The choice of either variable mode or continuous data input mode is determined as shown in the following diagram.

Master Clear of computer
- •
- • } Selection of variable mode by EF with Sel Var/Cont bits
- •

EF Change Mode instruction is executed
- •
- • } Selection of continuous data input mode by . . .
- •

EF Change Mode instruction is executed
- •
- • } Selection of variable mode by EF . . .
- •

EF Change Mode instruction is executed
- •
- • } Selection of continuous data input mode . . .
- •

etc.

5-39. As shown, successive EF Change Mode instructions alternate the choice of variable mode and continuous data input mode for subsequent selection. The series of EF Change Mode instructions and/or Master Clears need not be immediately consecutive. Intervening EF tape instructions with the Select Var/Cont bits provide operation in whichever mode was previously established for selection. (Intervening EF tape instructions without the Select Var/Cont bits cause operation in fixed block length mode operation.) An indicator labelled CONT INPUT in the MT Controls group on the computer control panel shows the preparation for continuous data input mode for subsequent selection. This indicator is illuminated by an EF Change Mode instruction, and extinguished by another EF Change Mode instruction, or by a computer Master Clear.

5-40. PROGRAMMING CONSIDERATIONS.

5-41. The number of lines in a properly recorded block, or group of data, is an integral multiple of six. An integral number of 36-bit computer words exists on a recorded tape. During recording or reading, an External Write or External Read instruction need be executed for each word. The External Write instruction transfers information from computer storage to the IOB register; the External Read instruction transfers information from the IOB register to computer storage. The data transfer link between IOB and the magnetic tape is the 36-bit Tape Register, TR, in the tape control section. A magnetic tape

writing operation involves the transmission of a word from IOB to TR, the break down in TR of the 36-bit word into six hexabit characters, the generation of a parity check bit for each character, and the transfer of each character, its parity check bit, and a sprocket bit to a line of tape. A reading operation involves the assembly of data from the tape in the Tape Register, a parity check on each line of tape as it is received in TR, and the transmission of a 36-bit word from TR to IOB.

5-42. The External Function instruction which initiates the reading or writing operation must be followed by an appropriate number of External Read or External Write instructions to transfer the information between the IOB register and the computer memory.

5-43. In reading operations, six data bits from six consecutive lines are automatically assembled into a 36 bit word by magnetic tape control before the 36 bit word is sent to IOB. Similarly, during recording operations, the 36 bit word sent to IOB by an External Write instruction is automatically disassembled by magnetic tape control, and then written six bits per line on six consecutive lines. The matter of interpretation of coded data and its storage, during reading or writing operations, must be taken care of by the program.

5-44. During writing operations, the six highest-order digits of the word in IOB are recorded on the first line of tape, the next six highest-order digits are recorded on the second line, etc. Tape recording is possible with the tape moving only in the forward direction. Reading operations, however, are possible with the tape moving either forward or backward. During read forward operations (involving fixed or variable block length Unitapes) the first line read is regarded as the highest order six bits, the second line as the next highest order six bits, etc. Similarly, during reading backward operations (involving fixed or variable block length Unitapes), the first line read is regarded as the lowest order six bits, etc. In the assembled 36-bit word sent to IOB, then, data from the first line read backward appears in IOB_{5-0}, data from the second line read appears in IOB_{11-6}, etc. The direction of tape movement during reading operations has no effect on the appearance in IOB of the word read.

5-45. Programming considerations differ according to the mode of tape operation. For example, parity checking, end of block detection, and end of record (end of information on the tape) detection differ in reading operations. In writing operations, fixed block length Unitapes must be recorded with the appropriate format for the equipment which will read and transcribe the data; the controlling computer program must keep the number of words written within the recording capacity of the length of Unitape being used, etc. Thus the requirements of the computer program is dependent upon which type of Unitape is to be processed.

5-46. By proper programming in variable block length mode operation, the Univac Scientific can be made to record pseudo-continuous data input Unitapes as well as pseudo-fixed block length Unitapes.

5-47. Variable block length Unitapes can be stopped after any word (six lines) is read or written. Fixed block length Unitapes can be stopped only in a blank space, (after an entire block is read or written). To reposition variable block length Unitapes after stopping within a block (as well as to reposition

fixed block length Unitapes after a stop), it is necessary to position the tape so that the blank space between blocks is under the read/write head. This would be inconvenient and time consuming in the case of continuous data input Unitapes, since a block may cover the length of the tape. Special code bits are therefore recorded periodically on continuous data input Unitapes, and positioning within a block is possible by detecting these code bits.

5-48. INITIATION OF A TAPE OPERATION

5-49. A tape operation cannot become effective until after the operation specification has been transmitted to the tape control section. The general procedures followed in determining if and when a tape operation is to be initiated are discussed subsequently. The following statements apply to all modes of tape operation only in so far as the particular type of tape operation being mentioned is possible in a particular mode.

5-50. The operation code specified by the External Function instruction is transferred from the X Register to IOB to the Tape Control Register, TCR, in the magnetic tape control section of the computer. The transfer of the code from X to IOB during the execution of the EF instruction may be delayed by the existence of an IOB lockout condition. After the lockout condition is removed by the external equipment, the operation code is sent to IOB and the output lockout condition is re-established. The tape operation code must then be transmitted to TCR in order for the tape control section to be notified of the desired operation. Simultaneous with the IOB to TCR (and IOB to BK) transmission, the IOB lockout condition is removed. Since this allows the loading of IOB by a second External Function instruction, the Tape Control Register is then protected against receiving another transmission from IOB until the first tape operation is completed, or until the status of the current tape operation warrants the removal of the protection. The Tape Control Register is protected, in general, against a transmission from IOB during the initiation of every tape operation; during a move and change bias operation; during the stop of tape movement when reading and writing is completed; and during a stop of tape movement due to a fault condition.

5-51. No protection is provided against the IOB to TCR transmission between blocks during reading or writing operations. If a transmission occurs at this particular time, the logical sum of IOB and the current content of the Tape Control Register is formed in TCR. This is not allowable except when an External Function instruction has been programmed to stop reading or writing operations. If a uniservo selection is not necessary for a particular tape operation, as is the case in a change of the reading bias, TCR is protected against another transmission until the change of bias is completed.

5-52. During an operation which moves a tape a specified number of blocks, the protection against another IOB to TCR transmission remains until the move is completed. (If a move operation is specified with the number of blocks to be moved quoted as zero, no protection on TCR is established and TCR is cleared. Another tape operation may be initiated immediately.)

5-53. For an operation which rewinds the tapes to their leader position, the protection against an IOB to TCR transmission is maintained until tape movement is under way, but released before it is completed. Thus, any number

of functional tape units may be rewinding tape concurrently.

5-54. After an IOB to TCR transmission, and before starting tape movement, several checks are made on the operation and uniservo specifications in TCR. This includes a check on the availability of the tape unit specified, if the designated operation requires a Uniservo selection. A Uniservo is given a logical designation by appropriately setting its selection switch located in the Tape Control Cabinet. (Switch selections are explained fully in the OPERATIONS section.) Uniservo "j" (as specified in $IOB_{15...12}$) is not available under any of the following conditions.

a. No Uniservo designation has been made in the operation code, i.e., $IOB_{15...12}=0$.

b. Uniservo j doesn't exist; i.e., the j designation exceeds the number of installed tape units and hence is not allowable, or none of the physical units have been designated as j. (This would include the case where the unit assigned the number j is out of use for maintenance purposes.) When a correct designation is made, Uniservo j becomes available.

   This condition precludes the case where two tape units are given the same designation. In this case, the first attempted IOB to TCR transmission is blocked until the switch settings are corrected.

c. Uniservo j is rewinding when referenced. Uniservo j is available, however, when rewinding is completed, if it is not interlocked at completion.

5-55. If Uniservo j is not available for any of the above reasons, tape movement is not started. No protection exists against another IOB to TCR transmission if the operation code currently in TCR specifies a read, write, or one of the rewind operations. However, if the operation code in TCR specifies a read or write operation, the execution of External Read or External Write instructions creates a computer lockout stall. If the operation code currently in TCR specifies a move operation on uniservo j with uniservo j not available, TCR is protected against an IOB to TCR transmission. An IOB lockout condition will result from the attempt to execute External instructions. If the operation code currently in TCR specifies a rewind operation, it is unlikely that Uniservo j is not available because it is currently rewinding or rewound with interlock. In the case where a rewind operation is specified with no Uniservo selection, or no assignment of j to a physical unit, another IOB to TCR transmission can occur. This effect of such a transmission is unpredictable since a logical sum is formed in TCR.

5-56. A check is also made before operation initiation for the readiness of uniservo j. The "not ready" conditions are discussed in the OPERATIONS section.

5-57. Several time delays are incurred before tape movement is started, and before actual reading or writing is possible. These delays result from the time consumed by the control functions mentioned above, and also such factors as the direction of tape movement, current positioning of the tape, interblock

spacing requirements, etc. Some of the above-mentioned times that are applicable and of concern to the programmer, and are possible to determine, are given in the tables on pages 5-28 and 5-43.

5-58. IOB CODE SELECTIONS FOR MAGNETIC TAPE OPERATION

FIXED BLOCK LENGTH MODE

FUNCTION	DIRECTION OF TAPE MOVEMENT	NUMBER OF BLOCKS	BLOCKETTE SPACING (inches)	BLOCK SPACING (inches)	(v) OF EF-v (in octal)
Write 128 lines/inch	Forward	Free Run	0.0	1.0	02 00006* u0000
				2.4	02 00106* u0000
			0.1	1.0	02 00026* u0000
				2.4	02 00126* u0000
			1.0	1.0	02 00046* u0000
				2.4	02 00146* u0000
Write 128 lines/inch	Forward	One Block	Same as above	Same as above	Add to each (v) above: 00 00600 00000
Write 50 lines/inch	Forward	Free Run	0.0	1.0	02 00016* u0000
				2.4	02 00116* u0000
			0.1	1.0	02 00036* u0000
				2.4	02 00136* u0000
			1.0	1.0	02 00056* u0000
				2.4	02 00156* u0000
Write 50 lines/inch	Forward	One Block	Same as above	Same as above	Add to each (v) above; 00 00600 00000
Read	Forward	Free Run	-	-	02 00002* u0000
		One Block	-	-	02 00602* u0000
	Backward	Free Run	-	-	02 00012* u0000
		One Block	-	- -	02 00612* u0000
Move n blocks	Forward	n blocks =bbbb	-	-	02 00004* ubbbb
	Backward	n blocks =bbbb	-	-	02 00014* ubbbb

*Four binary digits are reserved for the uniservo selection. In the table, three bits are represented by the octal digit u, and the fourth is the rightmost bit of the octal digit marked by an asterisk, *.

VARIABLE BLOCK LENGTH MODE

FUNCTION	DIRECTION OF TAPE MOVEMENT	NUMBER OF BLOCKS	BLOCKETTE SPACING (inches)	BLOCK SPACING (inches)	(v) OF EF -v (in octal)
Write 128 lines/inch	Forward	One**	0.0	1.0	02 00066* u0000
Read	Forward	-	-	-	02 00062* u0000
	Backward	-	-	-	02 00072* u0000
Move n blocks	Forward	n blocks = bbbb	-	-	02 00064* ubbbb
	Backward	n blocks = bbbb	-	-	02 00074* ubbbb

**Writing is continued until an EF Stop Tape Instruction is executed. The space created by the Stop defines the block.

CONTINUOUS DATA INPUT MODE

Read	Forward	-	-	-	02 00062* u0000
	Backward	-	-	-	02 00072* u0000

PROVIDE VARIABLE OR CONTINUOUS MODE FOR SUBSEQUENT SELECTION

Change Mode	-	-	-	-	02 00010 00000

*Four binary digits are reserved for the Uniservo selection. In the table, three bits are represented by the octal digit u, and the fourth is the right-most bit of the octal digit marked by an asterisk, *.

ALL MODES

FUNCTION	DIRECTION OF TAPE MOVEMENT	NUMBER OF BLOCKS	BLOCKETTE SPACING (inches)	BLOCK SPACING (inches)	(v) OF EF-v (in octal)
Stop Read or Write	-		-	-	02 00600 00000
Rewind	Backward	to leader	-	-	02 00200* u0000
Rewind Interlock	Backward	to leader	-	-	02 00400* u0000
Change Bias to Low	None	-	-	-	02 00001 60000
Change Bias to High	None	-	-	-	02 00001 70000
Change Bias to Normal	None	-	-	-	02 00001 50000

5-59. FIXED BLOCK LENGTH MODE OPERATION

5-60. FORMAT

5-61. The following arrangement of data is common to all fixed block length Unitapes.

a. Every line contains a sprocket bit, a parity bit, and six data bits.

b. A block consists of 720 consecutive lines; blocks are separated by blank spaces on the tape.

c. Blocks are subdivided into six blockettes of 120 consecutive lines each; blockettes may or may not be separated by blank spaces on the tape.

Since all data transmissions between the magnetic tape and the Univac Scientific are 36-bit transmissions from six lines on the tape, a blockette on a fixed block length Unitape can be regarded as holding 20 computer words; a block, 120 words.

*Four binary digits are reserved for the Uniservo selection. In the table, three bits are represented by the octal digit u, and the fourth is the right-most bit of the octal digit marked by an asterisk, *.

5-62. The Univac Scientific can read all Univac fixed block length formats. The fixed block length formats possible in recording tapes with the Univac Scientific are shown in figure 5-4.

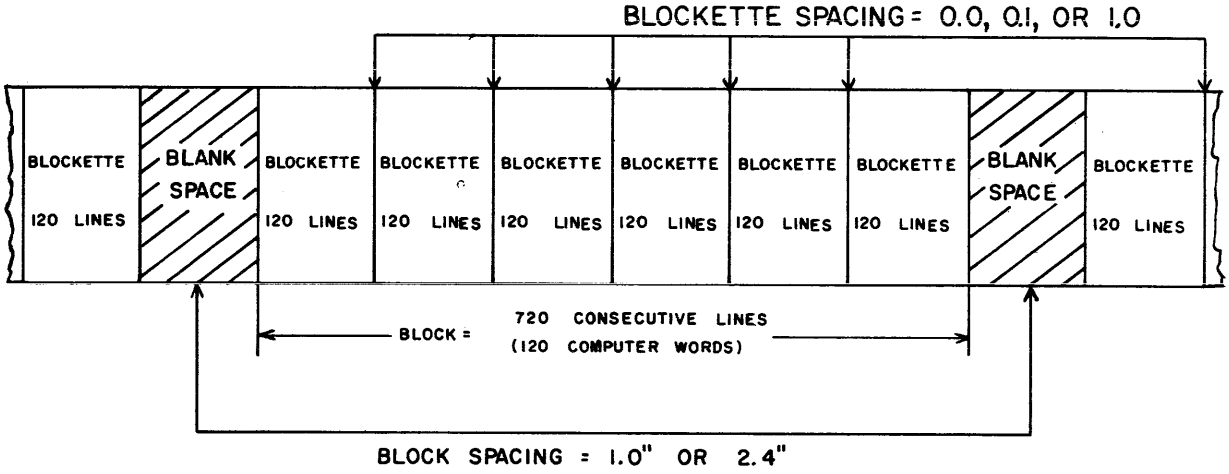


FIGURE 5-4. UNIVAC SCIENTIFIC FIXED BLOCK LENGTH UNITAPE RECORDING FORMATS.

5-63. The number of fixed blocks which can be recorded on a 1500 foot reel of magnetic tape is approximately 2700 (assuming 128 lines per inch, 1.0 inch block spacing, 0.0 blockette spacing). Special data identification information can be stored in any block. Sentinel blocks can be used to locate related sets of data on the tape, or the beginning and the end of recorded data on the tape. Data identification information need only to be coded in such a way that the program can differentiate it from other data.

5-64. PROGRAMMING AND TIMING

5-65. READ FORWARD OPERATION. - The EF Read Forward instruction, followed by External Read (ERjv) instructions, enables the designated tape unit to read blocks of data from the tape. The read forward operation must be terminated by an EF Stop Tape instruction programmed to follow the ER used to read the last word of the final block. However, if it is desired to read only one block, the stop code may be included in the EF Read Forward instruction which precedes the ER instructions.

5-66. A successful read operation is dependent principally on two things: proper program format, and observance of time limitations. (Time limitations are given in the table on page 5-28.) Initially, an EF Read Forward instruction containing the information mentioned in the above paragraph must be programmed. Following this EF instruction, there must be 120 ERjv (j=1) instructions programmed for each block of tape that is to be read. Following the reading of each block, a programmed check must be made to determine whether the information just stored has been read correctly from the tape, i.e., to determine whether a parity error occurred during the reading of the block. The "parity check" requires that an ERjv (j=0) instruction be programmed immediately following the last ER instruction for the block. The parity check is discussed in detail on page 5-18.

5-67. If a parity error is detected anytime during the reading of a block, tape movement is stopped automatically at the end of the block. A check is also made during reading to determine whether the proper number of lines have been recorded in the block. If a greater than, or less than, 720 line count is detected, an automatic tape stop occurs at the end of the block in error, and a Sprocket Error Fault is indicated. The sprocket error is discussed in detail on page 5-19.

5-68. Assuming the absence of both the parity and sprocket errors, another block of tape may be read; or if the desired number has already been read, the EF Stop Tape instruction should be programmed. This instruction must contain nothing but the magnetic tape master selection bit and the stop code bits. It must never contain Uniservo selection bits.

5-69. TIMING

5-70. Timing is of utmost importance in a successful read operation. Between the EF Read Forward instruction and the first ER instruction, there is a basic delay. This delay is the interval between starting the tape and transmitting an assembled word from the Tape Register to the IOB register. The basic delay may be increased by a leader delay (see page 5-5) and/or reversal delay (see page 5-4). If an ER instruction is attempted during the delay interval, the computer stalls and waits until the delay time has elapsed. Hence, this time may be used for other computations that do not involve the IOB register. The ER instruction should be executed at the end of the interval so that the computer is ready to read the word in IOB as soon as the TR to IOB transmission occurs.

5-71. If the execution of the ER instruction is so late that a second TR to IOB transmission occurs before IOB has been read, an IOB Read Fault results. This fault is discussed in detail on page 5-20.

5-72. A limited amount of time is available between successive External Read instructions. Again, an IOB Read Fault results when an ER is programmed so late that a second TR to IOB transmission occurs before IOB has been read and cleared of the first transmission. If more than one block is to be read, time is available for other than tape operations after the execution of the 120 ER's of IOB and the ER of IOA (for parity error). The times available are those necessary to cross the one inch or 2.4 inch spaces between blocks. An IOB Read Fault results when an ER is programmed later than the time allowed for crossing the particular block space. When the last block has been read, an EF Stop Tape instruction must be executed (assuming there is neither a parity nor a sprocket error) within a certain time.

5-73. PARITY ERROR

5-74. As each line of data is recorded on magnetic tape, a parity bit is generated and recorded for that line. If the number of "1's" in the six bit character is odd, the parity bit recorded is a "0." If the number of "1's" in the charac-

ter is even, the parity bit recorded is a "1." This always results in an odd sum if the character and parity bits are added together. By means of this parity bit, the accuracy of data transfer is checked as each frame of tape is read into the computer.

5-75. If a parity error occurs, a "1" is set into IOA_0; a stop of tape movement is effected at the end of the block in which the error occurred; and the tape control registers are cleared when the stop is effected. Thus, IOA must be read and checked at the end of each block read. This is accomplished by programming an ERjv (j=0) instruction immediately following the last ER instruction for the block. An ERjv (j=0) instruction reads the content of IOA and stores it at the v-addressed location of the instruction. The occurrence of a parity error can then be determined by checking the contents of the v-addressed location. If a parity error is detected, the block can be re-read in the opposite direction, and re-read at different bias levels. An attempt to re-read correctly at normal bias level should be tried before the bias level is changed. If none of these passes effect a correct reading, a computer stop can be programmed to indicate the unsuccessful attempt to read the block correctly. Before reading is resumed, a return to normal bias level should be effected. If a parity error occurs in the last of a group of blocks being read, the EF Stop Tape instruction should not be executed: since the tape has already been stopped, and the tape control registers have been cleared, the stop code in TCR would contain no tape unit designation. In this case, the lockout on IOB would be removed, but another IOB to TCR transmission would be blocked, and the computer would eventually stall. No actual computer fault would result.

5-76. FAULT CONDITIONS

5-77. SPROCKET ERROR. The sprocket bit, which is recorded automatically with the recording of each line of tape, provides not only the basic timing for reading and positioning operations, but also indicates to tape control that data is being read. In a read operation, a sprocket error means that either more than 720 lines, or less than 720 lines, have been counted in any particular block. A sprocket error is indicated by the illumination of the MT, B Fault, and Sprocket Error indicators on the control panel.

5-78. A sprocket error effects a computer fault stop immediately after the detection of the fault at the end of the block. The tape control registers are cleared when the error is detected. Tape movement is stopped at the end of the block with the improper number of lines. The stop occurs such that the position of the read/write head in the inter-block space is the same as after a normal stop. Resumption of reading is possible after the fault condition is cleared by a computer Master Clear.

5-79. A < 720 Sprocket Error is the result of the detection of a block space before a count of 720 lines has been accumulated. In a reading operation, a less than 720 count generally means that although the proper number of ER's have been programmed, less than 120 ER instructions are actually executed. Since not enough lines are left in the block to form the last word in TR, there is no TR to IOB transmission. Thus, the ER attempted at the end of the block prevents the execution of further instructions until the computer stop. This lockout condition is indicated by an ER instruction in the Program Control Register, PCR.

5-80. A > 720 Sprocket Error indicates that at least one sprocket signal is detected after the 720 line count has been accumulated, and before the block spacing is detected. The computer stop is immediate upon detection of the first extra sprocket signal. Sprocket signals recorded at 128 lines per inch density are sensed at 78 microsecond intervals. It is unlikely that additional External Read instructions or an External Function Stop Tape instruction will be executed during this short an interval before the computer stop.

5-81. IOB READ FAULT. An IOB Read Fault is incurred by the tardy execution of an External Read instruction, or may be caused by programming an insufficient number of ER instructions for reading a block of tape. This fault is the result of the following circumstances: Each word IOB receives from TR should be removed by an ER instruction before the next TR to IOB transmission occurs. If a second transmission occurs before IOB is cleared, an IOB Class I Computer B Fault is incurred. The computer is stopped immediately at the time of the second transmission. Tape movement is stopped, and the tape registers are cleared, at the end of the block in which the fault is detected.

5-82. The IOB Read Fault is indicated by illumination of the IOB computer fault indicator and the IOB Class I fault indicator.

5-83. The read error may not be detected, or may be detected but not immediately, if the erroneous program has one less than necessary External Read instructions (119 ER's, fixed block mode). If, or when, the error is detected depends upon whether the block being read is an intermediary or last of a series of blocks, or a single block. The effect of these circumstances is explained in the following two paragraphs.

5-84. The programming of one less than necessary External Read instructions for reading either a single block, or the last block of a series, does not directly cause a computer fault. The last six lines of the block are transmitted to IOB. Since the last External Read instruction is missing, IOB is not read and cleared before the next EF instruction is executed. Thus, a logical sum is formed in IOB and transmitted to TCR, making it impossible to predict the next tape operation. Depending upon this logical sum, a "false" tape operation may or may not be initiated.

5-85. If one less than necessary External Read instructions are programmed for the reading of any block but the last of a series, the IO fault could occur at the beginning of the next block, or could occur at the end of the last block in the series. The time at which the fault will occur is dependent upon the use of the computation time across the block space. If the first ER is executed before the first TR to IOB transmission for the block, the fault will go undetected until the end of the last block being read.

5-86. READ BACKWARD OPERATION.

5-87. The read backward operation is similar to the read forward operation except the tape movement is in the opposite direction. See the read forward operation for a detailed discussion.

5-88. WRITE OPERATION.

5-89. Following the execution of the EF Write instruction, 120 External Write instructions (EWjv; j=1) must be executed for each block to be written. When the desired number of blocks have been written, an EF Stop Tape instruction must be executed to stop the tape movement and clear all registers in the tape control system. If only one block is to be written, the stop bits may be included in (v) of the External Function instruction which initiates the writing operation. Then, tape movement is stopped automatically at the end of one block.

5-90. A certain amount of time is available for other computation between each successive instruction used for the writing operation. Computation done during this time must not exceed the timing requirements of the tape system. For a listing of available times, see the table on page 5-29.

5-91. FAULTS

5-92. A number of programming errors may lead to computer faults or operation delays, during a write operation. Some of the most common of these errors are as follows:

(1) Too few or too many External Write instructions programmed for the number of blocks to be written.

(2) Failure to stop the tape after the writing is completed.

(3) Use of too much time for other computation between successive tape instructions.

5-93. NO INFORMATION FAULT: The No Information fault occurs when the tape control system attempts to transcribe a word on the tape before the word to be written has been received from the computer. The computer is stopped immediately upon detection of this condition. Zeros are written on the tape until tape movement is stopped. The tape is stopped at the end of the block in which the fault is detected, and registers in the tape control system are cleared. At the time of the stop, the MT, B fault and No Information fault indicators on the computer control panel will be illuminated.

5-94. TOO FEW EXTERNAL WRITE INSTRUCTIONS: The programming of too few External Write instructions always results in a No Information fault. Detection of the erroneous program is not always immediate, however, as explained in the following paragraphs.

5-95. If too few External Write instructions are programmed for any but the last of a series of blocks, and none of the computing time available between blocks is used, the first few EW instructions for the next block are interpreted as the "missing" EW's and the fault is not detected until the last block. If the computing time between blocks is used, however, the fault is detected in the block for which too few EW's have been programmed, and the tape movement is stopped at the end of this block. If too few EW instructions are programmed for the last of a series of blocks, or if the stop code is included in the EF Write instruction (for one block), tape movement is always stopped at the end of the block being written.

5-96. TOO MANY EXTERNAL WRITE INSTRUCTIONS: In most cases the programming of too many External Write instructions also causes a No Information fault. The one exception occurs when too many EW instructions follow a "Write one block and stop" operation. In this case no fault occurs, but the execution of the first extra EW produces an IOB lockout. If an attempt is made to execute a second extra EW, operation is stalled. If only one extra EW is programmed, computer operation continues until an attempt is made to execute another EF instruction. At the time of the stop, IOB contains the word to be written by the first extra EW, and PCR contains either an EW or an EF instruction.

5-97. If too many EW instructions are programmed for any of a series of n blocks to be written, the extra EW's are interpreted as the first few EW's for the next block. Any additional EW instructions continue to be executed. After the nth block has been written the remaining EW instructions are executed, and writing of the (n + 1st) block is begun. Since not enough EW's are available to write this block, the No Information fault occurs as the result of "too few" EW instructions.

5-98. FAILURE TO STOP TAPE AFTER WRITING: Failure to stop tape movement after a writing operations also results in a No Information fault. Tape movement continues until a block space is produced, after which tape control attempts to write the "next" block. Since no EW instructions are executed for this block the computer is stopped by the fault condition, as is the tape movement.

5-99. EXCEEDING TIMING REQUIREMENTS: A No Information fault also occurs when timing requirements are exceeded. If the time between the EF instruction and the first EW, or between successive EW instructions is exceeded, the computer is stopped, and zeros are written on the tape. When the end of the block is reached, tape movement is stopped.

5-100. STOP TAPE OPERATION.

5-101. The stop tape operation effects a stop of tape movement after the desired number of lines have been written or read. The EF Stop Tape instruction should never contain Uniservo selection bits.

5-102. At the completion of a read or write operation, the stop code is transmitted from IOB to TCR. Since TCR has not been cleared, the stop code is added (logical sum) to the current contents of TCR. A tape stop is then initiated during which TCR is cleared. At the completion of a 'read/write one block stop' operation, the procedure is similar except that the stop code already exists in TCR. The time allowable between the last ER or EW and the EF Stop instruction is given in the tables on page 5-29. The restrictions governing the figures for allowable times are given in the footnotes for the tables.

5-103. MOVE FORWARD OPERATION.

5-104. During the initiation of the move forward operation, the number n of blocks to be moved is transferred to the Block Counter, BK. During tape movement, each successive line is counted. After 720 lines have been counted, the quantity one is subtracted from the Block Counter. If less than 720 lines are detected for a block, no error occurs; a false 720 line count is propagated and the move continues. When the Block Counter has been counted down to zero, the tape is stopped and the registers in the tape control system are cleared.

5-105. A move forward of n blocks when $m<n$ have been recorded results in a computer fault. When an attempt is made to move more blocks than have been recorded, tape movement does not stop until the trailer at the end of the tape is reached. This produces a Uniservo Interlock fault. This fault is indicated by illumination of the Uniservo Interlock indicator, and the MT B Fault indicator. If there is no further reference to IOB, computation continues until the fault stop is effective. If the illegal move operation is followed by two or more tape instructions (or any two or more instructions involving IOB), the computer may be stalled before the interlock fault stop becomes effective. The (v) of the first EF instruction executed following the move instruction remains in IOB. Computation continues until another instruction involving IOB is executed. This second instruction remains in PCR and stalls the computer.

5-106. SPROCKET ERROR DURING MOVE OPERATION

5-107. As soon as more than 720 lines are counted for any block, a sprocket error occurs. Tape movement is stopped at the end of the block in which the

error occurs, and the registers in the tape control system are cleared. A sprocket error effects a computer fault stop immediately after the detection of the fault at the end of the block. At the time of the fault stop, the MT, B fault, and Sprocket Error indicators on the computer control panel are illuminated. Resumption of operation is not possible until after a computer Master Clear.

5-108. MOVE BACKWARD.

5-109. A move backward is executed in the same manner as a move forward, the only difference being in the direction of the tape movement. The same select bits in (v) of EF-v are used, with Move Backward selection bits replacing the Move Forward selection bits.

5-110. A move backward of n blocks when m<n have been recorded produces an IOB lockout if another EF instruction is executed. The tape is rewound and stopped on the leader. Computation is continued until two instructions involving IOB are attempted. The first instruction involving IOB produces an IOB lockout. Computation continues until the second instruction involving IOB is executed. Then, since IOB contains the (v) of the previous EF, the second EF remains in PCR and stalls the computer.

5-111. REWIND.

5-112. The rewind operation causes the tape on the specified Uniservo to be rewound to its leader position. Two or more Uniservos may be operating at the same time if one or more of the Uniservos is rewinding.

5-113. Once the rewind operation is through its initiation phase, another tape operation may be started immediately on another Uniservo. Until the rewind initiation is completed, however, the code word for the next operation cannot be transmitted from IOB to TCR. An External Function instruction executed during this interval creates an IOB lockout condition. The time consumed by the rewind initiation phase depends upon the direction of the previous tape operation on the Uniservo. The rewind initiation phase may incur a reversal delay, according to the following circumstances: the tape unit is placed in a rewound condition until a tape clear or computer Master Clear is effected. Until the rewound condition on a tape unit is removed, a reversal delay (see page 5-4) is effected by the initiation of tape movement in a forward direction.

5-114. If the tape operation following the rewind requires the Uniservo currently being rewound, the operation is delayed until the rewinding is completed. The fact that Uniservo j is not currently available for operation is detected after the code word is in TCR. If the operation being delayed is a read or write operation, the execution of External Read or External Write instructions creates an IOB lockout condition. If the operation being delayed is a move operation, the next External Function creates an IOB lockout condition. If the operation being delayed is another rewind operation (on the same tape unit, which is very unlikely), no protection is provided against another IOB to TCR transmission.

5-115. REWIND INTERLOCK.

5-116. The discussion of rewinding given previously for the rewind operation applies also to the rewind interlock operation. The difference between the two operations is that a tape rewound with interlock cannot be referenced effectively for a tape operation until the Uniservo door interlock switch has been opened and closed. This occurs when the Uniservo is provided with another tape.

5-117. Referencing a Uniservo whose tape has been rewound with interlock before the Uniservo door interlock switch has been opened and closed causes a "not ready" condition. After eliminating the "not ready" condition, computation can be resumed. Recovery from the "not ready" condition is discussed in the OPERATION section.

5-118. CHANGE BIAS OPERATION.

5-119. The change in the bias level needs to be programmed only when the incorrect reading operation has occurred. A re-read at normal bias should be attempted first. If this is not successful, reading forward and backward at the high and low bias levels may accomplish a correct reading of the block. The depression of the computer Master Clear button accomplishes a return to the normal reading bias level. If a computer master clear does not occur, the return to the normal reading bias level must be programmed. For optimum reading performance, the return to normal reading bias should be accomplished before normal reading is resumed.

5-120. Until the bias change is completed, TCR is protected against another IOB to TCR transmission. An External Function instruction executed during this interval creates an IOB lockout condition. A second EF instruction attempted stalls the computer until the change is completed and the protection is removed.

5-121. If an instruction to change to high bias (low bias) is programmed at such a time when high (low) bias level already exists, no change occurs, the protection is not removed, and consequently the computer stalls. It is not permissable to program any other tape operation in the instruction which specified a change in bias level.

5-122. SYNOPSIS OF FIXED BLOCK LENGTH MODE OPERATION

TAPE OPERATION	PROGRAMMING AIDS
Read Forward/ Backward (pages 5-17 to 5-21)	Designate in (v) of EF instruction the following: magnetic tape master selection, read forward/backward, tape unit. See table page 5-14. Program 120 ER's for each block to be read. Observe limitations on allowable computation time between the EF Start instruction and first ER, between successive ER's, across blockette spaces, across block spaces, and between the last ER and EF Stop Tape instruction. See table page 5-28. Read IOA and check its contents for parity error after each block has been read. Terminate successful read operations with an EF Stop Tape instruction. Omit Uniservo selection in an EF Stop Tape instruction. Read and check IOA for the last block before programming an EF Stop Tape instruction. If a parity error has occurred, do not execute EF Stop Tape. Program a read forward or backward of no more than the number of recorded blocks. See tables page 5-52 for fault diagnosis and text references.
Write (pages 5-21 to 5-23)	Designate in (v) of EF instruction the following: magnetic tape master selection, write, block spacing, blockette spacing, recording density, tape unit. See table page 5-14. Program 120 EW's for each block to be written. Observe limitations on allowable computation time between successive EW's, across blockette spaces, across block spaces, between last EW and EF Stop Tape. See table page 5-29. Terminate the write operation with an EF Stop Tape instruction. Omit Uniservo selection in EF Stop Tape instruction.

Stop (page 5-23)	See tables page 5-52 for fault diagnosis and text references. Designate in (v) of EF instruction the following: magnetic tape master selection and stop code. See table page 5-16. Omit Uniservo selection in stop code. Terminate write, read forward, and read backward operations with an EF Stop Tape instruction.
Move Forward/ Backward (pages 5-23 to 5-24)	Designate in (v) of EF instruction the following: magnetic tape master selection, move forward/backward, number of blocks to be moved, tape unit. See table page 5-14. Program a move forward or backward of no more than the number of recorded blocks. Do not program an EF Stop Tape instruction following a move operation. See tables page 5-52 for fault diagnosis and text references.
Rewind (page 5-24)	Designate in (v) of EF instruction the following: magnetic tape master selection, rewind, tape unit. See table page 5-16.
Rewind Interlock (page 5-25)	Designate in (v) of EF instruction the following: magnetic tape master selection, rewind interlock, tape unit. See table page 5-16. Before referencing a Uniservo rewound with interlock, provide the unit with another tape, or at least, open and close the Uniservo door interlock switch.
Change Bias (page 5-25)	Designate in (v) of EF instruction the following: magnetic tape master selection, and bias level. See table page 5-16. Exclude any other tape operation from the EF Change Bias instruction. Return to normal bias level to resume normal reading, as soon as possible. Do not program two successive changes to high or low bias level

5-123. AVAILABLE COMPUTATION TIMES

5-124. Fixed Block Length Mode

5-125. The following table shows the recommended maximum times allowable for computation between tape operations. The safe times listed were based on theoretical timing conditions and then adjusted to include a safety factor. The safety factor adjustment takes into consideration fluctuations in normal operating characteristics. The manufacturing specifications for the equipment and the standards used in maintaining the equipment determined what degree of adjustment was necessary. Execution times for any instructions effecting the particular tape operation under consideration have not been subtracted from the stated times in the table. (Reference is made to the footnoted items in paragraphs 5-126 through 5-131.)

Situation	Safe Time
READ OPERATION	
Between EF Start Tape and first ER[1]	
1. 2" block space	38 ms
2. 4" block space	38 ms
(See also leader and reversal delays)	
Between successive ER instructions	
at 128 lines per inch	420 μs
at 50 lines per inch	1080 μs
Across the blockette space	
1. 2" blockette space	10 ms
0. 1" blockette space	0. 9 ms
Across the block space	
1. 2" block space	10 ms
2. 4" block space	21 ms
Between last ER and EF Stop Tape[2]	
1. 2" block space	1 ms
2. 4" block space	12. 5 ms

Situation	Safe Time
WRITE OPERATION	
Between EF Start Tape and first EW[3]	
1. 2" block space	38 ms
2. 4" block space	38 ms
(See also leader delay)	
Between successive EW instructions	
at 128 lines per inch	420 μs
at 50 lines per inch	1080 μs
Across the blockette space	
1. 2" blockette space	10 ms
0. 1" blockette space	0. 9 ms
Across the block space	
1. 2" block space	10 ms
2. 4" block space	21 ms
Between last EW and EF Stop Tape[3]	
1. 2" block space	1 ms
2. 4" block space	1 ms
REVERSAL DELAYS--movement in opposite direction. Tape <u>not</u> previously rewound.	
Between EF Start Tape and first ER[4]	
1. 2" block space	550 ms
2. 4" block space	550 ms
LEADER DELAY--starting reading/writing on rewound tape.	
Between EF Start Tape and first ER/EW	
1. 2" block space	1391 ms
If <u>unit</u> is in rewound condition[5]	1931 ms
2. 4" block space	1430 ms
If <u>unit</u> is in rewound condition[5]	1943 ms

The following times are consumed by the tape control circuitry before the next IOB to TCR transmission is allowed. In these cases, neither a plus nor a minus variation from the theoretical time is harmful in terms of a fault occurring. Since any variation is as likely to occur in one direction as the other, the theoretical times are quoted without a safety factor adjustment.

Situation	Theoretical Time
REWIND INITIATION DELAYS	
Previous movement in backward direction	70 ms
Previous movement in forward direction	635 ms
Tape unit in rewound condition[6]	70 ms
CHANGE BIAS DELAY	35 ms

5-126.[1] These times are based on the safe times quoted for stopping. See note 2.

5-127.[2] These times allow safe resumption of reading in either direction. The quoted time for 1.2 inch block spacing stops the tape close enough to the middle of the block space to allow tape acceleration to its free-running speed before the first line of the next (following or preceding) block is encountered. The quoted time for the 2.4 inch block space allows the same distance for tape acceleration (when reading is resumed in the same direction).

Fixed stopping time after the EF Stop is executed allows the tape to move through approximately half of the 1.2 inch block spacing, consuming theoretically 8.75 milliseconds. Hence, if the EF Stop immediately follows the last ER, the tape stops in the middle of the 1.2 inch block space (and the same distance into a 2.4 inch block space).

If reading is either (1) not to be resumed, or (2) to be resumed in the opposite direction, the maximum computation time between an EF Stop and the last ER is that time which allows movement of the tape up to the first line of the following block. If a read forward operation is followed by a write operation, the time between the last ER and the EF Stop Tape should not exceed one millisecond for either 1.2 or 2.4 inch block spacing. See note 3.

5-128.[3] The time between an EF Start instruction and writing the first line of the next block is fixed by internal timing conditions. The fixed time provides for the traversal of the portion of the (1.2 or 2.4 inch) block space remaining after the previous stop, assuming that the EF Stop instruction immediately followed the last EW (or ER). To maintain reasonably consistent block spacings, no more than one millisecond should be programmed between the last EW and the EF Stop if writing

is to be resumed. Theoretically, tape movement need not be stopped until the entire block space is traversed. However, if more than the block space is traversed before stopping, tape control interprets the operation as the intent to write the next block, and consequently issues a No Information fault.

5-129.[4] The time over 600 milliseconds required for a reversal operation depends upon how soon tape movement was stopped at the conclusion of the last operation. The times given assume that the EF Stop for the preceding read operation immediately followed the last ER. Based on the safe times quoted in the table for stopping a read operation, the reversal delay for 2.4 inch block spacing would be increased nine milliseconds; allowing one millisecond between the EF Stop and the last ER, for either 1.2 or 2.4 inch block spacing, does not appreciably change the reversal delay time quoted in the table.

5-130.[5] A tape unit on which a tape has been previously rewound (or rewound with interlock) is said to be in a rewound condition until a tape Clear or computer Master Clear is effected. Until this time, any attempted forward movement of the rewound tape or a replacement tape incurs the basic theoretical reversal delay of 600 ms.

5-131.[6] A rewind initiation of a tape on a unit which has undergone a previous rewind operation (with the same tape or a replacement tape) consumes the "previous backward direction" delay if the tape unit is still in a rewound condition (See footnote 5.) If an interim tape Clear or computer Master Clear has occurred, the basic reversal delay or "previous forward direction" delay in this case, is incurred. Regardless of the delay, no tape movement is started, and another tape operation can be initiated after the delay time.

5-132. VARIABLE BLOCK LENGTH MODE OPERATION

5-133. FORMAT

5-134. On variable block length Unitapes, each line of recording contains a sprocket bit, a parity bit, and six data bits. Lines are grouped in blocks of a variable number of words. The number of lines per block must be an integral multiple of six. Only full 36 bit words are transmitted between the computer and the magnetic tape. A block can thus vary in length from one computer word up to the capacity of rapid access storage. Variable block length Unitapes are recorded only at 128 lines/inch, 1.4 inch inter-block spacing, and no spacings within the block. The general format for variable block length Unitapes is shown in figure 5-5.

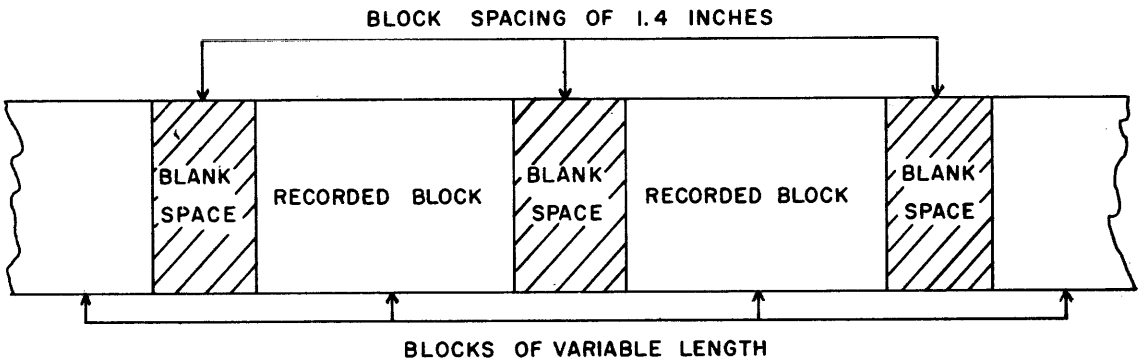


FIGURE 5-5. FORMAT FOR VARIABLE BLOCK LENGTH UNITAPES.

5-135. In the variable block length reading and moving operations, a block is recognized by the magnetic tape control circuitry if no sprocket bits, i.e., recorded lines, are detected for a distance of 0.06 inches. During reading operations this lack of recording is indicated by sending appropriate data to the IOA register. Also during reading, the "end of block" detection includes a check on whether the number of lines recorded in the block was an integral number of six. The end of block detection during moving operations is used to reduce the block count of the number of blocks to be moved.

5-136. The end of variable block length recording on the tape is detected by magnetic tape control when no sprocket bits, i.e., recorded lines, are detected during reading for a distance of approximately 4.0 inches. At this time, tape movement is automatically stopped and appropriate data is sent to the IOA register. Obviously the end of a block is always detected and signaled prior to the end of recorded data. Special coded data for <u>each</u> of these two conditions is sent to IOA, as is the indication of a parity error check on <u>each</u> word. The program must therefore examine closely the content of IOA. The method of signaling IOA to differentiate blocks and note the end of recorded data might be used in place of recorded sentinel blocks or other special identification data.

5-137. Variable block length Unitapes with 720 lines recorded per block have a pseudo fixed block length format: they are identical to fixed block length Unitapes with 0.0 blockette spacing and nominal one inch block spacing. True variable-length blocks are intended to be read and recorded only by Univac Scientific magnetic tape units

5-138. PROGRAMMING AND TIMING

5-139. READ FORWARD OPERATION

5-140. The EF Read Forward instruction with Var/Cont bits initiates reading of blocks containing variable numbers of words. Similar to fixed block operation, an External Read. ERjv (j=1) instruction must be programmed for each word to be read, with reading terminated by an External Function (EF-v) Stop Tape instruction.

5-141. Due to differences basic to the variable block mode, reading is accomplished somewhat differently than is reading in fixed block mode operation. Since the number of words per block is variable and relatively unrestricted, a parity check is made on each word as it is read. When, and only when, six lines have been assembled in the Tape Register, TR, the 36-bit word is transmitted to IOB and a parity check indication is sent to IOA. Thus, IOA must be read for each word and its contents checked. There is no automatic halt of tape movement when a parity error is detected. Therefore, a parity error may be ignored and tape reading continued. The parity check is discussed in more detail later.

5-142. The IOA Register is also used to receive indications of the end of a recorded block on the tape, and the end of recording on the tape. A lack of sprocket pulses for approximately 600 us may indicate a block space. If so, an end of block signal is generated, and a mod 6 check is made. The mod 6 check determines whether the block contains an integral number of 36-bit computer words; i. e., the number of recorded lines must be a multiple of six. If there was no mod 6 error, the end of block signal sets "10" in $IOA_{1,0}$. If a mod 6 error exists, the end of block signal sets "11" in $IOA_{1,0}$. A mod 6 error does not cause an automatic stop of tape movement. IOA must be read, and its contents tested, to determine the occurrence of a mod 6 error. The error may be ignored, and the tape program continued. The mod 6 error is discussed in more detail later.

5-143. If, during reading forward or backward, the tape control circuitry detects a lack of sprocket pulses for a distance of approximately four inches, an end of record signal is sent to IOA. The assumption is made that no more recorded information exists on the tape. A "1" is sent to IOA_2, and the tape movement is stopped automatically. During recording, the responsibility of knowing when to stop the writing, thus providing a unrecorded area of at least four inches on the far end of the tape, is left to the programmer. If this is done, tape movement onto the tape trailer should never occur during reading. The end of record indication sent to IOA prevents the programming error of "Read Forward n blocks when m<n blocks are recorded."

5-144. An end of record space between the tape leader and the first recorded block always exists because of the length of tape passed when writing is started with the tape rewound to the leader. This eliminates the possibility of the error, "read backward n blocks when m<n blocks are recorded."

5-145. The External Read of IOA must be the first External instruction to follow the External Function Read instruction. A check of the contents of IOA then determines the course of action. If $IOA_{2,1,0}$ is "000" or "001," then a word has been transmitted to IOB, and an External Read of IOB must follow. If the bits are "001," a parity error occurred during the reading of the word in IOB. Reading may be continued, or an EF Stop Tape instruction can be

programmed (after IOB is read). Either course of action is possible regardless of whether or not a parity error occurred. If $IOA_{2,1,0}$ is "010" or "011," the end of the block was detected, and an External Read of IOB should not be programmed. If the bits are "011," a mod 6 error exists in the block just read. Reading may be continued, or an EF Stop Tape instruction can be programmed. However, if reading is continued after a mod 6 error without a tape stop between blocks, the programmer is ignoring the incorrect assembly of words in the last block and the incorrect assembly of words in the next block. (This is discussed more fully in the paragraphs following on the mod 6 error.) If the External Read and the check of the contents of IOA reveals that $IOA_{2,1,0}$ is "100," the end of the recorded area on the tape has been detected. Tape movement is automatically stopped, and the program for reading this tape should be discontinued.

5-146. It was noted in the last paragraph that tape operation can be halted after reading any word within the block, as well as at the end of the block. The programmer may then re-position the tape at the beginning (end) of the block by a move backward (forward) of one block. Reading should not be attempted after an intra-block stop without first repositioning the tape. A correct reading operation, forward or backward, from an intra-block stop is almost impossible. A mod 6 error is most likely to occur with out-of-order assembly of the words read. Also, a parity error is almost certain to occur. A correct reading operation must be initiated from the beginning of the block (for reading forward) or from the end of the block (for reading backward). However, a move operation, either forward or backward, can be performed correctly after an intra-block stop. No indication of either a mod 6 error or a parity error are sent to IOA during moving operations. A stop between blocks is effected by an EF Stop Tape instruction after detection of the "end of block" bits in IOA.

5-147. A pattern for the order in which External instructions should be executed during a read operation is shown in the diagram on page 5-36. This diagram is amplified by the discussion following on parity error and mod 6 error.

5-148. ERRORS DURING READING

5-149. PARITY ERROR. The occurrence of a parity error during a read operation is evidenced by sending a "1" to IOA_0 at the same time a TR to IOB transmission occurs. The parity error exists in one or more of the six lines which were assembled in the Tape Register. If the nature of the data being read is such that one incorrectly read word will not alter the results of the computation, the parity error may be ignored, and the read operation continued without interruption. The parity error does not effect a computer fault or stop, nor a halt of tape movement. An attempt can be made to read the word correctly. If this is desired, an EF Stop Tape instruction should be programmed to halt tape movement immediately after the detection of the parity error. The tape may then be repositioned at the beginning of the block containing the parity error by a move backward operation of one block. The first re-read of the block should be at the normal bias level. If this attempt to read correctly fails, a

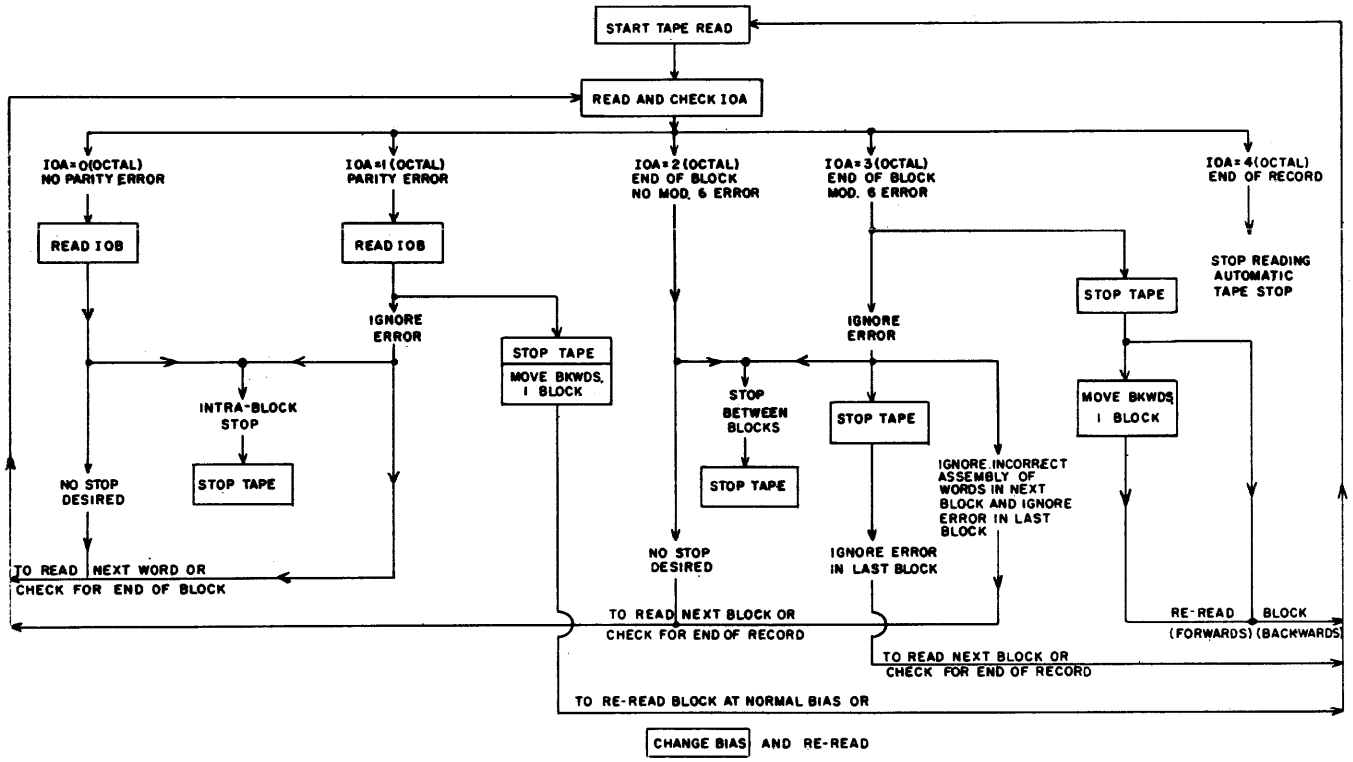
change in bias level may be programmed (after a move backwards of one block), and the block may be re-read again in the forward moving direction. If the change to high and/or low bias does not produce a correct reading, an EF Stop Tape instruction may be programmed. A change to normal bias level should be made as soon as possible for resumption or continuance of reading.

5-150. MOD 6 ERROR. The end of block indication in $IOA_{1,0}$ is "10" if the block was read without a mod 6 error. The end of block and mod 6 error indication in $IOA_{1,0}$ is "11." A check of the contents of IOA detects either condition. As is the case of the parity error, the mod 6 error may be ignored if so desired; neither a computer fault nor a halt of tape movement is caused by a mod 6 error. The tape operation can be stopped and corrective measures taken. Ignoring the error implies that the programmer is not concerned if words read from the tape are incorrectly assembled. Any incorrectly assembled words from the block just read are ignored if tape reading in the same direction is resumed after an EF Stop Tape instruction. If tape operation is not stopped after a mod 6 error, words from the next block to be read are also incorrectly assembled. The reason for this "mis-assembly" of words is discussed in the following paragraph.

5-151. A mod 6 error indicates that the number of lines per block was not an integral multiple of six, i.e., less than six lines were read from the tape for at least one of the 36-bit words supposedly recorded. If, for example, the last six bits of the second recorded word are missed during reading, the first line of the third recorded word is assembled in the Tape Register as part of the second word. Then, since six lines have been assembled in TR, a TR to IOB transmission occurs, and an External Read places this mis-assembled word in storage. From this point to the end of the block, all words which are placed in storage are assembled out of order. If one line is missed in the block, only five lines are assembled in the Tape Register for the last word in the block. The mod 6 check which occurs when inter-block spacing is detected, finds that less than 6 lines were assembled for the last word and notifies IOA of this fact by placing a "11" in $IOA_{1,0}$. The incomplete word remains in the Tape Register until TR is cleared. Since the mod 6 error does not cause a computer fault, nor a halt of tape movement, the error presumably can be ignored. (However, IOA must be read.) If another block has been recorded, and the reading operation is continued, the programmer is ignoring the mis-assembly of words in the last block from the incorrectly read word to the end of the block. Also, it must be noted that all the words of the next block will be assembled out of order, and the mod 6 error will reoccur at the end of this block. The incorrect assembly of words continues until the Tape Register is cleared at the end of reading a block. If an EF Stop Tape instruction is executed after the mod 6 error is detected by reading IOA, the next block can be read correctly. (TR is cleared automatically by a stop tape operation.) An EF Stop Tape instruction for this inter-block Stop can also be followed by an EF Read Backward instruction, or EF Move Backward and EF Read Forward instructions, in an attempt to re-read the block correctly.

5-152. IOB READ FAULT. An IOB Read fault is incurred if too few External Reads have been programmed to read a block, or if an External Read is programmed too late. The IOB Read Fault is discussed on page 5-20 under fixed block length mode operation, and is listed in the table on page 5-52. Although an IOB Read Fault could be caused by programming too few ER's while operating in variable mode, it is more likely that the fault will be the result of too much time elapsing between successive ER instructions. If this is the case, the program must be corrected before any successful reading can be accomplished.

5-153. OTHER FAULTS. Another less common programming error which may lead indirectly to a Read fault is the failure to execute an EF Stop Tape instruction at the completion of reading an intermediate block (not at the end of the record). This error is listed in the table on page 5-52.



INSTRUCTIONS ARE SHOWN ENCLOSED IN BOXES – THE NECESSARY "HOUSEKEEPING" INSTRUCTIONS ARE NOT GIVEN HERE.

FIGURE 5-6. PROGRAMMING FOR VARIABLE READING.

5-154. READ BACKWARD.

5-155. Except for the change in direction, the read backward operation is similar to the read forward operation. Parity check, end of block and mod 6 check, and end of record indications are all sent to IOA.

5-156. WRITE OPERATION.

5-157. No block spacing or writing density need be specified in the code word for the variable block writing operation. Writing in this mode is automatically carried out at 128 lines per inch density and 1.4 inch block spacing. If bits specifying other density or block spacing are contained in the word transferred to the Tape Control Register, these bits are ignored.

5-158. The block written may be as short as one word or as long as the number of words in rapid access storage. An External Write instruction is executed for each word to be written. After each word is written, the Tape Control Register, TCR, is inspected for stop code bits; if stop bits are present the tape is stopped, producing a block space, and the registers in the tape control system are cleared. If TCR does not contain stop bits, the next word to be written is transferred from IOB to TR. No blockette or block spaces are produced automatically during a variable mode writing operation; the only unrecorded areas existing on the tape are the block spaces produced by stopping the tape, and the bad spot areas skipped during the writing process.

5-159. BAD SPOT DETECTION, WRITING AND READING.

5-160. A bad spot on the tape is marked by punched holes preceding and following the unreliable recording area. A bad spot on the tape is detected in advance so that the bad spot area never occurs in the space between blocks. If a bad spot is anticipated before or during writing the third line of the current word, writing is interrupted so that three lines of the word are transcribed preceding (and three lines, following) the bad spot. If a bad spot is anticipated after writing the third line of the current word, all six lines of this word are recorded without interruption. The bad spot area is spanned by the lines of a future word with three lines recorded before and after the bad area. A bad spot may be anticipated after recording the third line of the word which later proves to be the last word of the block; i.e., an EF Stop Tape instruction is programmed to stop writing after the current word. In this case, the bad spot area interrupts writing of the first word of the next block. A bad spot detection after the third line of the last word, or during the stopping time between blocks, is effectively ignored until writing of another block is started. The starting time needed to traverse the 1.4 inch space between blocks insures that the bad spot is anticipated again in time to interrupt writing the first word of the next block.

5-161. Since writing after bad spot detection is not interrupted until a line count of three is reached, the same line count of three is reached during reading up to a bad spot area. This assumes that no mod 6 error occurred in the block previous to the bad spot. Reading after a bad spot is detected is continued past the last recorded lines preceding a bad spot area, up to a distance of one inch of the punched holes marking the bad area. However, since a number of inches of tape are left erased and unrecorded preceding and following the bad spot area, no data is picked up from the tape and assembled in the Tape Register. A move operation past a bad spot area is similar to a read operation except that no line count is kept and no lines are actually assembled in the Tape Register.

5-162. It is possible for the unrecorded area preceding the actual bad area on the tape to be interpreted, during reading only, as an end-of-block space. The end of block signal to IOA depends upon two conditions: (1) the detection of an 0.06 inch unrecorded area preceding, by at least one inch, any punched holes marking a bad spot area, and (2) a line count not equal to three. When a bad spot area is detected during reading a block recorded in variable mode operation, at least an 0.06 inch unrecorded area is detected, but the line count is three if no mod six error has occurred. If, however, a mod six error has occurred, and from one to five lines have been missed in the past block, the line count is 2, 1, 0, 5, or 4, and the unrecorded area between the last recorded line and the inch preceding the marked bad spot, is detected as an end of block space. The end of block signal is sent to IOA, as is a mod 6 error indication (unless the line count is zero). Tape movement is stopped in the bad space by a programmed stop for a mod six error. However, since the unrecorded area is long enough to be interpreted as an end of record area, an automatic tape stop always occurs. In this case tape movement is halted in the bad area.

5-163. During recording in fixed block length mode operation, the bad spot area can be left between words, or between any lines of a word, with the following exceptions: The bad spot area is never left between the last word of a block and the first word of the next block, i.e., between blocks; between the last two words of a block, or between any lines of these words; or between the first two words of a block, or between any lines of these words. When during variable mode operation, a bad spot area is detected during reading a block which was recorded in fixed block mode operation, the unrecorded area preceding the bad area is interpreted as an end of block space if the line count is 0, 1, 2, 4, or 5. Only if the line count is 3 is the bad area not detected as an end of block space. With a line count not equal to three, the end of block indications for a bad spot area result as described in the preceding paragraph.

5-164. At any time a legitimate block space is detected and a mod 6 error has occurred such that the line count is three, neither an end of block nor mod 6 error indication is sent to IOA. Note that a read backward (forward) of one block after an intra-block stop cannot be depended upon to effect an end of block signal to IOA in the block space preceding (following) the block. A mod 6 error is highly probable during such a reading operation, and this mod 6 error might result in a line count of three at the end of the block. If this is the case, no programmed detection of the end of block is possible and the EF Stop Tape instruction planned to stop tape movement is not executed.

5-165. ERRORS DURING WRITING.

5-166. Faults occurring during a variable mode writing operation are generally due to two types of programming errors:

(1) Failure to observe timing requirements.

(2) Failure to stop the tape when the writing has been completed.

Both of these errors lead to No Information Faults. The No Information Fault occurs when the tape control system attempts to write a word before the word to be written has been received from the computer. In variable block mode, the tape is stopped immediately upon detection of the fault and the registers in the tape control system are cleared. The computer is also stopped immediately upon detection of the fault. At the time of the stop, the MT and B-fault indicators on the computer control panel, and the No Information Fault indicator are illuminated.

5-167. If the available time between External Write instructions has been exceeded, the computer is stopped immediately. Note that to avoid a No Information fault, an EF Stop Tape instruction to stop the writing procedure cannot be executed later than the available computation time between External Writes.

5-168. If the No Information fault occurs because no EF Stop instruction is programmed to conclude a writing operation, the computer is stopped upon detection that the available time between External Write instructions has been exceeded. Tape movement is also stopped at the time.

5-169. STOP TAPE OPERATION.

5-170. An EF Stop Tape instruction is needed to complete a successful write, read backward, and read forward tape operation. During a reading operation, both inter-block and intra-block stops are possible. During a writing operation, the stop produces the inter-block space. The time allowable between the last ER or EW and the EF Stop instruction is given in the tables on page 5-43. The restrictions governing the figures for allowable times are given in the footnotes for the tables.

5-171. To stop reading in the middle of a block, an EF Stop Tape instruction is programmed after IOB is read. After an intra-block stop, reading can be resumed correctly at the beginning of this block (after a move backward one block), or at the beginning of the next block (after a move forward one block).

5-172. MOVE FORWARD OPERATION.

5-173. The move operation need not be terminated by an EF Stop Tape instruction. The move continues until the number of blocks specified have been moved. At that time, the tape is stopped and the registers in the tape control system are cleared. In order to safely program move operations, an accurate record should be kept of the number of blocks on the tape and the current position of the tape.

5-174. The IOB to TCR transmission of an EF move instruction releases the lockout on IOB. At this time, another EF tape instruction may be executed. Another IOB to TCR transmission is not possible, however, until the move operation is completed, and the protection on TCR is released.

5-175. The end of block detection, caused by the space between blocks, is used to count down, on the block counter (BK), the number of blocks moved. No end of block or end of record signal is sent to IOA. Neither is there a parity nor a mod 6 check made during a Move operation. Consequently, nothing is loaded into IOA, and the register need not be read or checked. An EF Move Forward instruction, specifying a move of one block and programmed after an intra-block stop, effects a move with no error indications into the space after the block in which the stop was made.

The computer condition resulting from erroneously programming a move forward of n blocks when m (<n) are recorded is discussed on page 5-23 and is listed in the table on page 5-52.

5-176. MOVE BACKWARD.

5-177. The move backward operation functions the same as the move forward operation except for the tape movement in the opposite direction. An EF Move Backward instruction, specifying a move of one block and programmed after an intra-block stop, effects a move with no error indications into the space preceding the block in which the stop was made.

5-178. The computer condition resulting from erroneously programming a move backward of n blocks when m (<n) are recorded is discussed on page 5-24 and is listed in the table on page 5-52.

5-179. REWIND, REWIND INTERLOCK, CHANGE BIAS.

5-180. The variable/continuous selection bits have no significance for the rewind, rewind interlock, and change bias operations. Hence, these select bits need not be included in the EF instructions which initiate these operations. Each of these operations is discussed in detail in the section for fixed block length mode operation.

5-181. SYNOPSIS OF VARIABLE BLOCK LENGTH MODE OPERATION

TAPE OPERATION	PROGRAMMING AIDS
Change Mode (page 5-10)	If tape control is not already prepared for variable block length mode operation, progran an EF Change Mode instruction preceding a read, write, or move operation. Designate in (v) of EF instruction the following: magnetic tape master selection and change mode. See table page 5-15.
Read Forward/ Backward (pages 5-32 to 5-36)	Designate in (v) of EF instruction the following: magnetic tape master selection, read forward/backward, tape unit, var/cont. See table page 5-15. Before reading a word, read and check contents of IOA; then, accordingly, read IOA again, or read IOB and/or program an EF Stop Tape instruction. See diagram page 5-36. Observe limitations on allowable computation time between the EF Start instruction and the first ER, between successive ER's and between the last ER and EF Stop Tape instruction. See table page 5-43. Omit uniservo selection in EF Stop Tape instruction. See tables page 5-52 for fault diagnosis and text references.
Write (pages 5-37 to 5-39)	Designate in (v) of EF instruction the following: magnetic tape master selection, write, tape unit, var/cont. See table page 5-15. Program an EW instruction for each word to be written. Observe limitation on allowable computation time between the EF Start and first EW, between successive EW's, and between the last EW and EF Stop Tape instruction. See table page 5-44. Program an EF Stop Tape instruction immediately after the last word to be written in each block. Omit uniservo selection in EF Stop Tape instruction. See tables page 5-52 for fault diagnosis and text references.

TAPE OPERATION	PROGRAMMING AIDS
Stop (page 5-39)	Designate in (v) of EF instruction the following: magnetic tape master selection and stop code. See table page 5-16. Observe requirements for reading IOB and/or IOA before programming an EF Stop Tape instruction to stop reading. See diagram page 5-36. Program an EF Stop Tape during reading for all desired stops, <u>except</u> at end of record; during writing, to terminate tape movement and create the block spacing. <u>Omit</u> uniservo selection in the stop code.
Move Forward/ Backward (pages 5-39 to 5-40)	Designate in (v) of EF instruction the following: magnetic tape master selection, move forward/backward, var/cont, number of blocks to be moved, tape unit. See table page 5-15. Program a move forward or backward of <u>no more</u> than the number of blocks that are recorded in the direction of movement. Do <u>not</u> read and check IOA; this register is not loaded during a move operation. Do <u>not</u> program an EF Stop Tape instruction following a move operation. See tables page 5-52 for fault diagnosis and text references.
Rewind	See text page 5-40. See table for Fixed Block Length Mode Operation, page 5-27.
Rewind Interlock	
Change Bias	

5-182. AVAILABLE COMPUTATION TIMES

5-183. Variable Block Length Mode

5-184. The following table shows the recommended maximum times allowable for computation between tape operations. The safe times listed were based on theoretical timing conditions and then adjusted to include a safety factor. The safety factor adjustment takes into consideration fluctuations in normal operating characteristics. The manufacturing specifications for the equipment and the standards used in maintaining the equipment determined what degree of adjustment was necessary. Execution times for any instructions effecting the particular tape operation under consideration have not been subtracted from the stated times in the table. (Reference is made to the footnoted items in paragraphs 5-185 through 5-191.)

Situation	Safe Time
READ OPERATION	
Between EF Start Tape and first ER[1] (1.4" block space)	41 ms
(See also leader and reversal delays)	
Between successive ER instructions (at 128 lines per inch)	420 μs
Across the block space (1.4" block space)	12 ms
Between last ER and EF Stop Tape	
Inter-block stop[2]	1 ms
Intra-block stop[3]	250 μs

Situation	Safe Time
WRITE OPERATION	
Between EF Start Tape and first EW (1.4" block space) (See also leader delay)	41 ms
Between successive EW instructions (at 128 lines per inch)	420 μs
Across the block space (1.4" block space)	12 ms
Between last EW and EF Stop Tape[4] (1.4" block space)	250 μs
REVERSAL DELAY--movement in opposite direction. Tape not previously rewound.	
Between EF Start Tape and first ER[5]	550 ms
LEADER DELAY--starting reading/ writing on rewound tape.	
Between EF Start Tape and first ER/EW If unit is in rewound condition[6]	1396 ms 1936 ms

The following times are consumed by the tape control circuitry before the next IOB to TCR transmission is allowed. In these cases, neither a plus nor a minus variation from the theoretical time is harmful in terms of a fault occurring. Since any variation is as likely to occur in one direction as the other, the theoretical times are quoted without a safety factor adjustment.

Situation	Theoretical Time
REWIND INITIATION DELAYS	
Previous movement in backward direction Previous movement in forward direction Tape unit in rewound condition[7]	70 ms 635 ms 70 ms
CHANGE BIAS DELAY	35 ms

5-185.[1] This time is based on the safe time quoted for an inter-block stop. See note 2.

5-186.[2] This time allows safe resumption of reading in either direction. The tape must be accelerated to its free-running speed before the first line of the next (following or preceding) block is encountered. Fixed stopping time <u>after</u> the EF Stop is executed allows the tape to move approximately 0.450 inches, consuming theoretically 7 milliseconds.

5-187.[3] The restriction on this time is derived from the available time between External Read instructions.

5-188.[4] The restriction on this time is derived from the available time between External Write instructions.

5-189.[5] The time over 600 milliseconds required for a reversal operation depends upon how soon tape movement was stopped at the conclusion of the last operation. The time given assumes that the EF Stop for the preceding read operation immediately followed the last ER. Allowing one millisecond between the EF stop and the last ER (as quoted for stopping in the table) does not appreciably change the reversal delay time quoted.

5-190.[6] A tape unit on which a tape has been previously rewound (or rewound with interlock) is said to be in a rewound condition until a tape Clear or computer Master Clear is effected. Until this time, any attempted forward movement of the rewound tape or a replacement tape incurs the basic theoretical reversal delay of 600 ms.

5-191.[7] A rewind initiation of a tape on a unit which has undergone a previous rewind operation (with the same tape or a replacement tape) consumes the "previous backward direction" delay if the tape is still in a rewound condition (see footnote 6). If an interim tape Clear or computer Master Clear has occurred, the basic reversal delay, or "previous forward direction" delay in this case, is incurred. Regardless of the delay, no tape movement is started, and another tape operation can be initiated after the delay time.

5-192. CONTINUOUS DATA INPUT MODE OPERATION

5-193. FORMAT

5-194. Continuous data information is recorded on magnetic tape by special off-line equipment. This equipment receives and records real-time observations which do not tolerate interruptions for purposes of formating the data. The recording density of lines per inch can be as few as 40. If continuously recorded observations are grouped by virtue of physical separations on the tape, each separate group comprises a block. Also, groups of data within a block

can be recognized by programmed detection of recorded codes.

5-195. Blank spaces of no more than 3 inches are used to separate blocks. As in variable block length reading operations, the end of a block is recognized if no recorded sprocket bits are detected for a distance of 0.06 inches. Indication of the end of the block is sent to IOA. A mod 6 check indication is sent to IOA with both the end of block detection and the detection of a particular recorded code called the Block Control Code. The mod 6 check determines whether the number of lines recorded in a group is an integral multiple of six. The group could be a block, or a set of observations identified by a Block Control Code. The end of the record on the tape is detected when no sprocket bits are noted for a distance of approximately 4.0 inches. Again, an indication is sent to IOA. With an end of record detection, the tape movement is stopped automatically. Blank spaces between blocks should be no longer than 3 inches in order that tape control can distinguish them easily from the end of record space.

5-196. The eight tracks of a line of tape are divided into four groups.

a. Data tracks - four binary digits of each line represent information.

b. Code tracks - two binary digits of each line represent a code used to identify or differentiate information.

c. Parity track - one binary digit is used for a parity check on each line.

d. Sprocket track - one binary digit is used as the sprocket bit for the line.

The bits described above are positioned on the line as shown in figure 5-7.

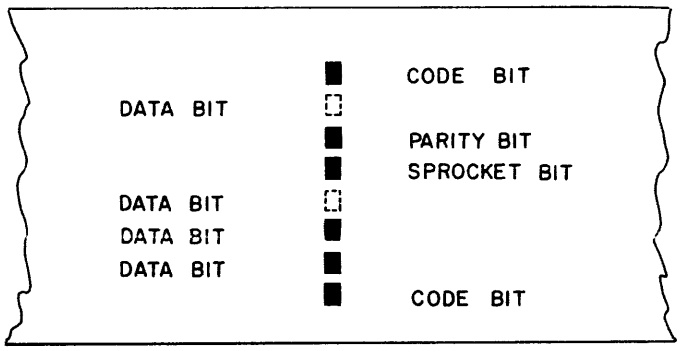


FIGURE 5-7. POSITIONING OF BITS ON A LINE

Magnetic tapes which have punched holes marking the bad spot areas should not be used during continuous data input recording. The off-line recording

equipment records through the bad spot area, and this information is disregarded when the marked bad spot is detected during reading operation. Magnetic tapes which are not marked for the bad spots can be used if the nature of the recorded data is such that all observations need not be correct. If maximum reliability of recorded data is required, then perfect tapes must be employed.

5-197. DATA ENTRY WORDS

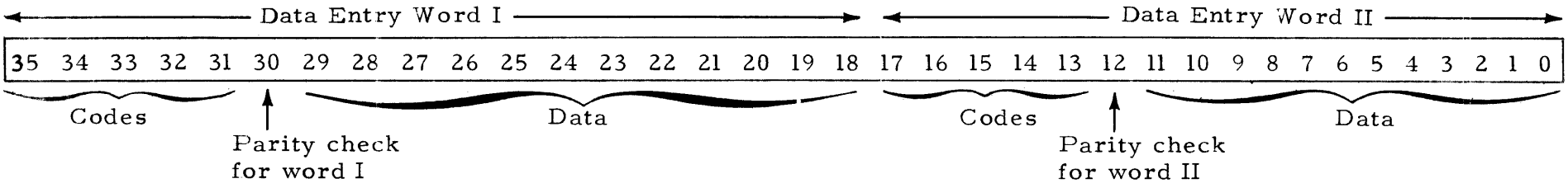
5-198. The Data Entry Word is the basic item, the observation recorded, on continuous input Unitapes. A Data Entry Word is assembled from each group of three consecutive lines on tape. A Data Entry Word consists of 12 bits of data, five code bits, and a parity check bit for the word. The five code bits identify the data in the Data Entry Word. The 12 data bits can be in any code, but are frequently a true 12-bit number, or three decimal digits in binary coded decimal form. Each line of tape is checked for a parity error. If a parity error occurs in any of the three lines from which the Data Entry Word is composed, the parity check bit of the word is "1." Otherwise, the parity check bit is "0." One code bit from the tape is discarded to allow for the parity check bit. Note that no parity error indication is sent to IOA. The check for a parity error must be programmed by inspecting the Data Entry Word. This is not necessary if the nature of the data is such that an occasional error can be tolerated.

5-199. After two Data Entry Words, or six consecutive lines of tape, are assembled in the Tape Register, the 36-bit word is transferred to IOB. During the transfer, the bits of the Data Entry Words are re-arranged to minimize the required shifting and masking operations which must be programmed. The appearance of Data Entry Words in IOB and on the tape is shown on page 5-48.

5-200. BLOCK CONTROL CODE

5-201. If both code bits of a line are "1," the line is said to contain a Block Control Code (BCC). Block Control Codes are used basically to initiate the actual reading of data from the tape, and to identify groups of Data Entry Words in the block. A Block Control Code is stored on the first line of the first Data Entry Word of a group. Recognition of a Block Control Code aids in positioning the tape and checking the synchronism of reading operations. Block Control Codes are recorded on the tape not closer than 2.4 inches and must also be recorded at a "mod 6" distance. A BCC mod 6 check determines whether the number of lines recorded from one Block Control Code to the next is an integral multiple of six. Since the Block Control Code appears more often, and hence is more effective in grouping data than is the inter-block space, the BCC check provides a more frequent check on the correct assembly of words than does the end of block mod 6 check.

IOB BIT POSITIONS



IOB BIT POSITIONS TRANSPOSED TO THE TAPE REGISTER

35	34	29	28	27	26	33	32	25	24	23	22	31	30	21	20	19	18	17	16	11	10	9	8	15	14	7	6	5	4	13	12	3	2	1	0

IOB BIT POSITIONS TRANSPOSED TO TAPE

Direction of tape movement ←

35	33	31	17	15	13	code bits
29	25	21	11	7	3	data bits
P	P	P	P	P	P	parity bits
S	S	S	S	S	S	sprocket bits
28	24	20	10	6	2	data bits
27	23	19	9	5	1	data bits
26	22	18	8	4	0	data bits
34	32	D	16	14	D	code bits
Data Entry Word I			Data Entry Word II			

(D = Discards)

Following are possible code combinations in IOB bit positions:

in $IOB_{35, 34}$ - 00, 01, 10, or BCC of 11

in $IOB_{17, 16}$ - 00, 01, 10

in $IOB_{33, 32}$ and $IOB_{15, 14}$ - 00, 01, 10

in $IOB_{31, 30}$ and $IOB_{13, 12}$ - 00, 10 (no parity error in word)
- 01, 11 (parity error in word)

These same code combinations are possible on tape, except that positions 31, D and 13, D cannot be 11. This combination would indicate a Block Control Code to tape control. The "1" in position D is discarded during word assembly, and the corresponding position in IOB receives a parity check bit.

5-202. PROGRAMMING.

5-203. The tape operations discussed in regards to continuous data input mode operation are the read forward, read backward, and stop operations. The rewind, rewind interlock, and change bias operations are programmed and accomplished in the same manner as they are in fixed and variable mode operation; no specific instructions concerning these operations are given in this section. The operation codes for the rewind, rewind interlock, change bias, and stop operations need not include the variable/continuous selection bits.

5-204. Neither a write nor a move operation is possible in the continuous data input mode. If an EF instruction with these operation codes, and the variable/continuous selection bits, is executed, the IOB output lockout is removed. No tape movement is started. The next External Function or External Write instruction creates an IOB lockout which is not removed since an IOB to TCR transmission is not allowed.

5-205. Note that an EF Change Mode instruction is necessary preceding the first EF Read tape instruction in continuous mode operation. The EF Change Mode instruction is discussed on page 5-10.

5-206. READ FORWARD

5-207. After the execution of the EF Read Forward instruction, tape movement is started, and lines are sensed from the tape. Data is discarded, however, until the first Block Control Code is sensed. When the first BCC is detected, the assembly of lines is begun in the Tape Register. The first word assembled includes the line in which the Block Control Code appeared. When six lines have been assembled, the 36-bit transmission from TR to IOB occurs. Each time such a transmission occurs, IOB must be read by an External Read instruction before the next transmission. Reading and assembly of words continues until an EF Stop Tape instruction is executed, or until an end of record space is detected.

5-208. The IOA Register is used to receive check indications during reading, similar to its use during variable block length reading. A check of the contents of IOA determines the course of action. Indications sent to IOA include the following: a BCC check, an end of block indication with a mod 6 check, and an end of record indication. These are explained subsequently.

5-209. BCC CHECK

5-210. With each TR to IOB transmission of a 36-bit word, a BCC check indication is sent to IOA. The BCC check error informs the programmer of a recording error, or reading error, such that Block Control Codes appear at line intervals not an integral multiple of six. If $IOA_{2,1,0}$ is "000" or "001," then a word has been transmitted to IOB, and an External Read of IOB must follow. If a Block Control Code appears in the word just assembled, and the line count was zero when the BCC was detected, then IOA_0 is equal to zero. (If the word contains no BCC, IOA receives the same indication of no BCC error.) If a Block Control Code appears in the word just assembled, and the line count was not zero when the BCC was detected, then IOA_0 is set to "1." The BCC error effects no stop of tape movement. Reading may be continued, or an EF

Stop Tape instruction may be programmed.

5-211. MOD 6 CHECK

5-212. The mod 6 check indication is sent to IOA with the end of block indication. The end of a block is detected when a 600 microsecond time lapse occurs with no detection of recorded sprocket bits. The end of block indication in IOA_1 = "1." If $IOA_{2,1,0}$ is "010" or "011," the end of the block was detected, and an External Read of IOB should not be programmed. If the bits are "011," a mod 6 error exists in the block just read. A mod 6 error indicates that the number of lines per block was not an integral multiple of six. The mod 6 error does not effect a stop of tape movement. Reading may be continued, or an EF Stop Tape instruction may be programmed. The lack of a mod 6 error, when a BCC error has occurred, might indicate faulty recording by the off-line equipment.

5-213. END OF RECORD

5-214. When four inches of tape pass under the read/write head without the detection of recorded sprocket signal, IOA_2 receives a "1." Thus, if an External Read of IOA, and a check of its contents, reveals that $IOA_{2,1,0}$ is "100," the end of the recorded area on the tape has been detected. Tape movement is automatically stopped, and the program for reading this tape should be discontinued.

5-215. IOB READ FAULT

5-216. The most probable cause of an IOB Read fault is the elapse of too much time between the execution of External Read instructions. If this is the case, the faulty program must be corrected before successful reading can be accomplished. The effects of an IOB Read Fault are given in the table on page 5-52.

5-217. READ BACKWARD

5-218. As in the read forward operation, reading is initiated by the detection of a Block Control Code. The assembly of lines begins, however, with the line adjacent, or immediately following, the line in which the BCC appears. This difference allows the bits of the Data Entry Words to be assembled in the same order as in reading forward. Data Entry Words in IOB have the same configuration in reading backward as in reading forward. Assembly of lines in a read backward operation continues until an EF Stop instruction is executed, or until detection of the tape leader causes an end of record stop.

5-219. Probably the most extensive use of the read backward operation is for re-positioning the tape in a backward direction. In this case, the reading most likely is desired only until a specific Block Control Code is reached or until a certain group of recorded observations is noted. The same indications are sent to IOA as in reading forward; the routine for reading backward must include External Reads of IOA.

5-220. STOP TAPE OPERATION.

5-221. An EF Stop Tape instruction terminates reading either in the inter-block space or within a block. Programming of the EF Stop Tape instruction is independent of the positioning of the Block Control Codes on the tape.

5-222. To stop reading in the middle of a block, the EF Stop Tape instruction is programmed after External Reads of IOA and IOB. An EF Stop Tape instruction programmed in the inter-block space follows an External Read of IOA. An EF Stop Tape instruction effects a "clear" of the tape system.

5-223. SYNOPSIS OF CONTINUOUS DATA INPUT MODE OPERATION

TAPE OPERATION	PROGRAMMING AIDS
Change Mode (page 5-10)	If tape control is not already prepared for continuous data input mode operation, program an EF Change Mode instruction before reading. Designate in (v) of EF instruction the following: magnetic tape master selection and change mode. See table page 5-15.
Read Forward/ Backward (pages 5-49 to 5-50)	Designate in (v) of EF Read instruction the following: magnetic tape master selection, read forward/backward, var/cont, tape unit. See table page 5-15. Observe limitations on allowable computation time between EF Start instruction and first ER, between successive ER's, between last ER and EF Stop Tape instruction. Before reading a word, read and check contents of IOA; then, accordingly, read IOA again, or read IOB and/or program an EF Stop Tape instruction. Program an EF Stop Tape for all desired stops during reading <u>except</u> at the end of record. <u>Omit</u> uniservo selection in EF Stop Tape instruction.
Stop (page 5-50)	Designate in (v) of EF instruction the following: magnetic tape master selection and stop code. See table page 5-16. Observe requirements for reading IOB and/or IOA before programming an EF Stop Tape instruction. <u>Omit</u> uniservo selection in the stop code.
Rewind Rewind Interlock Change Bias	See table for Fixed Block Length Mode Operation, page 5-27.

5-224. SYNOPSIS OF TAPE OPERATION ERRORS

5-225. The errors listed below either cause computer stops or stalls, or are directly responsible for computer faults. A Master Clear is required to remove all fault conditions. The registers and indicators listed are discussed in the OPERATIONS section.

FAULT	COMPUTER FAULT OR STOP COND.	INDICATORS	COMMENTS
<720 Sprocket Error (during reading only)	IOB lockout stall B Fault Stop	Sprocket Error fault indication MT B Fault indication (PCR) = ER Tape stopped	See page 5-19 for discussion Fixed mode only
>720 Sprocket Error (during reading)	B Fault Stop	Sprocket Error fault indication MT B Fault indication	See page 5-19 for discussion Fixed mode only
>720 Sprocket Error (during moving)	B Fault Stop	Sprocket Error fault indication MT B Fault indication	See page 5-23 for discussion Fixed mode only
IOB Read Fault, Class I	B Fault Stop	IOB Fault indication IOB Class I fault indication Tape stopped	See page 5-20 for discussion All modes
No Information Fault (too few EW's)	B Fault Stop	MT B Fault indication No information fault indication Tape stopped	See page 5-22 for discussion Fixed mode

FAULT	COMPUTER FAULT OR STOP COND.	INDICATORS	COMMENTS
No Information Fault (too many EW's)	B Fault Stop	MT B Fault indication No Information fault indication Tape stopped	See page 5-22 for discussion Fixed mode
No Information Fault (EW too late)	B Fault Stop	MT B Fault indication No Information fault indication Tape stopped	See page 5-38 for discussion Variable mode
No Information Fault (EF Stop too late during writing)	B Fault Stop	MT B Fault indication No Information fault indication Tape stopped	See page 5-38 for discussion Variable mode
No Information Fault (No EF Stop to terminate writing)	B Fault Stop	MT B Fault indication No Information fault indication Tape stopped	See page 5-38 for discussion Variable mode
Uniservo Interlock Fault	B Fault Stop	MT B Fault indication Uniservo Interlock fault indication Extinguish "Ready" light on uniservo Tape stopped	See page 5-62 for discussion All modes
Select Error	Possible IOB lockout stall	Select Error indication (IOB) =(v) of 1st EF Possible (PCR) =2nd EF, EW or ER after 1st EF Tape movement not started	See page 5-59 for discussion All modes

ERROR	COMPUTER FAULT OR STOP COND.	INDICATORS	COMMENTS
Read Backward n blocks when m (<n) are recorded	IOB lockout stall	(PCR) = ER (TCR) = Read backward (IOB) = 0 Tape rewound & stopped on leader	A stall on an IOB lockout is produced by the first ER instruction for block n + 1. Fixed mode only
Read Forward n blocks when m (<n) are recorded	IOB lockout stall B Fault Stop	(PCR) = ER (TCR) = Read forward (IOB) = 0 MT B Fault indication Uniservo Interlock fault indication	The tape movement is not stopped until the end of the tape is reached, which produces the Interlock fault. The first extra ER command establishes the IOB lockout and stall. Fixed mode only.
Move Backward n blocks when m (<n) are recorded	Possible IOB lockout stall	Possible (PCR) = 2nd EF; EW or ER after 1st EF (TCR) = Move backward (BK) = n - m (IOB) = (v) for 1st EF Tape rewound and stopped on leader	See page 5-24 Fixed & variable modes.
Move Forward n blocks when m (<n) are recorded	B Fault Stop Possible IOB lockout stall	Possible (PCR) =2nd EF; EW or ER after 1st EF. (TCR) = Move forward (BK) = n - m (IOB) = (v) for 1st EF Uniservo Interlock fault indication MT B Fault indication Tape movement stopped on trailer	See Page 5-23 Fixed and variable modes

ERROR	COMPUTER FAULT OR STOP COND.	INDICATORS	COMMENTS
Failure to terminate reading with an EF Stop Tape instruction 1) last recorded block	B Fault Stop	(TCR) = logical sum of (TCR); and (v) of next EF-v MT B Fault indication Uniservo Interlock fault indication	Tape movement is not stopped until the end of the tape is reached. This produces the Uniservo Interlock fault. Since TCR is not cleared, the specification of an operation by an EF instruction is added logically to the read operation code still in TCR. The resulting behavior of the tape units is unpredictable. Fixed mode only
2) an intermediate block	B Fault Stop	IOB Fault indication IOB Class I fault indication Tape stopped	The IOB Class I fault occurs when no ER's are programmed to receive the recorded information being read from the tape. The cause of the fault is interpreted as "too few ER's" discussed under IOB Read Fault on page 5-20 Fixed and variable modes
Failure to terminate writing with an EF Stop Tape instruction	B Fault Stop	MT B Fault indication No Information Fault indication. Tape stopped	See page 5-22 for discussion. Fixed block mode
Programming a Change Bias & any other tape operation in the same EF instruction	Possible lockout stall	Possible (PCR) = ER; EW Possible illumination of Read Bias indicators No Tape movement	The specified tape operation is stalled, because of the absence of a Uniservo selection, while the bias change is completed. At completion, TCR is cleared and IOB is ready to accept another EF or an EW instruction. The execution of a second EW or of an ER instruction causes the IOB lockout stall. If a block count was not specified with a Change Bias/Move combination, TCR is cleared immediately, and no bias change occurs. Another EF can be executed immediately. Fixed and variable modes

5-226. OPERATION

5-227. OPERATION INDICATORS.

5-228. Tape operation is reflected by the condition of indicators on the uniservos, the tape control cabinet, and the left section of the computer control panel. These indicators and their reactions to tape operation and tape operation faults are discussed in the following paragraphs.

5-229. UNISERVO

5-230. The uniservo (see figure 5-8) used with the Univac Scientific 1103A is depicted to the left. The uniservo "Ready" indicator (green) is between the tape reel panel doors, upper center section of the uniservo. This indicator is illuminated when the uniservo interlock circuit is energized; i. e., when power has been applied to the uniservo, when the tape reel panel door switch is set to its ON position, and when the forward limit, buffer tape detector, left tape loop, and right tape loop switches are in their normally closed position. If the Ready indicator is not illuminated, operation cannot be initiated on this uniservo.

5-231. A failure to have the interlock circuit not energized because power has not been applied to the uniservos would not usually occur. Power is normally applied to the uniservos (and the tape control system) at the same time power is turned on for the computer. Normally, if it is noted that the uniservo Ready indicator is not illuminated, the interlock circuit is not energized because one of the switches in the circuit is open. The condition which caused the switch to be opened must be corrected before any operation on the unit can be undertaken. The interlock switches and the conditions which cause them to open are discussed in the following paragraphs.

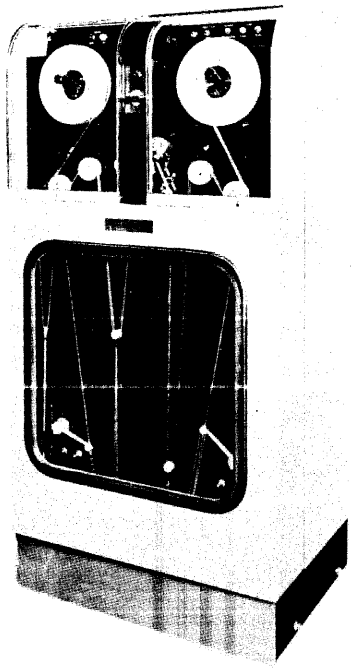


Figure 5-8

5-232. The tape reel panel door switch is located immediately below the Ready indicator. This interlock switch must be set to OFF to open the left tape reel panel door and cannot be returned to its ON position until the door is closed. Effectively, then, opening the uniservo door causes the Ready indicator to be dropped. The door must be closed and the door switch reset to ON before the interlock circuit is energized.

5-233. A plastic buffer tape is inserted between the read/write head and the metallic recording tape. This plastic tape serves to reduce both tape wear and friction. When this tape is broken or the supply is exhausted, the detector interlock switch is opened, dropping the "Ready" indicator. The replacement of the plastic tape by maintenance procedures closes this interlock switch.

5-234. The forward limit interlock switch is opened when the magnetic tape moves into the trailer area on its far end, i.e., the left-hand tape reel is depleted and the right-hand tape reel contains all of the tape. (The forward limit switch is opened when the "rubber bumpers" on the trailer are detected). To close this switch, to energize the uniservo interlock circuit, and to illuminate the "Ready" indicator, the tape must be rewound past its trailer position. This is accomplished by opening the panel door and manually turning the reel in the counter-clockwise direction several times (until the switch no longer makes contact with the rubber bumpers). The complete rewinding of the tape onto the left-hand reel can then be accomplished by the normal rewind operation which can be instigated manually from the computer control panel, or under program control.

5-235. The right and left tape loop switches are opened, and the "Ready" indicator dropped, when the tape loops are out of normal position. This could be caused by tape breakage or possibly could result from faulty operation of control circuits in the uniservo. Maintenance procedures are necessary to correct these conditions.

5-236. De-energizing the uniservo interlock circuit, and dropping the "Ready" indicator, could also be caused by blowing a fuse. A blown fuse in the uniservo cabinet is shown by the illumination of the Fuse indicator, which is located inside the right-hand tape reel door, above the tape reel mounting. Illumination of the Fuse indicator shows that some part of the uniservo is inoperative. However, the uniservo interlock circuit is not necessarily de-energized. Detection and replacement of the blown fuse is a maintenance procedure.

5-237. Other indicators located on the uniservo are the Temp Fault, Rewind Interlock, Master Tape, and Clutch, Stop and Go. The Temp Fault, Rewind Interlock and Master Tape indicators are located inside the right-hand tape reel door, above the tape reel mounting.

5-238. The Temperature indicator is illuminated when a temperature rise above 120° F. is detected in the uniservo. This condition causes a computer A Fault and illuminates the Temp indicator in the A Fault Group on the com-

puter control panel. Computer operation is halted by an A Fault condition. Operation is resumed after corrective maintenance by depressing the Clear A Fault button (unless a B Fault has resulted from tape reading or writing occurring at the time of the computer stop. If this is the case, either (1), the MTB fault indicator and the No information fault indicator on the computer control panel are illuminated, or (2), the IOB fault indicator on the computer control panel is illuminated).

5-239. The Rewind Interlock indicator is illuminated when the magnetic tape has been rewound with interlock on the left-hand tape reel. This condition indicates that the tape on this uniservo should be replaced before this unit is used again. Opening the door to replace this tape drops the "Ready" condition of this unit (because the door switch must be set to OFF) and drops the rewound with interlock condition.

5-240. The Master Tape indicator is illuminated when an attempt is made to write on a tape fitted with a Master Tape Ring. The Master Tape Ring is a flat metal spiral which is inserted in the inner circumference of a tape reel. This tape reel, fitted on the left tape reel mount, prevents the initiation of a writing operation on the uniservo. (The resulting "not ready" condition is discussed later under indicators on the Supervisory Control Panel.) The Master Tape Ring is easily inserted and removed from the inner side of the tape reel.

5-241. Inside the left-hand tape reel door, above the tape reel mounting, are indicators labeled Clutch, Stop (red) and Go (green). The Go indicator is illuminated when the clutch is energized. This indicator remains illuminated until a stop tape signal is received by the tape drive mechanism to operate the brake, thus releasing the clutch. At this time the Stop indicator is illuminated, and remains illuminated, until the tape is re-started or until power is dropped from the unit.

5-242. TAPE CONTROL CABINET

5-243. This cabinet is located immediately to the left of the Power Supply Cabinet. Located inside the right-hand door are the Logical Number Selection switches and the Select Error indicator. (See figure 5-9). Each uniservo is physically defined by one of the numbers 1, 2, ... 10, depending upon the number of uniservos installed. For instance, if an installation has eight uniservos, it would be expected that the numbers 1 ... 8 would define the eight units. The Logical Number Selection switches are labeled uniservo 1, uniservo 2, etc. Encircling the switches are the numbers 1 through 10. A tape unit is assigned a logical designation by turning the appropriate selection switch so that the white line on the switch is in line with the number desired. The numbers available for logical assignment depends upon the number of tape units installed, i.e., if eight uniservos are installed, any of these units may be logically assigned any of the numbers one through eight. Thus, if the eight tape units at an installation are physically defined as uniservo 1 ... 8, the switches labelled uniservo 9 and uniservo 10 should be set to the logical designations of 9 and 10. It is not allowable for two switches to be set to the same logical designation even though some of the switches define a non-existent tape unit.

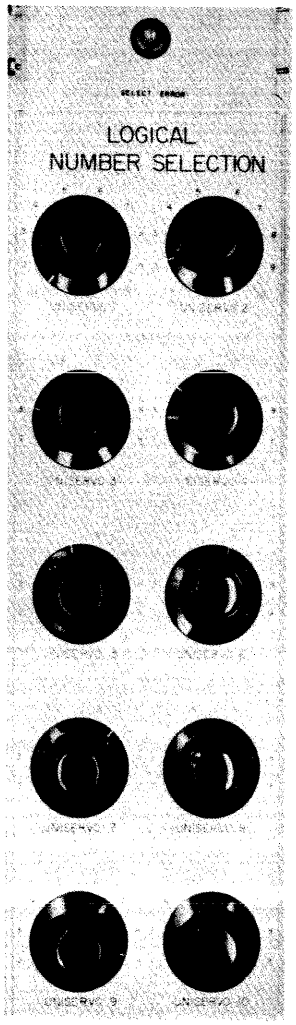


Figure 5-9

5-244. The Select Error indicator is illuminated by setting two Logical Number Selection switches to the same number, i.e., giving two "uniservos" the same logical designation. The error is indicated regardless of whether or not the "uniservo" is non-existent or out of service for maintenance reasons. The Select Error prevents an IOB to TCR transmission for the first EF tape instruction attempted. The IOB lockout is not removed; hence the next attempted External instruction causes a computer stall. Correcting the selection causes the Select Error light to be extinguished. Operation is automatically resumed.

5-245. Caution must be used in opening the tape control cabinet door, or any cabinet door, to note the condition of the switches and the error indicator. Before the cabinet door is opened, the Bypass Cabinet Interlock key in the Test Switch Group, right section of the computer control panel, must be turned to its Abnormal position. The failure to do this before opening a cabinet door causes an emergency power drop to the computer system, and maintenance procedures are necessary to resume operation.

5-246. SUPERVISORY CONTROL PANEL

5-247. Represented on the left section of the Supervisory Control Panel are components of the tape control system. Indicators here assist the operator in manual operation of the uniservos, and aid in diagnosing certain error conditions. Registers whose contents are displayed are TCR, Tape Control Register, TR, Tape Register, and BK Block Counter.

5-248. The lower right corner of the tape control panel is illustrated in figure 5-10. In the MT Controls group are buttons labelled Clear and Start. The Start button is depressed to manually initiate a tape operation. Depression of the Clear button effects a Clear of the tape control system only, not the entire computer. This should not be confused with the computer Master Clear, as depression of the tape Clear button does not disturb the main computer control.

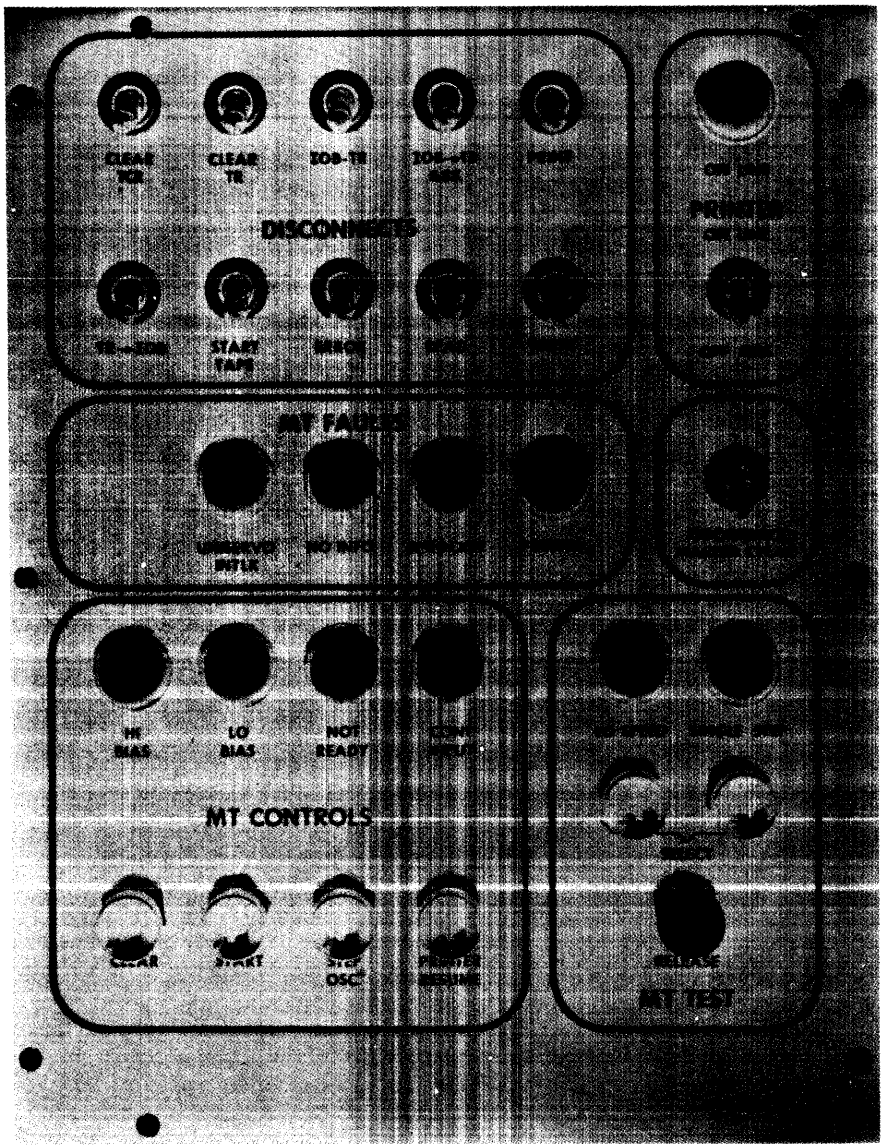


Figure 5-10.

5-249. Indicators in the MT Controls group are labelled Hi Bias, Lo Bias, Not Ready, and Cont Input. The illumination of the one of the indicators labelled high or low bias shows the selection of a read bias other than normal. Illumination of the indicator labelled Cont Input shows that an EF Change Mode instruction has prepared tape control for reading in the continuous data input mode. While this indicator is illuminated, an EF Read instruction with the Select Var/ Cont bits causes reading in continuous data input mode. The indicator is extinguished by another EF Change Mode instruction, or by a computer Master Clear.

5-250. The Not Ready indicator is illuminated when an attempt is made to start operation on the uniservo specified in TCR, and this uniservo is found to be "not ready." A referenced uniservo is not ready when it has no power; when a rewind interlock condition exists on this tape unit, or this tape unit is currently rewinding with interlock at completion; or when an attempt is made to write on a reel of tape fitted with a Master Tape Ring. Another External Function or External Write instruction executed before the elimination of the not ready condition creates an IOB lockout condition. Computer operation can continue until a second EF or EW instruction is executed, at which time the computer is stalled. The first External Read attempted with a not ready condition causes a computer stall. Steps should be taken immediately upon noting the Not Ready indication to eliminate the condition. The corrective procedure is as follows:

(1) Eliminate the not ready condition, according to the cause, as explained subsequently.

(2) Depress the tape START button.

5-251. A not ready condition due to no power in the specified uniservo, occurs when the uniservo interlock circuit is de-energized. This is evidenced by the extinguished Ready light on that uniservo. (The uniservo Ready light and uniservo interlock circuit are discussed in the paragraphs on uniservo indicators). Unless the lack of power is due to the tape reel panel door being open, the restoration of power will probably have to be corrected by maintenance procedures.

5-252. A not ready condition due to a rewind interlock condition on the specified uniservo, is indicated by the illumination of the Rewind Interlock indicator on that uniservo. (A rewind interlock condition exists after a tape unit has been rewound with interlock until the tape reel panel door switch is opened on that uniservo). Recovery from this "not ready" condition is accomplished by replacing the tape reel on this uniservo, or by merely opening and closing the uniservo tape reel panel door switch. No protection is provided against another IOB to TCR transmission before the removal of a Rewind Interlock "not ready" condition. However, any External instruction following the EF tape instruction which created the "not ready," is most likely to be an ER or EW instruction.

5-253. If the Not Ready indicator is illuminated, and the Master Tape indicator on the uniservo is illuminated, then the not ready condition is caused by an attempt to write on a tape reel which has been fitted with a Master Tape Ring. The "not ready" condition can be eliminated by any of the following steps.

(a) Place a different tape reel on the uniservo, or replace the same tape reel on the uniservo, after removing the Master Tape Ring; or

(b) Specify a different uniservo for operation by setting a different uniservo number in TCR; or

(c) Specify a different uniservo for operation by changing the settings of the Logical Number Selection switches in the tape control cabinet.

5-254. In the MT Fault group on the tape control panel are indicators labelled No Info, Sprocket, and Uniservo Intlk. All of these faults illuminate the MT B Fault indicator and cause a B Fault stop. The No Information and Sprocket Error faults are discussed elsewhere. (See the table on page 5-52.) These faults can be cleared, and their indicators, the MT Fault indicator, and the B Fault indicator are extinguished, by depressing the Master Clear button on the computer control panel.

5-255. The Uniservo Interlock fault is caused by a drop of power in a uniservo which is in operation. The failure of power is due to the uniservo interlock circuit being de-energized. (The uniservo interlock circuit is discussed under uniservo indicators). This condition is shown on the faulty uniservo by the extinguishment of the "Ready" indicator. The computer is stopped approximately 15 ms after the power failure. Computation can be resumed after depression of the Master Clear button. However, the procedure should be as follows. Depress the Force Stop button, if the computer fault stop has not yet occurred; Master Clear the computer; correct the fault condition; make the desired selections on the computer control; and depress the computer Start button. Correction of the fault condition will probably require maintenance procedures unless the fault was caused by inadvertently opening the tape reel panel door on the tape unit referenced.

5-256. PREPARATION FOR OPERATION

5-257. The procedure for preparing for tape operation under program control, or manual control is as follows, assuming that the uniservos have been properly equipped with tape reels. (The left tape reel should not be fitted with a Master Tape Ring.)

a. Determine whether those Logical Number Selection switches which physically define installed uniservos have been set to the logical number designations used in the program. (The numbers which can be used for logical designations cannot exceed the number of installed uniservos. Also two switches cannot be set to the same number.)

b. Check for the illumination of the green "Ready" indicator on each tape unit to be used. If this indicator is not illuminated, the resulting Not Ready indication leads to an operational delay.

c. Check for the illumination of the Rewind Interlock indicator on each tape unit to be used. This condition should be eliminated before attempting any operation on the uniservo to prevent a Not Ready indication and an operational delay.

The procedure to replace any rewound tape is as follows:

a. Turn the door switch to its OFF position and open the left tape reel panel door.

b. The clip connecting the magnetic tape and the leader should be positioned immediately below the tape reel. Disconnect the magnetic tape and the leader by releasing the clip.

c. Pull forward the holding latch on the tape reel mounting. This releases a locking pin directly under the knob from its position in one of the slots in the inner circumference of the tape reel. Remove the tape reel from the tape reel mounting.

d. Place another rewound tape on the tape reel mounting so that the tape winding is in the clockwise direction. Return the holding latch to its closed position, first positioning the tape reel so that the locking pin is inserted into any of the slots in the inner circumference of the tape reel.

e. Join the magnetic tape and the leader by fastening the clip connection.

f. Turn the tape reel counterclockwise until any tape slack is taken up.

g. Close the panel door and set the door switch to its ON position.

5-258. MANUAL OPERATION.

5-259. Certain tape operations can be manually initiated, but only if the main computer operation is interrupted, and if the computer is set to operate in Test mode. The operations which can be successfully completed after a manual initiation are rewind, rewind interlock, move forward and backward, and change bias. To initiate one of these operations, use the following procedure:

a. Depress the computer Force Stop button.

b. Depress the computer Master Clear button.

c. Set the computer to the Test Mode.

d. Set the desired operation code and uniservo selection (except for a Change Bias operation) in TCR by depressing the appropriate "set" (black) buttons.* (The white button is depressed to clear the register.)

e. If a Move operation is desired, the number n of blocks to be moved is inserted in the Block Counter, BK.

f. Depress the tape Start button.

The computer should be returned to Normal Mode operation after completion of the tape operation.

5-260. At the completion of the move forward, move backward, and change bias operations, the Tape Control Register and the Block Counter are automatically cleared. If the bias is changed to high or low, one of the bias indicators in the MT Controls group is illuminated. If a manual rewind operation is performed, TCR is cleared when the rewind operation is initiated.

* The bits set into $TCR_{11...0}$ are as follows:

For a rewind operation,	010 000 00u uuu
For a rewind interlock operation,	100 000 00u uuu
For a move forward operation,	000 000 10u uuu (fixed block length)
For a move backward operation,	000 001 10u uuu (fixed block length)
For a change to low bias,	000 000 001 110
For a change to high bias,	000 000 001 111
For a change to normal bias,	000 000 001 101

APPENDIX A

UNIVAC SCIENTIFIC INSTRUCTION REPERTOIRE (MODEL 1103A)

INSTRUCTION (and code notations)	FUNCTION (and anomalies)	MC TO MC EXECUTION TIME (in microseconds)
01 Floating Point Polynomial Multiply, FPuv	$\underline{(u)}(Q)_i + (v) \rightarrow Q$ u=Q or v=A or Q yields abnormal result.	Min: 302 Max: 656
02 Floating Point Inner Product, FIuv	$(Q)_i \rightarrow F_4$; $\underline{(u)}(v) + (Q)_i \rightarrow Q$ v=Q yields abnormal result.	Min: 324 Max: 678
03 Floating Point Unpack, UPuv	$\underline{(u)}_m \rightarrow u$; $(u)_c \rightarrow v$ If u, v=A, A or Q, Q $(u)_m$ is lost.	60
04 Floating Point Normalize Pack, NPuv	Pack (u) with mantissa from $(u)_i$ and characteristic from $v_{34} \ldots v_{27}$ u=v yields abnormal result.	142 + 4s Min: 142 Max: 278
05 Floating Point Round Option, FRj-	If j=1, normalize without rounding until a Master Clear or FRj- with j=0	18
11 Transmit Positive, TPuv	$\underline{(u)} \rightarrow v$ If u=v=A, then $(A)_f = D(A_R)_i$.	38
12 Transmit Magnitude, TMuv	$\underline{\lvert(u)\rvert} \rightarrow v$ If u=v=A, then $(A)_f = D\lvert(A_R)\rvert_i$.	38
13 Transmit Negative, TNuv	$\underline{(u)} \rightarrow v$ If u=v=A, then $(A)_f = D(A_R)_i$.	38
14 Interpret, IP--	(PAK) $\rightarrow$ v of (F_1), $F_2 \rightarrow$ PAK	30
15 Transmit U Address, TUuv	$\underline{u_{29} \ldots u_{15}} \rightarrow v_{29} \ldots v_{15}$ Fault if v=A or Q.	38
16 Transmit V Address, TVuv	$\underline{u_{14} \ldots u_0} \rightarrow v_{14} \ldots v_0$ Fault if v=A or Q.	38
17 External Function, EF-v	(v) $\rightarrow$ IOB to select and operate external equipment	28

INSTRUCTION (and code notations)	FUNCTION (and anomalies)	MC TO MC EXECUTION TIME (in microseconds)
21 Replace Add, RAuv	D(u) + D(v)→A; $(A)_R$→u if u ≠ A If u=A, then $(A)_f = D(A_R) + D(v)$ If v=A, then $(A)_f = 2D(u)_i$. If u=v=A, then $(A)_f = 2D(A_R)_i$.	60
22 Left Transmit, LTjkv	Shift (A) left k places; j=0, (A_L)→v; j=1, (A_R)→v	32 + 2k
23 Replace Subtract, RSuv	D(u) - D(v)→A; $(A)_R$→u if u≠A If u=A, then $(A)_f = D(A_R)_i - D(v)$. If v=A, then $(A)_f = 0$.	62
27 Controlled Complement, CCuv	(u) ⊕ (v)→A_R; (A_R)→u if u ≠ A If v=A, then $(A_R)_f = 0$.	52
31 Split Positive Entry, SPuk	S(u)→A; shift (A) left k places	32+2k
32 Split Add, SAuk	S(u) + (A); shift (A) left k places	32+2k
33 Split Negative Entry, SNuk	[S(u)]→A; shift (A) left k places	34+2k
34 Split Subtract, SSuk	(A) - S(u); shift (A) left k places	34+2k
35 Add and Transmit, ATuv	(A) + D(u); (A_R)→v if v ≠ A	44
36 Subtract and Transmit, STuv	(A) - D(u); (A_R)→v if v ≠ A	46
37 Return Jump, RJuv	(PAK)→$u_{14} \ldots u_0$; v→PAK Fault if v=A or u=A or Q	36
41 Index Jump, IJuv	D(u) - 1→A if u ≠ A; if D(u) > 0, (A_R)→(u), jump to v If u is A, then $(A)_f = (A)_i - 1$. Fault if jump to v=A	No Jump: 42 Jump: 52
42 Threshold Jump, TJuv	If D(u) > (A), jump to v $(A)_f = (A)_i$ Fault if jump to v = A.	No Jump: 42 Jump: 42

INSTRUCTION (and code notations)	FUNCTION (and anomalies)	MC TO MC EXECUTION TIME (in microseconds)
43 Equality Jump, EJuv	If D(u)=(A), jump to v $(A)_f=(A)_i$ Fault if jump to v=A.	No Jump: 54 Jump: 54
44 Q - Jump, QJuv	Q_{35}=1, jump to u; Q_{35}=0, jump to v; shift (Q) left one Fault if jump to u=A or v=A.	18
45 Manually Selective Jump, MJjv	If j=0, jump to v; j=manual selection of 1, 2, or 3, jump to v Fault if jump to v=A.	No Jump: 18 Jump: 18
46 Sign Jump, SJuv	A_{71}=1, jump to u; A_{71}=0, jump to v Fault if jump to u=A or v=A.	18
47 Zero Jump, ZJuv	(A)≠0, jump to u; (A)=0, jump to v Fault if jump to u=A or v=A.	30
51 Q-Controlled Transmit, QTuv	(A)=S[(u)⊗(Q)]; (A_R)→v if v≠A If u=A, $(A)_f=S[(A_R)_i \otimes (Q)]$. If u=Q, $(A)_f=S(Q)$.	44
52 Q Controlled Add, QAuv	$(A)_f=(A)+S[(u)\otimes(Q)]$; $(A_R)_f$ v if v ≠ A If u=A, $(A)_f=(A)_i+S[(A_R)_i \otimes (Q)]$. If u=Q, $(A)_f=(A)_i+S(Q)$.	46
53 Q Controlled Substitute, QSuv	$(A)=S[(u)\otimes(Q)]+S[(v)\otimes(Q)]$; (A_R)→v if v ≠ A If v = A, $(A)_f=S[(u)\otimes(Q)]$. If u = Q, v = A, $(A)_f=S(Q)$. If v = Q, $(A)_f=S[(u)\otimes(Q)]+S(Q)$! If u = v = Q, $(A)_f=2^{36}-1$.	74
54 Left Shift in A, LAuk	D(u)→A, if u≠A; shift (A) left k places; (A_R)→u if u≠A If u=A, $(A)_f=(A)_i$ shifted	44+2k
55 Left Shift in Q, LQuk	(u)→Q; shift (Q) left k places; (Q)→u	42+2k

INSTRUCTION (and code notations)	FUNCTION (and anomalies)	MC TO MC EXECUTION TIME (in microseconds)
56 Manually Selective Stop, MSjv	If j=0, stop; if j=manual selection of 1, 2, 3, stop; v→PAK Fault if jump to v=A	No Stop: 18 Stop: 4
57 Program Stop, PS--	Final stop with indication	2
61 Print, PR-v	$v_5 \ldots v_0$ to TWR to operate typewriter	34
63 Punch, PUjv	$v_5 \ldots v_0$ to HPR; punch HPR, and 7th level if j=1	34
64 Floating Point Add, FAuv	$(u) + (v) \longrightarrow Q$ v=A or Q yields abnormal results	Min: 156 Max: 308
65 Floating Point Subtract, FSuv	$(u) - (v) \longrightarrow Q$ v=A or Q yields abnormal results	Min: 156 Max: 308
66 Floating Point Multiply, FMuv	$(u)(v) \longrightarrow Q$ v=Q yields abnormal results	Min: 182 Max: 386
67 Floating Point Divide, FDuv	$(u) \div (v) \longrightarrow Q$ v=A yields abnormal results	Min: 650 Max: 660
71 Multiply, MPuv	$(u) \longrightarrow Q$; $(Q)(v) \longrightarrow A$ If v=A, $(A)_f = 0$ If v=Q, $(A)_f = (u)^2$	Min: 116 Max: 410
72 Multiply Add, MAuv	$(u) \rightarrow Q$; $(A)_i + (Q)(v) = (A)_f$ If v=A, $(A)_f = (A)_i + (u)(A_L)_i$ If v=Q $(A)_f = (A)_i + (u)^2$	Min: 188 Max: 482
73 Divide, DVuv	$(A)_i = (Q)(u) + R$; $0 \leq R < \lvert(u)\rvert$ $(A)_f = (R)$; $(Q) \longrightarrow v$ If v=A, $(A)_f = D(Q)_f$	$482+8\ A_{71}$
74 Scale Factor, SFuv	$D(u) \longrightarrow A$, if u≠A; shift (A) left 36 places; then shift until $A_{35} \neq A_{34}$; $k \longrightarrow v_{14} \ldots v_0$ Fault if v=Q or A If u=A, $0 \leq k \leq 71$ If u≠A k=0 or $37 \leq k \leq 71$ k=37, $(A)_i$ was 0's or 1's k=0, initial $A_{35} \neq A_{34}$ and $(A_L)_i$ was 0's or 1's	$122+2\gamma$ $\gamma = (36-k) \bmod 72$
75 Repeat, RPjnw	$w \longrightarrow v$ of (F_1) execute NI n times, jump to F_1	$54+R_n+_p$
76 External Read, ERjv	If j=0, (IOA)→ v; j=1, (IOB) → v If j=0, $v_7 \ldots v_0$ = (IOA) $v_{35} \ldots v_8 = 0$	30
77 External Write, EWjv	If j=0, $v_7 \ldots v_0 \longrightarrow$ IOA; j=1, (v)→IOB	28

TABLE OF POWERS OF TWO

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125

MAGNETIC DRUM RESERVE SPACE ADDRESSES

Drum addresses which are coded to reference reserve space locations are listed in the left-hand column. The locations referenced are listed in the right-hand column. This list of addresses assumes that 160 reserve space locations are available. The letter x specifies a particular drum group 4,5,6, or 7.

FOUR INTERLACE

CODED ADDRESS		INTERLACED ADDRESS
x0000	=	x0000
x0001	=	x0004
x0002	=	x0010
.		.
.		.
.		.
x0047	=	x0234
x2000	=	x0001
x2001	=	x0005
.		.
.		.
.		.
x2047	=	x0235
x4000	=	x0002
.		.
.		.
.		.
x4047	=	x0236
x6000	=	x0003
x6001	=	x0007
x6002	=	x0013
.		.
.		.
.		.
x6047	=	x0237

EIGHT INTERLACE

CODED ADDRESS		INTERLACED ADDRESS
x0000	=	x0000
x0001	=	x0010
x0002	=	x0020
.		.
.		.
.		.
x0023	=	x0230
x1000	=	x0001
x1001	=	x0011
.		.
.		.
.		.
x1023	=	x0231
x2000	=	x0002
.		.
.		.
.		.
x2023	=	x0232
etc.		etc.
x7000	=	x0007
.		.
.		.
.		.
x7023	=	x0237

SIXTEEN INTERLACE

CODED ADDRESS		INTERLACED ADDRESS
x0000	=	x0000
x0001	=	x0020
x0002	=	x0040
.		.
.		.
.		.
x0011	=	x0220
x0400	=	x0001
.		.
.		.
.		.
x0411	=	x0221
x1000	=	x0002
.		.
.		.
.		.
x1011	=	x0222
etc.		etc.
x7400	=	x0017
.		.
.		.
.		.
x7411	=	x0237

THIRTY-TWO INTERLACE

CODED ADDRESS		INTERLACED ADDRESS
x0000	=	x0000
x0001	=	x0040
.		.
.		.
.		.
x0004	=	x0200
x0200	=	x0001
.		.
.		.
.		.
x0204	=	x0201
x0400	=	x0002
.		.
.		.
.		.
x0404	=	x0202
etc.		etc.
x7600	=	x0037
.		.
.		.
.		.
x7604	=	x0237

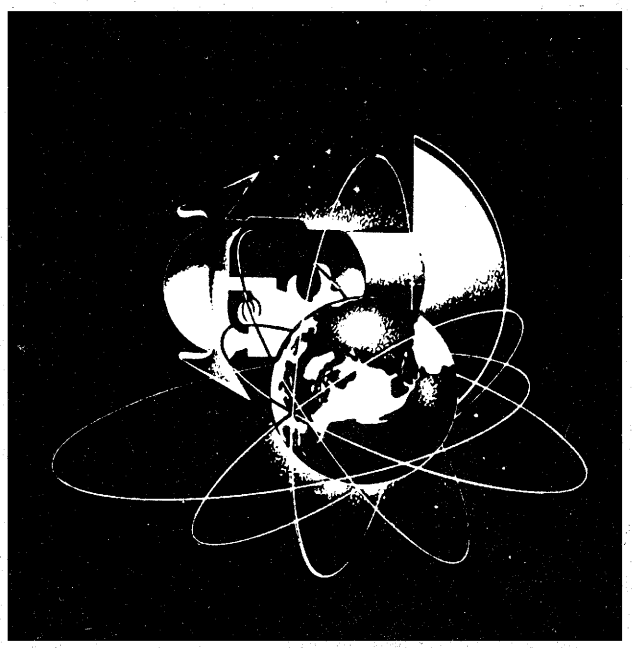
SIXTY-FOUR INTERLACE

CODED ADDRESS		INTERLACED ADDRESS
x0000	=	x0000
x0001	=	x0100
x0002	=	x0200
x0100	=	x0001
x0101	=	x0101
x0102	=	x0201
etc.		etc.
x3700	=	x0037
x3701	=	x0137
x3702	=	x0237
x4000	=	x0040
x4001	=	x0140
x4100	=	x0041
x4101	=	x0141
etc.		etc.
x7700	=	x0077
x7701	=	x0177

IOB SELECT BITS AND CODE WORDS

EQUIPMENT SELECTED BY EF-V	(v) OF EF-v (in octal)	FUNCTION
Photoelectric Paper Tape Reader (IOB Select bit = IOB_{33})	10 00002 00000 10 00001 00000 10 00003 00000	Start free run Stop free run Step
Univac Card Equipment (IOB Select bit = IOB_{35})	40 00000 00000 40 00000 00000 40 00000 00040 40 00000 00020 40 00000 00010 40 00000 00004 40 00000 00002 40 00000 00001 Combined code words effect simultaneous functions, e.g.. 40 00000 00012 40 00000 00005	Start Interrupt Start free run Stop free run Start, pick punch card Start, pick read card Start, punch Start, read Start, pick punch card, punch Start, pick read card, read
Univac Magnetic Tape System (IOB Select bit = IOB_{31})	See separate tables for tape system in section 5.	

univac® SCIENTIFIC model 1103A programming manual



UNIVAC—The FIRST Name in Electronic Computing Systems

U 1519