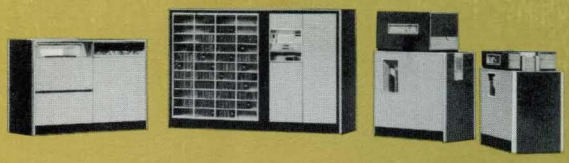


UNIVAC®
Solid-State 80

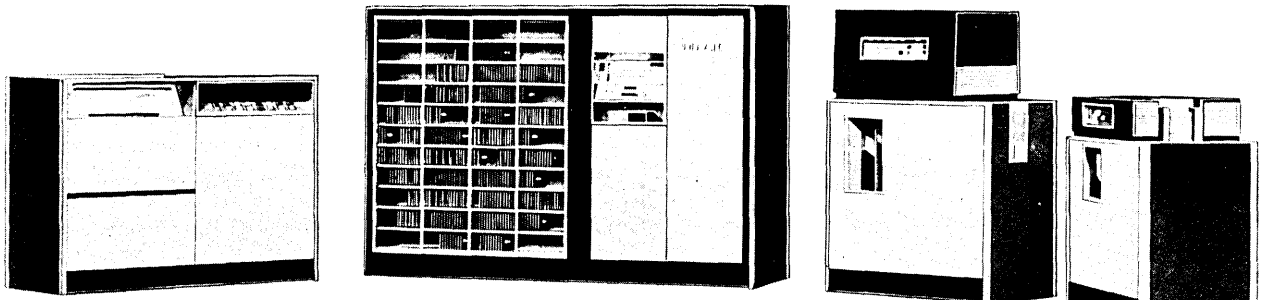
PROGRAMMING

Instruction Codes and Programming Aids

PROGRAMMING

Instruction Codes and Programming Aids

Preface



HIGH-SPEED PRINTER | CENTRAL PROCESSOR | READ-PUNCH UNIT | HIGH-SPEED READER

UNIVAC Solid-State Computers for processing information recorded in 80-column punched-cards are the latest addition to the new line of Remington Rand magnetic amplifier, solid-state data processing systems. With the UNIVAC Solid-State 90 for 90-column punched-card processing already in customer operation, Remington Rand now offers the multiple advantages of solid-state computing systems to every 80 or 90-column punched-card user.

This manual contains a summary of the functional and programming characteristics of the UNIVAC Solid-State 80. To provide a basic reference in programming, particular emphasis has been placed on the programming instructions affecting each of the four units comprising the system.

Techniques of operation, outstanding features of the UNIVAC Solid-State 80, and advanced machine programming methods are not detailed in this manual but are subjects of separate publications.

Contents

4. MINIMUM LATENCY

5. INDEX REGISTER PROGRAMMING

6. TABLES

1. General Description

The UNIVAC Solid-State 80 is a general purpose, digital computing system capable of performing a wide range of data processing tasks requiring large volumes of input-output data. The system is composed of the High-Speed Reader, the Central Processor, the Read-Punch Unit and the High-Speed Printer. Joined by electronic circuitry, the four compact, easily operated units are sure, fast and reliable.

INPUT

Its principal input device is the High-Speed Reader, which provides the system with a constant flow of data read from cards at the rate of 450 cards per minute. Like each of the other units of the system, the High-Speed Reader is under the control of a program that is stored in the Central Processor.

PROCESSING

As well as performing the storage, logical, and arithmetic operations, the Central Processor coordinates the activities of the system making full and simultaneous use of its input, output and processing abilities. Output emerges from this new UNIVAC System in either punched or printed form.

OUTPUT

The Read-Punch Unit accomplishes punching at the rate of 150 cards per minute. In addition to punching output, the unit can be utilized to read data into the system. In fact, reading occurs as an integral part of every punching cycle, even when the unit is operating at maximum speed. Printing is performed on the High-Speed Printer at the rate of 600 lines per minute and may be executed in an almost limitless variety of formats.

HIGH-SPEED READER

Data cards stacked in the input magazine of the High-Speed Reader are fed into continuously revolving rollers that transport them through two reading stations to one of three output stackers. During its course through the transport system, a card conveys its contents to the computer when it is brush-sensed at each of the two read stations in turn. Once inserted between the rollers a card moves without interruption until it reaches its designated stacker.

If it were possible to instantaneously halt this unit in the midst of its 450 cards a minute operation, the following situation would be revealed. To begin with, four cards would be committed to the system. The card labeled A, that was in transit from the input magazine, would be near the rollers, as shown in Figure 1. Depending on the exact stopping point in a cycle, the next two cards, designated B and C, would be either at or approaching each of the reading stations. The progress of the fourth card, labeled D, would have been suspended at some point on its way to an output stacker. Thus, it becomes apparent that the ability to feed, read, and select cards at the same time enables the High-Speed Reader to sustain its input speed at a maximum rate.

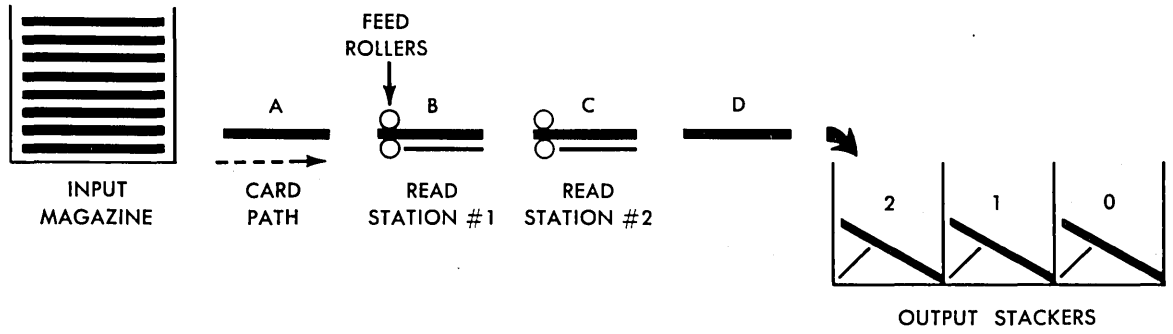


FIGURE 1. Functional Diagram of the Transport Mechanism of the High-Speed Reader

READ-PUNCH UNIT

Since the Read-Punch Unit functions as both an input and output device, it is only logical that its card movement differs from that of the High-Speed Reader. Between the card cycles, except during the initial and final phases of a program, cards are disposed throughout this unit's transport system as follows:

Cards awaiting processing are stacked in the input magazine while five others, one at each station, are engaged as shown in Figure 2. This figure depicts card D at the punch station. Flanking the punch station are two wait stations containing cards C and E. Card C at Wait Station #1 is the one that will receive data. Moreover, Figure 2 shows card B and F at Read Station #1 and #2 respectively.

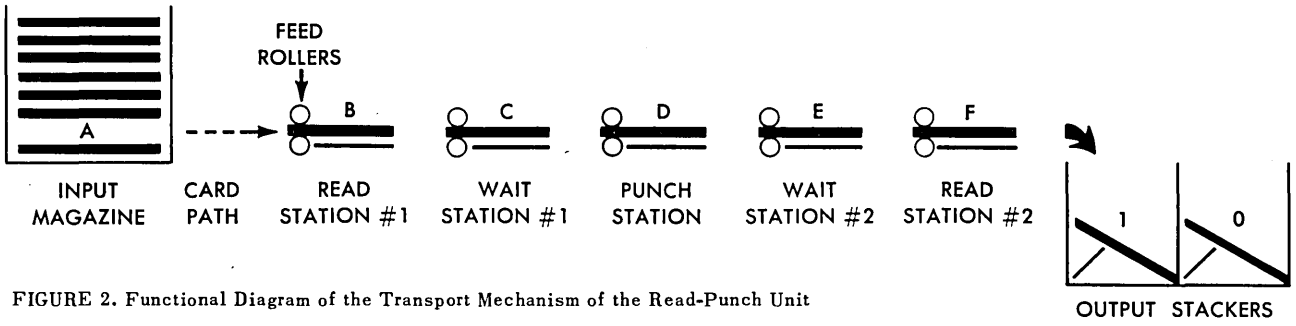


FIGURE 2. Functional Diagram of the Transport Mechanism of the Read-Punch Unit

When a card cycle instruction is given, a series of actions constituting a cycle is automatically performed. All cards engaged in the unit advance one station. Card A is read as it moves from the input magazine to Read Station #1. Card B moves from Read Station #1 to Wait Station #1. Card C moves from Wait Station #1 to the punch station where it is punched upon arrival. Card D moves from the punch station to Wait Station #2. Card E is read as it moves from Wait Station #2 to Read Station #2 and Card F moves from Read Station #2 to the selected output stacker. Figure 3 shows all the cards advanced at the completion of a card punching cycle.

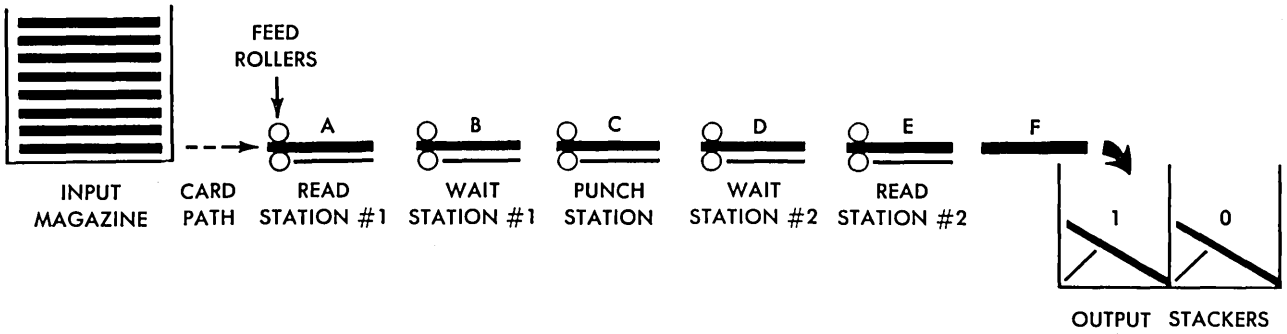


FIGURE 3. Functional Diagram of the Transport Mechanism of the Read-Punch Unit

HIGH-SPEED PRINTER

The High-Speed Printer is the other output mechanism of the UNIVAC Solid-State 80 System and is capable of printing large volumes of computed or tabulated information at a maximum rate of 600 lines per minute. Since each line has 130 print positions that can be occupied by any combination of characters selected from the letters of the alphabet, the numbers 0 through 9, and 15 special symbols, virtually unlimited line flexibility can be achieved. Six lines per inch vertically and 10 characters per inch horizontally can be legibly printed. The length of the line as well as the spacing between lines is completely variable. In addition, the paper feed will handle any sprocket-fed paper up to and including card stock, either blank or printed, from 4 to 21 inches in overall width. At least five carbon copies can be made by using paper between 11 and 13.5 pounds in weight. The printer will stop automatically on detection of a low paper supply. Moreover, impression control permits variation in the strength of the printing hammer stroke. And finally, fine vertical adjustments of the paper position may be made while the printer is in operation.

CENTRAL PROCESSOR

Under the guidance of an internally stored program, the Central Processor controls the many interdependent activities of this UNIVAC Solid-State 80 System and houses the main storage area as well. Adhering to the instructions in the control unit of the Central Processor, logical circuitry accepts data from the input units, transfers it between the arithmetic and storage components, and transmits processed data to the output units. Similarly, the control unit governs the arithmetic elements in the performance of such operations as addition, subtraction, multiplication, division, and logical comparisons.

Related directly to the control unit is the operator's control panel and keyboard, both of which are located on the Central Processor. By means of the panel and keyboard, the system's operations can be interrupted for the insertion of data or the display of the contents of addressable registers or of any storage location. The design of the control panel and keyboard reduces to a few simple steps the formerly complex procedures of starting, intervention, error-detection and corrective action.

Along with this thorough control, the Central Processor provides storage for 50,000 digits on its magnetic drum. Experience has proven that storage of such magnitude is able to accommodate large volumes of data and instructions while still reserving ample room for tables created during operation.

MAIN STORAGE

The main storage area, located on the high-speed (17,670 rpm) magnetic drum is divided into a high-speed and a standard access section as shown in Figure 4. These segments are further subdivided into bands. Every band in main storage is designed to accommodate 200 computer words, each consisting of 10 digits plus sign. Bands one through 20, whose maximum access time is 3.4 milliseconds, comprise the standard access portion of the drum. The high-speed access sector, on the other hand, is composed of five bands whose maximum access time is .85 milliseconds. Standard access bands are equipped with one read-write head per band, while the high-speed access bands are serviced by four read-write heads spaced at 90 degree intervals around the circumference of the bands. Since each storage band contains 200 computer words, the difference in access time between the high-speed and standard sectors is accounted for by the number of read-write heads that service a band. Access time is important because it influences the programmer's choice of storage locations for both instructions and data.

BUFFER STORAGE

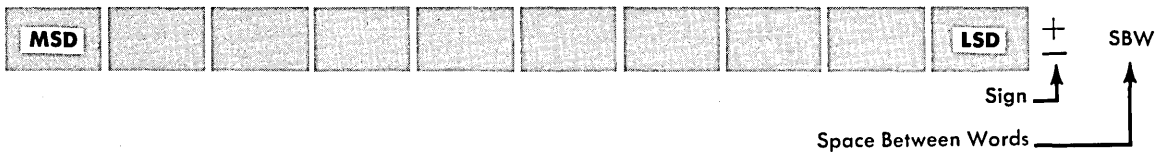
Buffer storage, also located on the high-speed drum, compensates for the disparity between the Central Processor's electronic speed of computation and the mechanical speed of the input-output units. There are six buffers, two for the High-Speed Reader, three for the Read-Punch Unit, and one for the High-Speed Printer. All input and output information flowing from main storage to the external units and vice versa must pass through these buffers. In this way, with the buffers acting as an intermediary, the main storage never becomes involved with the external units. Instead, the main storage communicates only with the buffers which transmit or receive information at electronic speeds. Another advantage of the buffer component springs from the fact that it can send data to, or receive data from external units automatically, thus leaving the computer control section free to continue directing the processing of data.

TIMING BAND

The timing band, also located on the magnetic drum, synchronizes the operations of the system by sending various types of timing pulses to the control, arithmetic, and input-output units. This band also controls the addressing of storage locations.

THE WORD CONCEPT

Information, as already mentioned, is stored on the drum in increments of ten-digit words (plus sign) as shown below. Word time for the main storage area is .017 milliseconds.



Three computer words are required to accommodate a 10 column group, known as a card word, when it is in the system in 80-column card code. However, if this data in 80-column card code is translated to computer code, it occupies only two computer words. This concept will be developed in greater detail in the discussion of 80-column punched-cards.

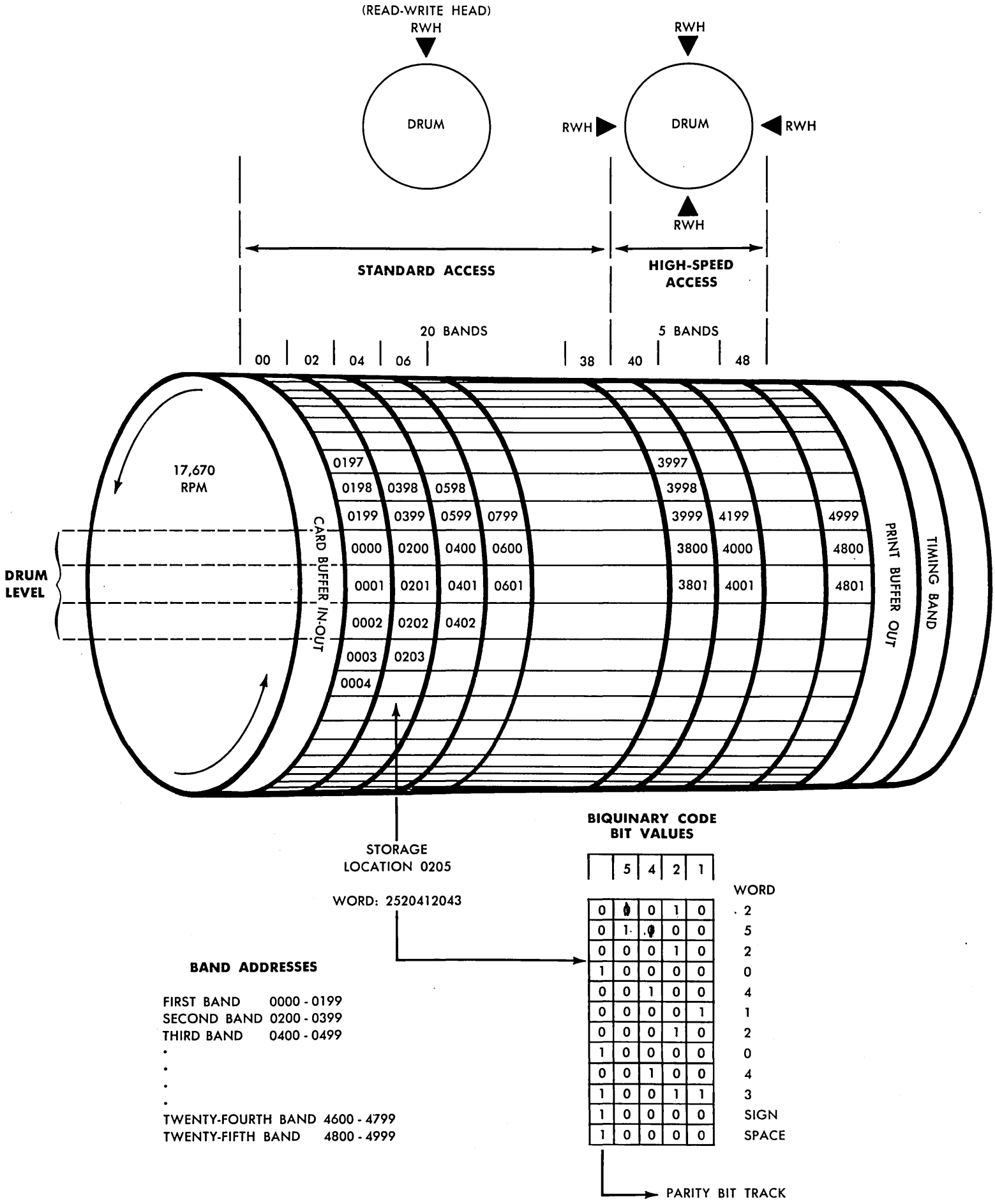


FIGURE 4. Magnetic Storage Drum

BINARY REPRESENTATION

Each character position of a computer word contains five binary positions in which the presence or absence of a bit has significance. Four of these bit positions represent a value in the 5 4 2 1 binary-coded decimal code. The fifth bit is a check bit and is assigned by the computer in such a way that the total number of binary "1's" in a digit is an odd number. The computer binary codes for each decimal digit or alphabetic character are shown in the Translation Table in the last two columns headed (Zone) and (Numeric.)

METHOD OF STORAGE

Starting with the most significant digit of a word, the pattern of bits representing each digit plus parity bit is recorded in the form of magnetized spots on the surface of the drum as it revolves beneath the read-write heads. At the instant of writing, five bits of a digit are entered in parallel across a band, each being deposited on a *track*, which is merely a further vertical subdivision of a band. The remainder of the word is written on the drum in the same manner—serially; that is, digit by digit, five bits at a time. The sign designation and space between words are also entered as part of the word.

PERMANENT STORAGE

Any magnetized spot recorded on the drum will remain there permanently or until it is erased for the recording of another spot on the same location. The system may be turned off completely without losing the magnetized spots.

ONE-WORD REGISTERS

The UNIVAC Solid-State 80 contains four one-word registers; rA, rX, rL, and rC. The first three registers hold data or instructions to be operated upon by the arithmetic or control circuits of the computer. The last register, rC, holds the instruction being executed.

INDEX REGISTERS

The Central Processor is equipped with three Index Registers: rB_1, rB_2, and rB_3, each having a four-digit capacity. The Index Registers may be loaded with an increment which is then used, as the program directs, to modify the operand address of instructions before their execution.

2. The 80-Column Punched Card

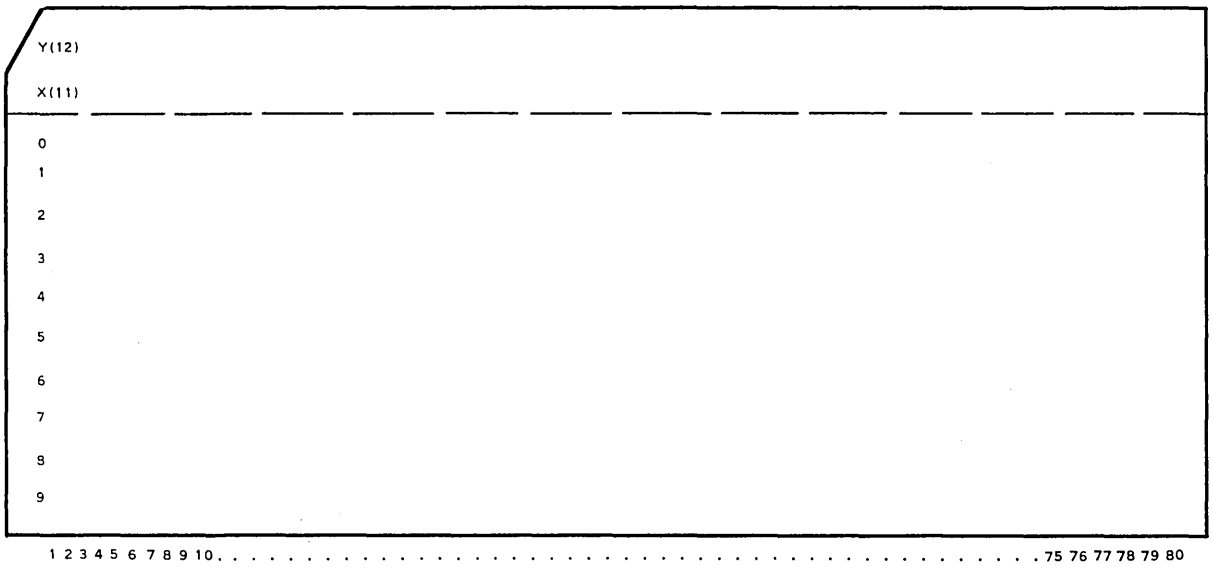


FIGURE 5. The 80-Column Punched Card

THE CARD

The 80 vertical columns of an 80-column punched-card are divided into two sections, one labeled with digits 0 through 9 (below the dotted line in Figure 5), and the other labeled 11 and 12 (above the dotted line). The 11 and 12 positions are referred to as the X and Y zone punches respectively. A group of columns signifying a unit of information, such as employee badge number or employee name in a payroll application, is known as a "field." The number of columns comprising a field varies with the type of information represented.

REPRESENTING NUMERIC AND ALPHABETIC CHARACTERS

On the card a decimal digit is represented by a single hole punched in a position of the 0 through 9 section corresponding to the digit. Two punches in a single vertical column are necessary to denote alphabetic characters. One of the punches is an X, Y, or 0 zone punch while the other is a digit from 1 through 9. Figure 6 illustrates how alphabetic symbols are punched in card code.

An examination of Figure 6 (on the following page) reveals that the letters A through I are represented by a 12 (Y) punch and a 1 through 9 punch respectively. Letters J through R are shown to be combinations of an 11(X) punch and a 1 through 9 punch, and letters S through Z respectively, of a 0 with a 2 through 9 punch.

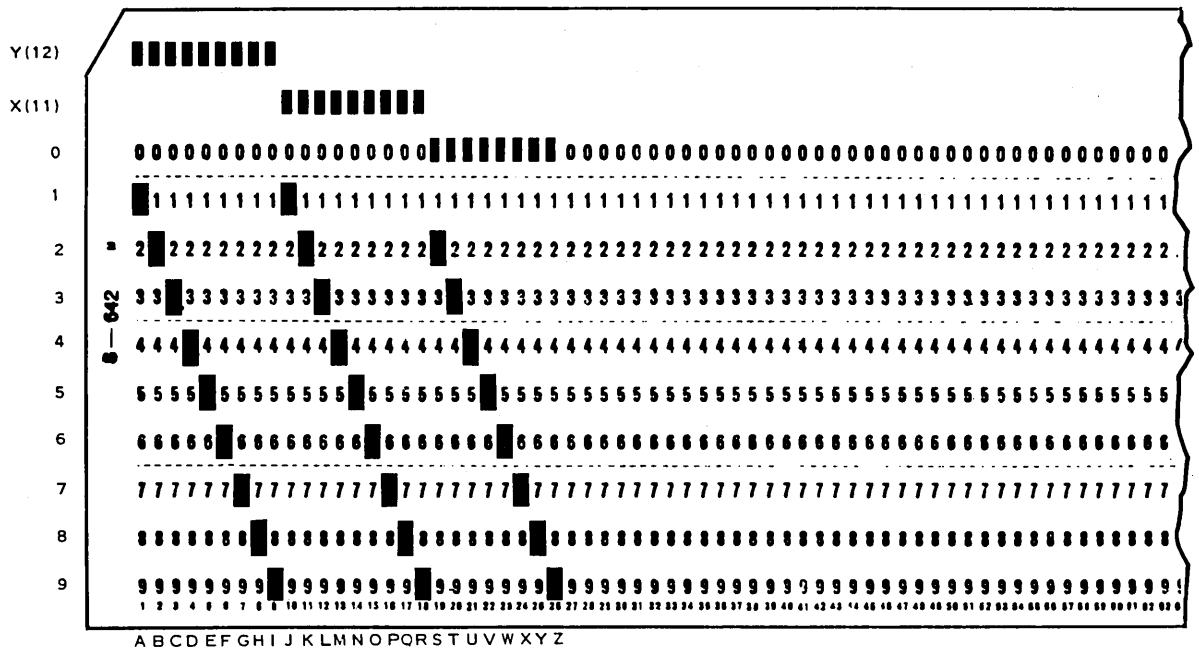


FIGURE 6. Alphabetic Character Representation

CARD CODE TO COMPUTER CODE

Although information enters the buffers in 80-column card code, it can only be transferred and compared for equality in this form. To be compared for magnitude or to be worked on arithmetically, information must first be translated to computer code. The translation may be performed automatically as the information leaves the buffer and is deposited in storage. Or it may be read into storage from the buffer in 80-column card code and translated later.

In 80-column card code, the data from a ten-column card group occupies three computer words. The first digit position of all three words contains the information from the first column; the second digit position, the information from the second column; and so on.

The computer word which represents the card positions as 12 (Y), 11 (X), 0, 1 is called the unprimed word. The word representing the positions 2, 3, 4, 5 is termed the primed word, and the word representing the positions 6, 7, 8, 9 is termed the duo-primed word. Figure 7 depicts this relationship as it applies to the numbers 1 through 9 and 0 punched in card code. It shows these numbers recorded in the first 10 columns of the card and their appearance in three computer words in card code.

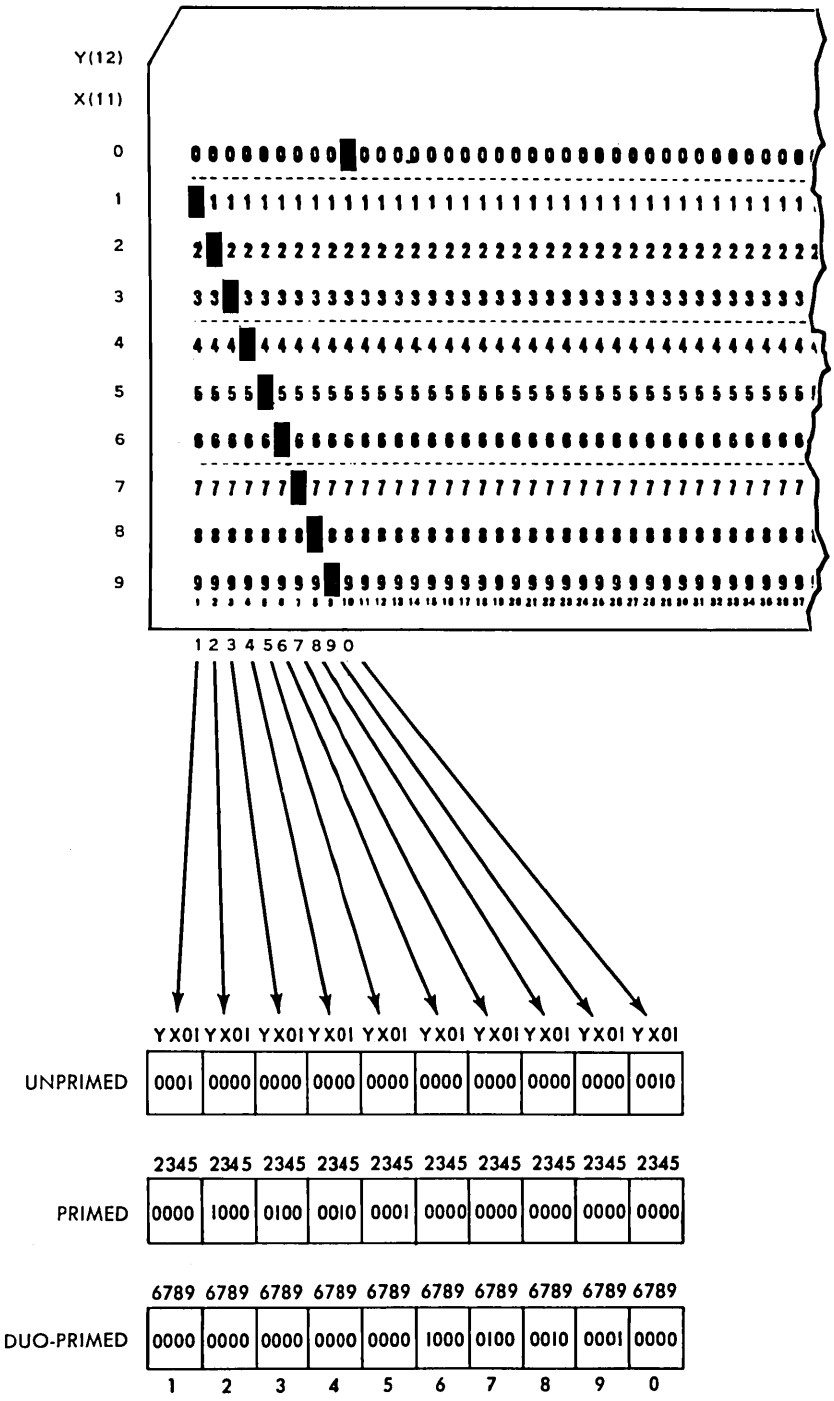


FIGURE 7. Numeric Representation on the Card and in the Computer in Card Code

It can be seen from Figure 7 above and Figure 8 on the following page that both the letters of the alphabet and the numbers 1 through 9 and 0 appearing on the card in *card code* are represented in unprimed, primed and duo-primed digits. The four bits of each unprimed digit from left to right, denote punch positions Y, X, 0, 1. The bits of the primed digit represent, from left to right,punch positions 2, 3, 4, 5; and the digits of the duo-primed portion from left to right signify punch positions 6, 7, 8, 9.

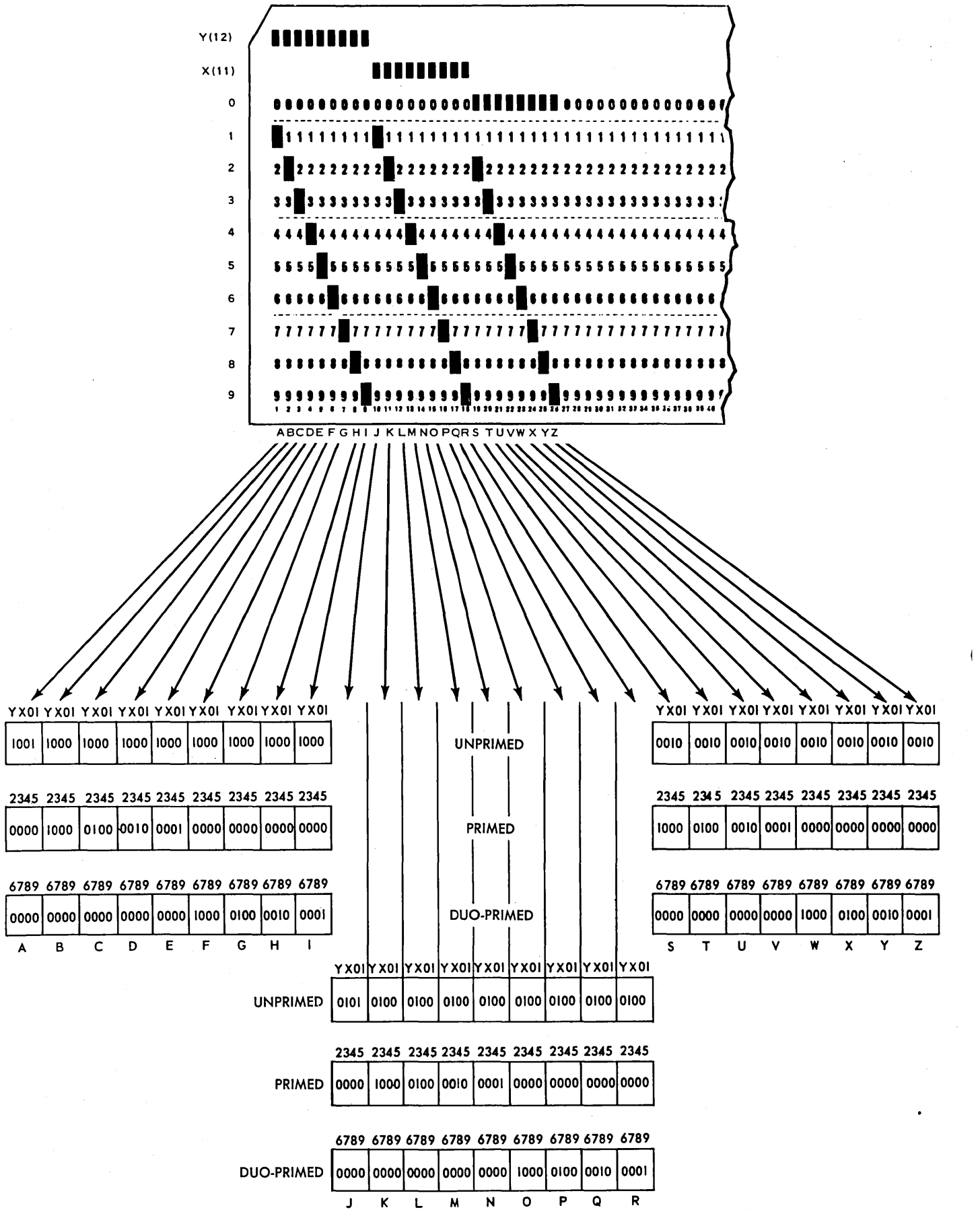


FIGURE 8. Alphabetic Representation on the Card and in the Computer in Card Code

When information coded in 80-column card code is translated to the modified biquinary that is the computer code for the UNIVAC Solid-State 80, two computer words are needed to represent a ten-column card group. The first word is called the numeric word, and the second, the zone word. The bit positions of both these words represent from left to right the value 5 4 2 1. With card-coded data that is numbers only, the numeric word is significant but the zone word contains no bits. Consequently, the numbers, 1 through 9 and 0 would be represented in one computer word in *computer code* as follows:

	5421	5421	5421	5421	5421	5421	5421	5421	5421	5421
NUMERIC	0001	0010	0011	0100	1000	1001	1010	1011	1100	0000
	1	2	3	4	5	6	7	8	9	0

Alphabetic information when translated from 80-column to computer code requires the presence of bits in both the numeric and the zone words. Hence the alphabet appears in *computer code* as follows:

	5421	5421	5421	5421	5421	5421	5421	5421	5421	5421
NUMERIC	0001	0010	0011	0100	1000	1001	1010	1011	1100	0001
ZONE	0001	0001	0001	0001	0001	0001	0001	0001	0001	0010
	A	B	C	D	E	F	G	H	I	J

	5421	5421	5421	5421	5421	5421	5421	5421	5421	5421
NUMERIC	0010	0011	0100	1000	1001	1010	1011	1100	0010	0011
ZONE	0010	0010	0010	0010	0010	0010	0010	0010	0011	0011
	K	L	M	N	O	P	Q	R	S	T

	5421	5421	5421	5421	5421	5421				
NUMERIC	0100	1000	1001	1010	1011	1100				
ZONE	0011	0011	0011	0011	0011	0011				
	U	V	W	X	Y	Z				

Certainly then, both the numeric and zone words must be considered in operations using computer-coded alphabetic information. Naturally, when a card field containing both numbers and letters is translated to computer code, both the numeric and zone portions are meaningful.

SPECIAL SYMBOLS

Special symbols appear in the computer in *card code* as:

	YX01	YX01	YX01	YX01	YX01	YX01	YX01	YX01	YX01	YX01
UNPRIMED	1000	0100	0100	0000	0011	0100	0010	1000	0010	0000
	2345	2345	2345	2345	2345	2345	2345	2345	2345	2345
PRIMED	0100	0100	0000	0100	0000	0010	0100	0000	0010	0000
	6789	6789	6789	6789	6789	6789	6789	6789	6789	6789
DUO-PRIMED	0010	0010	0000	0010	0000	0010	0010	0000	0010	0000
	.	$	-	#	/	*	,	&	%	Δ

The same symbols appear in *computer code* as:

	5421	5421	5421	5421	5421	5421	5421	5421	5421	5421
NUMERIC	0101	0101	0101	1111	0001	0110	0101	0111	0110	0110
	5421	5421	5421	5421	5421	5421	5421	5421	5421	5421
ZONE	0001	0010	0000	0001	0011	0010	0011	0001	0011	0000
	.	$	-	#	/	*	,	&	%	Δ

Other symbols appear in *computer code* as follows:

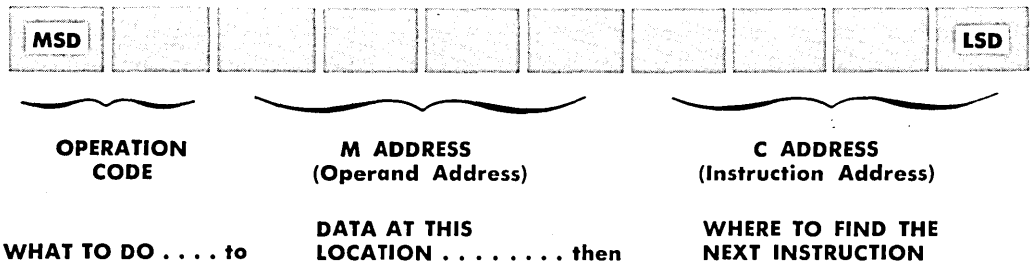
	5421	5421	5421	5421	5421	5421	5421	5421	5421	5421	5421
NUMERIC	0110	0000	1110	1111	1101	0111					
	5421	5421	5421	5421	5421	5421	5421	5421	5421	5421	5421
ZONE	0001	0011	0000	0000	0000	0000					
	:	+	;	'	(	)					

These and other symbols are found in the Translation Table at the end of this manual.

3. Instructions

GENERAL

The UNIVAC Solid-State 80 Computer employs a one and one-half address instruction code system with one instruction per computer word. The format of an instruction word is illustrated below:



The m address is usually the address of a word in storage. The operation code tells the computer what to do with this word; the c address is the storage location of the next instruction word. These fields may have different significance for some special instructions, as noted in the instruction definitions.

The m address of instructions can be modified through the use of an Index Register. Such modification adds one word time to the instruction cycle.

When a word is transferred from a storage location or register, the contents of the storage location or register from which the word was transferred remain unchanged.

When a word is transferred into a storage location or register, the previous contents of the storage location or register are erased, except in the 20 and 35 instructions.

INSTRUCTION CYCLE

A three or four-phase cycle (with an additional step if Index Registers are used) is associated with each instruction depending upon whether an operand is required from drum storage. If setting-up the instruction is considered the starting point, the instruction cycle is:

Staticize the Instruction	Index Register Modification (if any)	Search for the Operand	Execute the Instruction	Search for the Next Instruction
(1)	(2)	(3)	(4)	(5)

(1) STATICIZE THE INSTRUCTION: The instruction located by the previous search (5) is transferred from the drum location to the Static Register (operation code only) and register C (the entire word). This phase requires one word time, which is .017 milliseconds.

(2) INDEX REGISTER MODIFICATION: When modification is indicated, the m address of the instruction is altered by the Index Register specified. This phase requires one word time.

(3) SEARCH FOR THE OPERAND: If the first address part -- the m address -- of the instruction refers to a drum storage location, the address of the next available storage location on the drum is compared with the first address part of the contents of register C every word time until a match is obtained. Register C contains the entire instruction. If an operand is required from storage this phase requires a minimum of one word time and a maximum of 200 word times. Register A, X or L may be the operand address of most instructions.

(4) EXECUTE THE INSTRUCTION: The operation indicated in the instruction is performed. The time required depends upon the type of operation to be performed.

(5) SEARCH FOR THE NEXT INSTRUCTION: Every word time the address of the next available storage location on the drum is compared with the second address part -- the c address -- of the contents of register C until a match is obtained. This phase requires a minimum of one word time.

ADDRESS MODIFICATION WITH INDEX REGISTERS

If an instruction is to be modified by the use of Index Registers, it must contain an indication of the Index Register to be used. The indication is a combination of the sign bit and the second or least significant digit of the operation code. Only the 4 bit of the 5 4 2 1 bits of this digit is used. Since this bit is not utilized in the least significant digit of any operation code, the normal execution of an instruction is not otherwise affected. The Index Register must be loaded initially with the desired increment.

The four combinations of the two bit-positions are interpreted as shown in the table below:

MODIFICATION TABLE

Sign Bit	0 5421	C 5421	Condition	Type of Modification
0	XXXX	X0XX	0	No modification
0	XXXX	X1XX	1	Modify with Index Register 1 (rB_1)
1	XXXX	X0XX	2	Modify with Index Register 2 (rB_2)
1	XXXX	X1XX	3	Modify with Index Register 3 (rB_3)

Normally the programmer is not concerned with these indications when he writes instructions. The specification is made in the translation key and is handled automatically by the load routine.

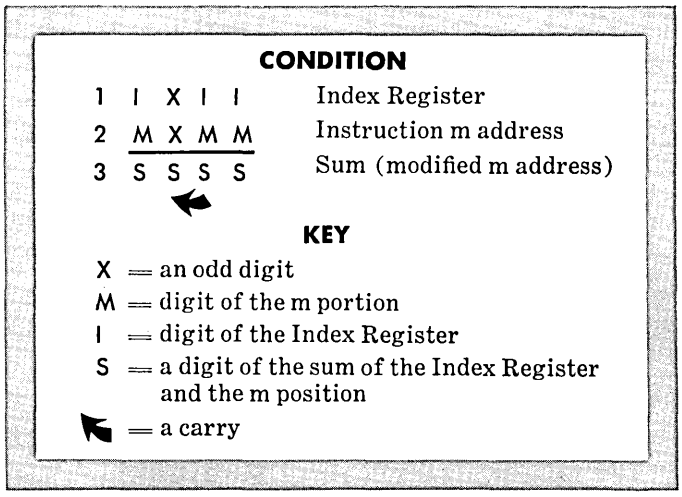
During the staticizing phase all instructions are examined for the presence of one of the above conditions. If condition 0 is found to be present, the modification phase is bypassed, and the instruction is executed in the normal manner with a three- or four-phase cycle. If condition 1, 2, or 3 exists, however, the second phase of the instruction cycle is the addition of the specified Index Register's contents to the m address of the contents of register C. The computer then searches for the new m address during the third phase, and the execution of instructions proceeds in the normal manner.

BAND MODIFICATION FEATURE

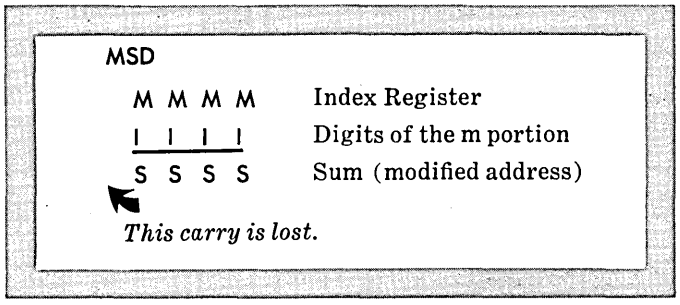
A band modification feature enables modification with Index Registers to be restricted to the same band in which it was initiated. If, during modification, the m address is altered to refer to another band, the computer will cause a return to the corresponding location in the original band. This band modification occurs when at least two of the following conditions are present:

1. The hundreds digit of the contents of the Index Register is odd.
2. The hundreds digit of the instruction's m address is odd.
3. During modification a carry occurs from the tens digit to the hundreds digit.

The figure below represents the conditions which, in combination, cause band modification to be effected.



Note: When the four digits of the m address are added to those of the Index Register, any carry from the most significant digit position is lost.



ADDRESSABILITY OF REGISTERS

As a result of their addressability, registers A, X, or L may be the m address of many, and the c address of almost all instructions. The sole restriction is that they cannot be used in the m portion of the 50, 60, and 65 orders, nor in the m portion of instructions which do *not* address a one-word storage location. The arithmetic registers are addressed by non-numeric digits in the least significant digit of c or m.

	LSD of m or c
BITS	DIGIT
rA = 0101	1/4
rX = 0111	3/4
rL = 0110	2/4

INSTRUCTION REPERTOIRE

The timing for each instruction is shown in the right-hand column following the description of the instruction. Timing is shown as the number of word times required for the instruction cycle in minimum latency. One extra word time should be added when Index Register modification is to be accomplished. Timing in milliseconds can be obtained by multiplying the word time given by 0.017.

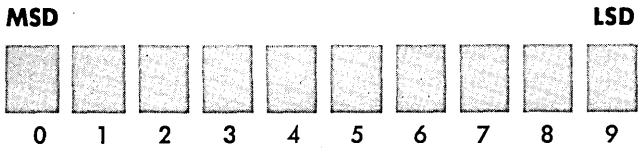
The following pages contain descriptions of the instructions used with the UNIVAC Solid-State 80 System. In the descriptions, the following conventions are used:

- m represents the address of a storage location or register which usually contains the operand
- c represents the address of a storage location or register which usually contains the next instruction
- (m) represents the contents of a storage location or register
- rA represents register A
- rL represents register L
- rC represents register C
- rX represents register X

A dash substituted for the m or c portion of an instruction means the computer ignores that portion when the instruction is executed.

ARITHMETIC INSTRUCTIONS

		Word Time
70 m c	Add algebraically (m) to (rA) and place the sum in rA.	5
75 m c	Subract algebraically (m) from (rA) and store the difference in rA.	5
85 m c	Multiply (rL) by (m) and store the 10 most significant digits of the product in rA and the 10 least significant digits in rX. Both rA and rX will have the sign of the product. Multiplication can be shortened for multipliers having less than 10 significant digits by placing a sentinel just to the left of the most significant digit of the multiplier, m. This sentinel stops the multiplication after the last significant multiplier digit is used. Sentinel (0101 or 1101).	5 plus the number of digits in the multiplier plus the sum of these digits.
55 m c	Divide (m) by (rL). The quotient with sign is placed in rA unrounded, and the unrounded remainder is placed in rX. If the divisor is zero, equal to, or less than the dividend, overflow occurs. A sentinel should be placed in rX to stop the division as soon as the desired number of digits has been developed in the quotient. For the placement of the sentinel, the digit positions are numbered, *from left to right, 0 through 9*, instead of the usual way (1 through 10).	5 + 2 times the number of digits in the quotient + the sum of the odd digit-positions in the quotient + the sum of the nines complement of the even digit-positions in the quotient.



Digit position for division sentinel only

The sentinel (0101) is placed as follows:

Purpose	Position
To develop 2 digits in the quotient 2	All other positions must contain zeros
To develop 4 digits in the quotient 4	
To develop 6 digits in the quotient 6	
To develop 8 digits in the quotient 8	

To develop 10 digits in the quotient, the contents of rX need not be changed, providing that the non-numeric combination (0101) is not present in rX. If there is a possibility that the combination is present, the program should fill rX with zeros.

Note: If an overflow occurs during the add, subtract or divide operations, the next instruction is obtained from storage location c + 1. If the c address is on drum level 199, overflow will cause control to revert to the instruction on level 000 of the same band. Also, if an arithmetic register is used as the c address of an instruction which happens to cause overflow, the next instruction is still taken from the addressed register—after a delay of 1 word time.

TRANSFER INSTRUCTIONS

		Word Time
25 m c	Transfer (m) to rA	4
*60 m c	Transfer (rA) to m.	4
05 m c	Transfer (m) to rX.	4
*65 m c	Transfer (rX) to m.	4
30 m c	Transfer (m) to rL.	4
*50 m c	Transfer (rL) to m.	4
77 – c	Transfer (rA) to rL .	3
06 m –	Clear rX to zero and set its sign storage to plus. Ignore c. Next instruction at m.	3
26 m –	Clear rA to zero and set its sign storage to plus. Ignore c. Next instruction at m.	3
31 m –	Clear rL to zero and set its sign storage to plus. Ignore c. Next instruction at m.	3
23 m –	The contents of rC are transferred to rA. The next instruction is at m. This instruction bypasses the buffer test for the Read-Punch Unit, and the interlocks are ignored.	3

"m" may *not* be register A, X, or L.

LOGICAL INSTRUCTIONS

		Word Time
BUFF		
20 m c	Superimpose or buff the 1 bits of (m) onto (rA) and leave the result in rA. The sign of rA is undisturbed.	4
ERASE		
35 m c	Change the bits in each digit position of (rA) to binary zero wherever (m) has a 0 bit in the corresponding bit position. The sign of rA is undisturbed.	4
36 m -	Clear (rA) to zero. The sign is undisturbed. The next instruction is in storage location m.	3
86 m -	Clear rA and rX to zero. The signs of rA and rX assume the sign of rL.	14

Note: In the following shift instructions, the m has been replaced by onoo. For these instructions the n is always the second most significant digit of m portion.

RIGHT CIRCULAR SHIFT

32 onoo c	Shift (rA) n digit positions to the right into rX the contents of which are also shifted, (n digit positions) to the right into rA. The sign is undisturbed. Index Register modification is not effective.	3 + n

LEFT SHIFT

37 onoo c	Shift (rA) n digit positions to the left losing the most significant digits and bringing in zeros in the least significant digit positions on the right. The sign is undisturbed. Index Register modification is not effective.	3 + n

SKIP

00 m -	Skip to next instruction at address m.	2

EQUALITY COMPARISON

82 m c	Compare (rA) and (rL) . If they are equal, transfer control to m; if they are unequal, program control passes to c.	3

MAGNITUDE COMPARISON

87 m c	Compare (rA) and (rL) . If (rA) is algebraically greater than (rL) , the next instruction is in m; if not, the next instruction is in c.	3

STOP

67 m c	Stop the computer. The computer stops with the stop instruction in rC, but before the search for the next instruction is started. Normally, when the computer is restarted, the first step will be to search for the next instruction specified by the c address. In this case the m digits are ignored, and may be used as a code to indicate the reason for stopping. However, if desired, the m address may be used as an alternate restart location by depressing the "m" button on the control panel.	–

LOAD INDEX REGISTER

02 m c	Transfer the four digits comprising m into the specified Index Register. The same combination (see note) which indicates normal Index Register modification specifies the Index Register to be loaded.	3

INCREMENT AND UNLOAD INDEX REGISTER

07 m c	Using the address modification method, add the four digits comprising m to the contents of the specified Index Register. The sum is placed in the Index Register specified and also in the m portion of rA. The remainder of rA is cleared to zeros, and the sign of rA is set to plus.	4

Note: These two instructions will serve all three Index Registers. The same combination which indicates normal modification with Index Registers is used to specify the Index Register to be used. This combination of the sign bit and the 4 bit of the operation code's least significant digit has been explained under "Address Modification with Index Registers."

TRANSLATION INSTRUCTIONS

		Word Time
12 - c	Translate from card code to computer code. Before this command is given, rA must contain the unprimed word; rL, the primed word, and rX, the duo-primed word of the field to be translated. After the command is executed, rA will contain the numeric, and rX, the zone word in computer code. The signs are unchanged.	3
17 - c	Translate from computer code to card code. Before the command is given, rA must contain the numeric, and rX, the zone word in computer code. After the command has been executed, the unprimed word of the card code is in rA; the primed word, in rL, and the duo-primed word, in rX. Translation is effective for all alphanumeric character codes. The signs of rA and rX are positive, and the sign of rL is unchanged.	3

Note: Translation affects all alphabetic, numeric, and special characters (63 characters).

INPUT INSTRUCTIONS (High-Speed Reader)

Figure 9 shows a functional diagram of the High-Speed Reader's transport mechanism and the high-speed magnetic drum, focusing on the relation between the read stations and the buffer area.

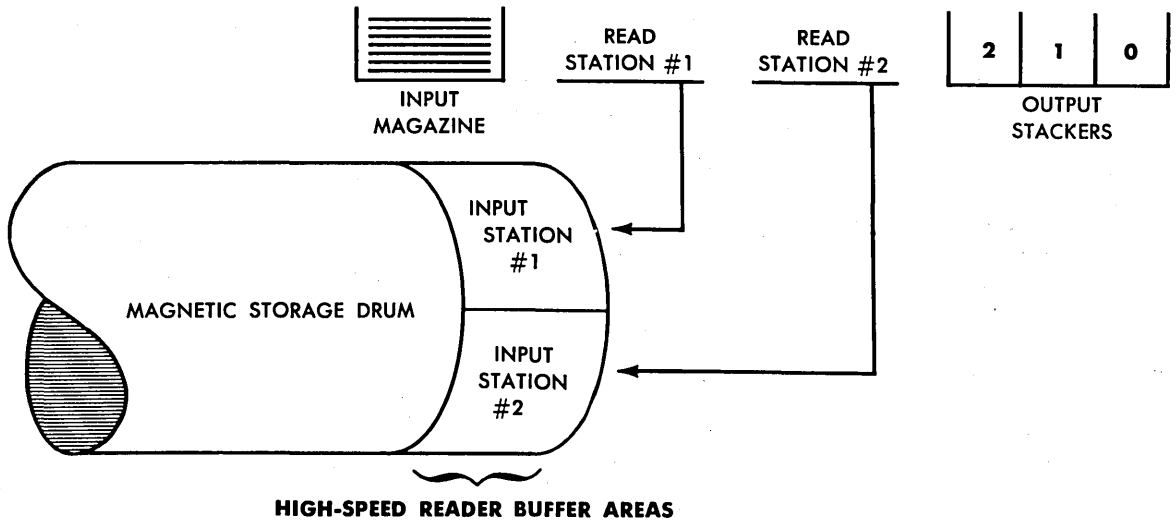


FIGURE 9. Relationship of Read Stations to Buffer

Cards are read at both stations simultaneously, automatically conveying their contents into the High-Speed Reader Buffer. The card images are not entered in sequential word positions on the buffer band, but are recorded in a fixed pattern called the High-Speed Reader Buffer Interlace. Five instructions direct the operation of the High-Speed Reader:

HIGH-SPEED READER CARD CYCLE

		Word Time
72 m c	This instruction initiates card movement in the High-Speed Reader, by placing a card between the continuously revolving feed rollers. The card will be read at each station, in turn, and the data stored in the buffer band. The next instruction is at c. The computer is free to operate on other instructions during the moving and reading of cards. However, if a 72 instruction is given before the preceding 72 has begun to feed a card, the second 72 is not executed and (rC) go to rA. The next instruction is found at m.	3 - if executed 4 - if not

BUFFER TEST

42 m c	Test the High-Speed Reader Buffer. If it is loaded, (rC) go to rA. The next instruction is at m. If the buffer is not loaded, control is transferred to c and rA is not altered.	3 if c address taken, otherwise 4

BUFFER UNLOAD

96 $\underbrace{XX00}_{m}$ c	Transfer the contents of the High-Speed Reader Buffer into the storage band according to its predetermined interlace pattern. The data is transferred untranslated. The two most significant digits of m (XX) designate the storage band.	203

BUFFER UNLOAD AND TRANSLATE

96 $\underbrace{XX01}_{m}$ c	Unload the contents of the High-Speed Reader Buffer into the storage band designated by the two most significant digits of m. During the transfer, translate the data (as in the 12 instruction) to computer code. Only the two most significant digits of m (XX) denote the storage band to which the data is transferred. The two least significant digits of m must be 01.	215

Note: A special card image interlace pattern will be formed in storage if automatic translation is used.

STACKER SELECTION

47 $\underbrace{onoo}_{m}$ c	Select the output stacker on the High-Speed Reader, (n = stacker 0,1, or 2). Timing: To operate on the card at the second read station, the instruction must be given within 120.8 milliseconds after the image is available in the buffer. If not, the card will enter the previously selected stacker.	3

INPUT-OUTPUT INSTRUCTIONS (Read-Punch Unit)

Figure 10 (on the following page) depicts the functional relation between the Read-Punch Unit's transport system and the Read-Punch Buffers located on the high-speed drum.

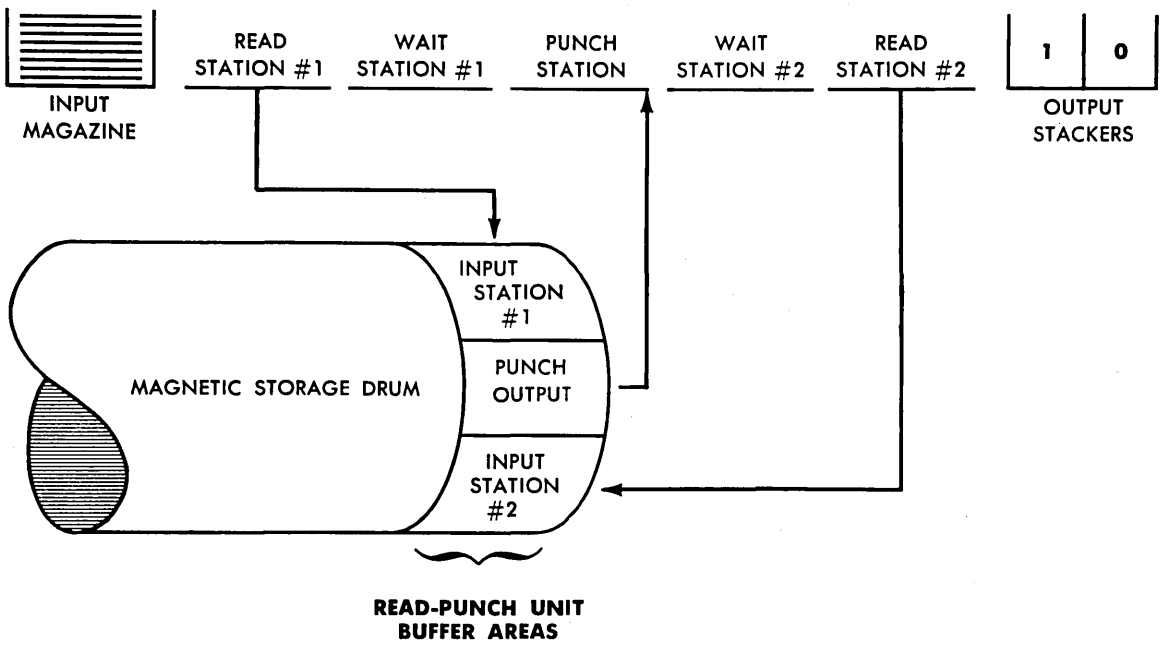


FIGURE 10. Relationship of Punch, Read Stations to Buffer

During a card cycle the card images read at the first and second read stations are simultaneously recorded on the Read-Punch Buffer. As in the High-Speed Reader, this information enters in conformance with the fixed interlace pattern of the proper buffers. Similarly, a fixed interlace pattern in the Punch Buffer will always contain the data to be punched. Six instructions direct the operation of the Read-Punch Unit:

BUFFER TEST

22 m c	Test the Read-Punch Input Buffer. If it is loaded, place (rC) in rA, and the next instruction is at m. If the buffer is not loaded, control is transferred to address c, where the next instruction is found. Register A is unaltered.	3 if c address taken, otherwise, 4.

BUFFER UNLOAD

46 $\underbrace{\text{XX00}}_{\text{m}}$ C	Transfer the contents of the Read-Punch Input Buffer into the storage band designated by the two most significant digits of m (XX). The data is not translated. In storage the data is arranged according to an interlace pattern.	203

BUFFER UNLOAD AND TRANSLATE — Word Time

46 $\underbrace{XX01}_{m}$ c — Unload the contents of the Read-Punch Input Buffer into the storage band designated by the two most significant digits of m. During the transfer, translate the data (as in the 12 instruction) to computer code. Only the two most significant digits of m (XX) denote the storage band to which the data is transferred, but the two least significant digits of m must be 01. — 215

Note: A special card image interlace pattern will be formed in storage if automatic translation is used.

CARD CYCLE - BUFFER LOAD

81 $\underbrace{XX00}_{m}$ c — Initiate card movement in the Read-Punch Unit and a storage to buffer transfer. Data is transferred unchanged. That is, the output card image (which should have been previously arranged in storage according to the interlace pattern) is transferred from the m band to the Punch Buffer. The Read-Punch Unit will then advance the cards one station. During the movement two cards are brush-sensed, storing their images in the Read-Punch Input Buffer in the read interlace pattern. When all the cards have been advanced one station, the Read-Punch Unit punches the data from the Punch Buffer Area, completing the punch cycle. The two most significant digits of the m address designate the storage band. — 103

CARD CYCLE – TRANSLATE

81 $\underbrace{XX01}_{m}$ c — Initiate card movement in the Read-Punch Unit and a storage to buffer transfer. Data is translated during the transfer. That is, the output card images (which should have been previously arranged in storage according to the interlace pattern) are transferred from the m band to the Punch Buffer. They are translated en route. The Read-Punch Unit will then advance the cards one station. During the movement, two cards are brush-sensed, their image being stored in the Read-Punch Input Buffer in the read interlace pattern. When all the cards have been advanced one station, the Read-Punch Unit completes the punch cycle by punching the data from the Punch Buffer. Only the two most significant digits of m (XX) designate the storage band from which the data is transferred, but the two least significant digits of m must be 01. — 210

Note: A special card image interlace pattern will be formed in storage if automatic translation is used.

STACKER SELECT

57 – c — Select Output Stacker #1 in the Read-Punch Unit. This instruction must be given within 116 milliseconds after the Read-Punch Input Buffer is loaded if it is to operate on the card at the second read station; otherwise stacker 0 is automatically selected. The m portion is ignored. — 3

OUTPUT INSTRUCTIONS (High-Speed Printer)

Word Time

PRINTER TEST

27 m c — If the Printer is not busy, (rC) go to rA, and control is transferred to address m. If the Printer is busy, control is transferred to c and rA remains unaltered. — 3 if c address is taken, otherwise 4.

ADVANCE PAPER

16 $\underbrace{oonn}_{m}$ c — Wait until the previous printer operation is completed. Then advance the paper in the Printer the number of lines indicated by the two least significant digits of m (nn) , which can vary from 00 to 79. If the value of nn is in the fifties, sixties, or seventies a *binary* 5, 6, or 7 is used in place of biquinary. Once the paper movement is begun, the computer is free for other operations. — 4

ADVANCE AND PRINT

11 $\underbrace{XXnn}_{m}$ c — Advance the paper nn lines (nn = 00 through 79) and print one line. While the paper advance is taking place, data is transferred from the storage band indicated by the two most significant digits of m (XX) to the Print Buffer. Registers A and X are used for the transfer and their previous contents are therefore destroyed. Before this instruction is given, the data to be printed must be arranged in the storage band (XX) according to the print interlace pattern. The computer is released for other operations as soon as the buffer band is loaded. To advance the paper 50 through 79, the digits 5, 6, or 7 are coded in *binary* form and placed in the normal n position of the instruction. — 592

ZERO SUPPRESSION

62 — c — This instruction is used to suppress characters to the left of the most significant digit of a field. It operates on 6-bit computer code, with the numeric word in rA and the zone word in rX. Results are placed in rA and rX. Zeros (0000 0000) and commas (0011 0101), preceding the first significant digit, are changed to blanks (0000 0110). — 4

SUMMARY OF INSTRUCTION CODES

				Word Times
ARITHMETIC:				
70	m	c	(m) + (rA) ⟶ rA	5
75	m	c	(rA) - (m) ⟶ rA	5
85	m	c	(rL)x(m) ⟶ rA, rX	5+NDM+SDM
55	m	c	(m) ÷ (rL) ⟶ rA, rX	5 + 2NDQ+SODQ+SNCED
TRANSFER:				
25	m	c	(m) ⟶ rA	4
60	m	c	(rA) ⟶ m	4
05	m	c	(m) ⟶ rX	4
65	m	c	(rX) ⟶ m	4
30	m	c	(m) ⟶ rL	4
50	m	c	(rL) ⟶ m	4
77	----	c	(rA) ⟶ rL	3
26	m	----	0's ⟶ rA	3
31	m	----	0's ⟶ rL	3
06	m	----	0's ⟶ rX	3
23	m	----	(rC) ⟶ rA	3
TRANSLATE:				
12	----	c	80 CC (rA), (rL) & (rX) ⟶ MC, rA, rX	3
17	----	c	MC (rA), (rX) ⟶ CC (rA, rL, & rX)	3
COMPARISON:				
82	=	≠	(rA) : (rL)	3
87	>	≤	(rA) : (rL)	3
EDIT:				
86	m	----	0 ⟶ rA, rX, sign of rL goes to sign of rA, rX	14
20	m	c	Superimpose (m) on (rA) ⟶ rA	4
35	m	c	Erase (m) from (rA) ⟶ rA	4
32	- n --	c	Shift ⟶ n, rA & rX	3+N
37	- n --	c	Shift ⟵ n, rA	3+N
62	----	c	Zero suppress (rA)	4
36	m	----	Erase, 0 ⟶ rA, sign unchanged	3
SKIP, STOP:				
00	m	----	Skip to m	2
67	m	c	Stop	Indefinite
INDEX REGISTER:				
02	m	c	m ⟶ IR	3
07	m	c	m + (IR) ⟶ IR, m of rA 0 ⟶ OC & c of rA	4

				Word Time
READ-PUNCH UNIT (Input-Output):				
81	XX00	c	⊙ Band XX Int ⟶ B, Move Cards, Sense, Punch	103
81	XX01	c	⊙ Band XX Int ⟶ Translate ⟶ CC ⟶ B, Move Cards, Sense, Punch	210
46	XX00	c	(B) ⟶ I Int Band XX	203
46	XX01	c	(B) ⟶ Translate ⟶ MC, I Int Band XX	215
22	Yes	No	Test: Buffer loaded? Yes, (rC) ⟶ rA	4,3
57	-1--	c	Select Stacker #1 (Segregate)	3
HIGH-SPEED READER (Input):				
72	m	c	Feed Card, Sense	3
96	XX00	c	(B) ⟶ J Int Band XX	203
96	XX01	c	(B) ⟶ Translate ⟶ MC, J Int Band XX	215
42	Yes	No	Test: Buffer loaded? Yes, (rC) ⟶ rA	4,3
47	-n--	c	Select Stacker #0, 1 or 2 (Sort)	3
HIGH-SPEED PRINTER (Output):				
11	XXnn	c	Advance nn Lines & Print from Band XX	592
16	--nn	c	Advance nn Lines	4
27	No	Yes	Test: Printer in operation? No, (rC) ⟶ rA	4,3

KEY:

NDM	Number of digits in multiplier
SDM	Sum of digits in multiplier
2NDQ	Two times the number of digits in quotient
SODQ	Sum of odd digit-positions in quotient
SNCEDQ	Sum of the nines complement of even digit-positions of quotient
----	Ignore m or c or portion thereof
CC	80-Column Card Code
MC	Computer (Machine) Code
B	Buffer
Int	Interlace

4. Minimum Latency

GENERAL

In programming the UNIVAC Solid-State 80 Computer the user need not concern himself with tedious timing considerations, since interlock and the various buffer tests prevent timing errors in coordinating operations of the four units. Moreover, the interlace patterns for card images allow him to work on successive fields in a single drum revolution. To take complete advantage of the system's speed, however, the programmer will recognize that certain elementary timing factors should be included in a "tight" program. He will, for example, be aware that he can minimize the time needed to secure data from its storage locations. In computer terminology, he would want to "code in minimum latency."

Two features of the UNIVAC Solid-State 80 Computer enable the programmer to achieve minimum latency. To begin with, the one and one-half address instruction code specifies the address of both the operand and of the next instruction. The programmer can consequently arrange instructions in a pattern which effectively minimizes the time it takes to locate them for execution. Second, the 1,000 words of high-speed access storage reduce the maximum selection time of data stored there to only .85 milliseconds.

MINIMUM LATENCY RULES

Minimum latency, therefore, can be accomplished by means of the following simple rules:

THE FIRST RULE: For instructions having an m address...

The location of an instruction requiring an operand is m-2. The address of the next instruction, c, is determined by adding, to the location of the m portion of the current instruction, only the time required for the third and fourth phases of the instruction cycle. Thus, the c address of an add instruction, for example, would be m+3.

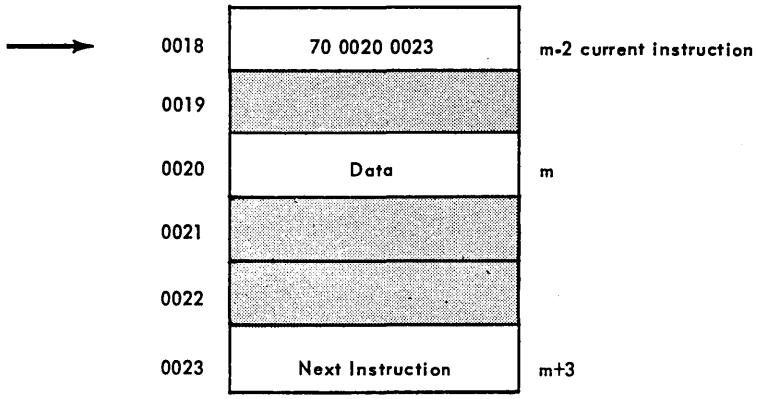
A brief example will illustrate the first rule. There is a data word in storage location 0020.

thus m = 0020
and m-2= 0018

Pursuing the first rule further, assume that the instruction in storage location 0018 is an add (70), in which the third and fourth phases consume 3 word times in minimum latency. According to the first rule, the address of the next instruction is fixed by adding 3 to m of the current instruction.

thus since m = 0020
0020+3 = 0023, the location of the next instruction

Consequently, to take advantage of minimum latency coding an add instruction the m address of which is 0020 must be placed in 0018 while the next instruction is in 0023. A graphic presentation of the first rule appears on the following page.



Normally, the shaded storage locations are also occupied with either data or instructions which are omitted here to emphasize the first rule of latency.

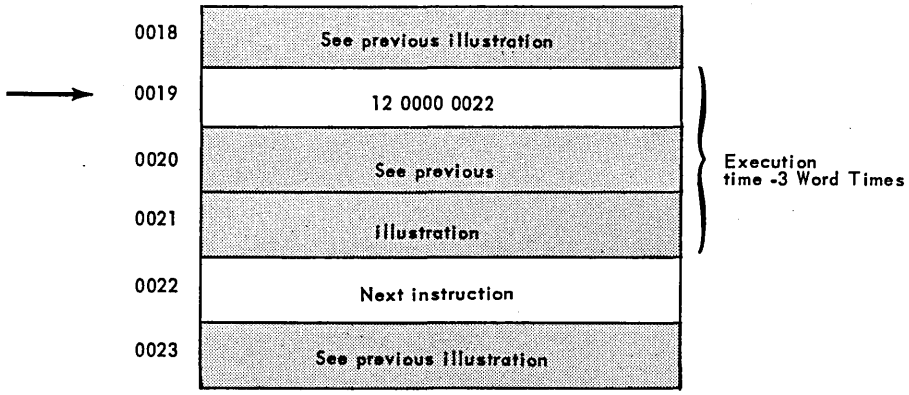
The m and c portions of this instruction may be located in any other band at the same level.

THE SECOND RULE: For instructions which do not have an m address...

The location of an instruction in which the m address is ignored is not computed with reference to the m address. Frequently, the address of such an instruction will have been determined by the fact that it was the c address of a previous instruction. Occasionally the instruction may be interspersed with others, because its location happens to be one of those bypassed by the cycle of a previous instruction, such as those shaded areas in the foregoing illustration (0019, 0021, 0022).

The c address of the instruction in which the m address is ignored, is consequently governed by the location of the instruction itself. By adding to this location the number of word times required for the entire execution of the instruction, the location of the next instruction is determined.

For example, if a translation instruction (12) is in location 0019, its c address should be 0022, because the entire execution takes three word times. Graphically this would appear as:



Certain instructions, such as the 11, 46, 96, and 81 should be placed in specific locations and should contain specific c addresses. These locations can be learned from the Minimum Latency Table, which also aids in using Rules 1 and 2.

GENERAL RULE: Instructions and data whose location in a standard access band would cause more than fifty word times to elapse before access to them is gained should be placed in a high-speed access band. Naturally, any of the addresses computed according to Rules 1 and 2 can be used on the high-speed access, as well as on the standard access bands. In using the high-speed access bands, the computed address can be added to 4000, 4200, 4400, 4600, and 4800. In addition, the address can be added to 4050, 4250,... 4850; 4100, 4300,... 4900; 4150, 4350,... 4950. These additional locations may be used because the high-speed access bands are serviced by four read-write heads which equalize the access time to the above addresses.

5. Index Register Programming

INTRODUCTION

In the UNIVAC Solid-State 80 System, each Index Register has the capacity to hold four digits. Consequently, there is no restriction on the manner in which the operand address can be modified: for every possible modification there is a value that can be stored in an Index Register to effect the modification.

There are instructions to store initial values in Index Registers and instructions to augment the contents of an Index Register by any increment desired. Thus, complete flexibility in the use of these registers is provided. The instruction that increments the contents of an Index Register also delivers the result of the incrementation to register A, where it is readily accessible for testing. Consequently, the Index Registers provide easy control for looping. Moreover, the instruction always remains in its original form in main storage, even though in execution it may be modified by the contents of an Index Register. Therefore, an instruction can be called on again and again without the programmer's having to be concerned with whether or not it has been modified.

APPLICATION OF INDEX REGISTERS

The basic purpose of Index Registers is to make address modification a hardware, rather than a program, function. The result of this feature is, in effect, an expansion of main storage, since address modification, a significant characteristic of most programs, can be achieved with considerably fewer program steps. In addition, the use of Index Registers often produces a significant decrease in program running time.

HIGH-SPEED READ ROUTINE

BUFFER UNLOAD

For example, in the High-Speed Read Routine with four reserve storages when the contents of the Reader Buffer are transferred to the main storage interlace, the routine must do several things. First of all, it must transfer the first read station image from the interlace to one of the four reserve storages. For the UNIVAC Solid-State 80, this image will consist of 24 words, and it will take 48 instructions (twenty-four 25 instructions and twenty-four 60 instructions) to transfer the image from the interlace to the reserve storages. If this operation were programmed in a straight-line fashion, 192 instructions would be required: 48 to transfer the image from the interlace to the reserve storages on band 02; 48 to transfer the image to the reserve storages on band 04; and so on.

Actually, without Index Registers, this operation is programmed once (48 instructions) and another set of approximately 48 instructions are used to modify this base set of instructions to transfer the image to the proper reserve storages. However, with Index Registers, this operation can be programmed once with all the 60 instructions calling on a specific Index Register and also addressing the reserve storage band 02. Then, if the image is to be transferred to the reserve storages on band 02, the Index Register can be set to zero; the set of instructions can be executed, and the transfer will have been effected properly. Similarly, if the transfer is to the reserve storage on band 04, the Index Register can be set to 200. The same instructions can then be executed, and again the transfer will be effected properly. An Index Register setting of 400 delivers the image to the reserve storage on band 06; and a setting of 600, to the reserve storage on band 08. The use of Index Registers in this instance represents approximately a 50 per cent saving in space (only 48 base instructions and 4 instructions to set the Index Register to its proper value are required) and also about 50 per cent saving in time.

COMPARE IMAGES

The next operation which the High-Speed Read Routine must perform is to compare the second read station image in the interlace with an image in one of the four reserve storages. It will take 72 instructions (twenty-four 25's, twenty-four 30's and twenty-four 82's) to effect this comparison. Each of the 30 instructions must have the ability to address any one of the four reserve storages. Again, the use of Index Registers in this area effects an approximate savings of 50 per cent in both space and time.

GET NEXT IMAGE

Finally, when the Read Routine is called upon to produce the next image, the routine must transfer that image from any one of the four reserve storages to working storage. Once more Index Registers allow this transfer to be effected with a saving of approximately 48 words of storage and a 50 per cent saving in time. Thus, throughout the High-Speed Read Routine, which is used over and over again in almost every existing run, Index Registers save approximately 168 words of storage and 50 per cent of the time that would be spent in a routine which does not have access to Index Registers.

EDIT

Another common operation in UNIVAC Solid-State 80 routines is the editing of alphanumeric information. The basic characteristic of this operation is processing the primed word of the information in exactly the same way as the unprimed and duo-primed word. If all instructions in such editing routines that call on the data for operands specify an Index Register, the same instructions which are used to process the unprimed word can also be used to process the primed and duo-primed words with a negligible expenditure of time. This approach represents a 50 per cent saving in space for all such operations in a program.

6. Tables

MINIMUM LATENCY TABLE

Operation Code	I (Location of Instruction Word)	D (Location of Data – Operand)	C (Location of Next Instruction)
05 25 30	m – 2	m	m + 2
50 60 65	m – 2	m	m + 2
20 35	m – 2	m	m + 2
70 75	m – 2	m	m + 3
06 26 31 77	I		I + 3
82 87 02	I		I + 3
12 17	I		I + 3
57 47 72	I		I + 3
16 62 07	I		I + 4
32 37	I		I + 3 + n
42 22 27	I		I + 3 (c) ; I + 4 (m)
86	I	I + 14 = (NI)	
23	I	I + 3 = (NI)	
85	m – 2	m	m + (3 + NDM + SDM)*
55	m – 2	m	m+(3+2NDQ+SODQ+SNCEDQ)*
11	Level 197		Level 189
81 (Untranslated)	Level 098		Level 001
96 (Untranslated)	Level 198		Level 001
46 (Untranslated)	Level 098		Level 101
96 (Translated)	Level 198		Level 013
81 (Translated)	Level 098		Level 108
46 (Translated)	Level 098		Level 113

KEY*

NDM = Number of digits in multiplier	**SDM** = Sum of digits in multiplier	**2NDQ** = Two times the number of digits in quotient	**SODQ** = Sum of odd digit-positions in quotient	**SNCEDQ** = Sum of nines complement of even digit-positions in quotient

KEY TO CARD INTERLACE TABLE

The 80-column punched card is represented in the computer as 24 words. Each group of ten columns forms a data word of three images called the unprimed, primed, and duo-primed images. Each image is a computer word and is an exact representation of the holes appearing on a particular section of the card: a punch equals a "1" bit.

Rows YX	0	1	2	3	4	5	6	7
0 1 2 3 4 5	0'	1'	2'	3'	4'	5'	6'	7'
6 7 8 9	0''	1''	2''	3''	4''	5''	6''	7''
Columns	1-10	11-20	21-30	31-40	41-50	51-60	61-70	71-80

The signs of all images are positive.

Any combination of punches may be represented within the computer; however, translation instructions are provided only for the standard Hollerith code.

To locate the interlace factor, the base of the symbol is:

I = input on Read-Punch Unit
J = input on High-Speed Reader
O = output on Read-Punch Unit

Added to this base is the first subscript to indicate the station; e.g. J_2 refers to input of the second read station of the High-Speed Reader. The second subscript refers to the column group on the card (see above).

The absence of an apostrophe refers to the unprimed position. An apostrophe indicates the primed portion and a double apostrophe, the duo-primed portion; e.g. O''_{12} refers to the duo-primed portion of card-column group #2 which is output on the first and only punch station of the Read-Punch Unit. With a symbol such as this, the corresponding interlace factor can be added to the m band of the input or output instruction to determine the exact location of the card group.

To locate data which is automatically translated as it leaves the buffer, a similar procedure is followed. A symbol with a base and subscripts, as indicated above, is constructed. The absence of an apostrophe indicates the numeric word, and the apostrophe indicates the zone word. The double apostrophe is not used. To locate the corresponding factor, however, those columns of the interlace table marked "translated" are used.

INTERLACE TABLE
High-Speed Reader Interlace

UNTRANSLATED				TRANSLATED			
Storage Location 00XX		Storage Location 01XX		Storage Location 00XX		Storage Location 01XX	
00	50	00	50	00	50	00	50
01	51 J14	01	51	01	51	01	51
02	52	02	52 J"23	02	52	02	52
03 J10	53	03	53	03	53	03	53
04	54	04 J"27	54	04	54 J'13	04	54 J23
05	55 J'14	05	55	05	55	05	55
06	56	06	56 J24	06 J27	56	06	56
07 J'10	57	07	57	07	57	07	57
08	58	08 J20	58	08	58	08	58
09	59 J"14	09	59	09	59	09	59 J'23
10	60	10	60 J'24	10	60	10	60
11 J"10	61	11	61	11 J'27	61 J14	11 J'17	61
12	62	12 J'20	62	12	62	12	62
13	63 J15	13	63	13 J10	63	13	63
14	64	14	64 J"24	14	64	14	64
15 J11	65	15	65	15	65	15	65
16	66	16 J"20	66	16	66 J'14	16	66 J24
17	67 J'15	17	67	17	67	17	67
18	68	18	68 J25	18 J'10	68	18 J20	68
19 J'11	69	19	69	19	69	19	69
20	70	20 J21	70	20	70	20	70
21	71 J"15	21	71	21	71	21	71 J'24
22	72	22	72 J'25	22	72	22	72
23 J"11	73	23	73	23	73 J15	23 J'20	73
24	74	24 J'21	74	24	74	24	74
25	75 J16	25	75	25 J11	75	25	75
26	76	26	76 J"25	26	76	26	76
27 J12	77	27	77	27	77	27	77
28	78	28 J"21	78	28	78 J'15	28	78 J25
29	79 J'16	29	79	29	79	29	79
30	80	30	80 J26	30 J'11	80	30 J21	80
31 J'12	81	31	81	31	81	31	81
32	82	32 J22	82	32	82	32	82
33	83 J"16	33	83	33	83	33	83 J'25
34	84	34	84 J'26	34	84	34	84
35 J"12	85	35	85	35	85 J16	35 J'21	85
36	86	36 J'22	86	36	86	36	86
37	87 J17	37	87	37 J12	87	37	87
38	88	38	88 J"26	38	88	38	88
39 J13	89	39	89	39	89	39	89
40	90	40 J"22	90	40	90 J'16	40	90 J26
41	91 J'17	41	91	41	91	41	91
42	92	42	92 J27	42 J'12	92	42 J22	92
43 J'13	93	43	93	43	93	43	93
44	94	44 J23	94	44	94	44	94
45	95 J"17	45	95	45	95	45	95 J'26
46	96	46	96 J'27	46	96	46	96
47 J"13	97	47	97	47	97 J17	47 J'22	97
48	98	48 J'23	98	48	98	48	98
49	99	49	99	49 J13	99	49	99

INTERLACE TABLE
Read-Punch Unit Interlace

UNTRANSLATED				TRANSLATED			
Storage Location 00XX		Storage Location 01XX		Storage Location 00XX		Storage Location 01XX	
00	50	00	50	00	50	00	50 O14
01	51	01	51 I14	01	51	01	51
02	52 I"23	02	52	02	52	02 O10	52
03	53	03 I10	53 O"13	03	53	03	53
04 I"27	54	04	54	04	54 I23	04	54 I'13
05	55	05 O"17	55 I'14	05	55	05	55 O'14
06	56 I24	06	56	06	56	06 I27	56
07	57	07 I'10	57 O14	07	57	07 O'10	57
08 I20	58	08	58	08	58	08	58
09	59	09 O10	59 I"14	09	59 I'23	09	59
10	60 I'24	10	60	10	60	10	60
11	61	11 I"10	61 O'14	11 I'17	61	11 I'27	61 I14
12 I'20	62	12	62	12	62	12	62 O15
13	63	13 O'10	63 I15	13	63	13 I10	63
14	64 I"24	14	64	14	64	14 O11	64
15	65	15 I11	65 O"14	15	65	15	65
16 I"20	66	16	66	16	66 I24	16	66 I'14
17	67	17 O"10	67 I'15	17	67	17	67 O'15
18	68 I25	18	68	18 I20	68	18 I'10	68
19	69	19 I'11	69 O15	19	69	19 O'11	69
20 I21	70	20	70	20	70	20	70
21	71	21 O11	71 I"15	21	71 I'24	21	71
22	72 I'25	22	72	22	72	22	72
23	73	23 I"11	73 O'15	23 I'20	73	23	73 I15
24 I'21	74	24	74	24	74	24	74 O16
25	75	25 O'11	75 I16	25	75	25 I11	75
26	76 I"25	26	76	26	76	26 O12	76
27	77	27 I12	77 O"15	27	77	27	77
28 I"21	78	28	78	28	78 I25	28	78 I'15
29	79	29 O"11	79 I'16	29	79	29	79 O'16
30	80 I26	30	80	30 I21	80	30 I'11	80
31	81	31 I'12	81 O16	31	81	31 O'12	81
32 I22	82	32	82	32	82	32	82
33	83	33 O12	83 I"16	33	83 I'25	33	83
34	84 I'26	34	84	34	84	34	84
35	85	35 I"12	85 O'16	35 I'21	85	35	85 I16
36 I'22	86	36	86	36	86	36	86 O17
37	87	37 O'12	87 I17	37	87	37 I12	87
38	88 I"26	38	88	38	88	38 O13	88
39	89	39 I13	89 O"16	39	89	39	89
40 I"22	90	40	90	40	90 I26	40	90 I'16
41	91	41 O"12	91 I'17	41	91	41	91 O'17
42	92 I27	42	92	42 I22	92	42 I'12	92
43	93	43 I'13	93 O17	43	93	43 O'13	93
44 I23	94	44	94	44	94	44	94
45	95	45 O13	95 I"17	45	95 I'26	45	95
46	96 I'27	46	96	46	96	46 I22	96
47	97	47 I"13	97 O'17	47 I'22	97	47	97 I17
48 I'23	98	48	98	48	98	48	98
49	99	49 O'13	99	49	99	49 I13	99

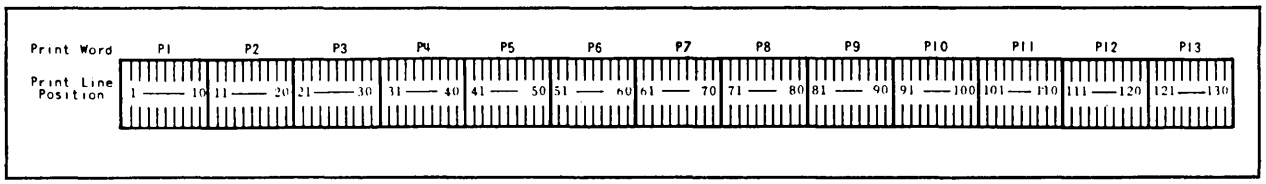
TRANSLATION TABLE

80-Column Code ⟷ Computer Code						
407 Punching Character	rA (Y, X, 0, 1)	rL (2, 3, 4, 5)	rX (6, 7, 8, 9)	rX (Zone)	rA (Numeric)	Printing Character
0	0010	0000	0000	0000	0000	0
1	0001	0000	0000	0000	0001	1
2	0000	1000	0000	0000	0010	2
3	0000	0100	0000	0000	0011	3
4	0000	0010	0000	0000	0100	4
5	0000	0001	0000	0000	1000	5
6	0000	0000	1000	0000	1001	6
7	0000	0000	0100	0000	1010	7
8	0000	0000	0010	0000	1011	8
9	0000	0000	0001	0000	1100	9
A	1001	0000	0000	0001	0001	A
B	1000	1000	0000	0001	0010	B
C	1000	0100	0000	0001	0011	C
D	1000	0010	0000	0001	0100	D
E	1000	0001	0000	0001	1000	E
F	1000	0000	1000	0001	1001	F
G	1000	0000	0100	0001	1010	G
H	1000	0000	0010	0001	1011	H
I	1000	0000	0001	0001	1100	I
J	0101	0000	0000	0010	0001	J
K	0100	1000	0000	0010	0010	K
L	0100	0100	0000	0010	0011	L
M	0100	0010	0000	0010	0100	M
N	0100	0001	0000	0010	1000	N
O	0100	0000	1000	0010	1001	O
P	0100	0000	0100	0010	1010	P
Q	0100	0000	0010	0010	1011	Q
R	0100	0000	0001	0010	1100	R
S	0010	1000	0000	0011	0010	S
T	0010	0100	0000	0011	0011	T
U	0010	0010	0000	0011	0100	U
V	0010	0001	0000	0011	1000	V
W	0010	0000	1000	0011	1001	W
X	0010	0000	0100	0011	1010	X
Y	0010	0000	0010	0011	1011	Y
Z	0010	0000	0001	0011	1100	Z
8,3 #	0000	0100	0010	0001	1111	#
Y,8,3 .	1000	0100	0010	0001	0101	.
X,8,3 $	0100	0100	0010	0010	0101	$
0,8,3 ,	0010	0100	0010	0011	0101	, Comma
Blank	0000	0000	0000	0000	0110	Space (N.P.)
8,4 @	0000	0010	0010	0011	0000	+
Y,8,4 □	1000	0010	0010	0001	0110	:
X,8,4 *	0100	0010	0010	0010	0110	*
0,8,4 %	0010	0010	0010	0011	0110	%
Y &	1000	0000	0000	0001	0111	&
X—	0100	0000	0000	0000	0101	—
0,1 /	0011	0000	0000	0011	0001	/
1,8,4	0001	0010	0010	0000	0111	)
6,8,4	0000	0010	1010	0000	1111	' Apostrophe
5,8,4	0000	0011	0010	0000	1110	;
5,8,3	0000	0101	0010	0000	1101	(

Note: N.P. refers to non-printing characters.

KEY TO PRINT INTERLACE TABLE

The 130 characters that can be printed on one line are divided into 13 10-digit print words. These words, and their corresponding print positions, are shown in the diagram. Before a print instruction is executed, the 13 print words are accumulated by program instructions on a specific main storage band in fixed word locations of the print interlace pattern.



To locate the interlace factor, the base of the symbol is P. The first subscript indicates the print line position, 1 through 13 (see diagram). The absence of an apostrophe refers to the unprimed or numeric word of a word pair in computer code. An apostrophe denotes the primed or zone word.

With this symbol, the corresponding interlace factor can be added to the m band of the print instruction to determine the placement of data to be printed.

INTERLACE TABLE
High-Speed Printer Interlace

LOCATION 00XX		LOCATION 01XX	
00 P1	50 P4	00	50
01	51	01	51
02	52	02	52
03	53	03 P7	53
04	54	04	54
05 P'1	55 P'4	05	55
06	56	06	56
07	57	07	57
08	58	08 P'7	58
09 P10	59	09	59
10	60	10	60
11	61	11	61
12	62 P13	12	62
13	63	13	63
14 P'10	64	14	64
15	65	15	65 P3
16	66	16	66
17	67 P'13	17	67
18 P6	68	18	68
19	69	19	69
20	70	20	70 P'3
21	71	21	71
22	72	22	72
23 P'6	73	23	73
24	74	24	74
25	75	25 P9	75
26	76	26	76
27	77	27	77
28	78	28	78 P12
29	79	29	79
30	80	30 P'9	80
31	81 P2	31	81
32	82	32	82
33	83	33	83 P'12
34	84	34 P5	84
35	85	35	85
36	86 P'2	36	86
37	87	37	87
38	88	38	88
39	89	39 P'5	89
40	90	40	90
41 P8	91	41	91
42	92	42	92
43	93	43	93
44	94 P11	44	94
45	95	45	95
46 P'8	96	46	96
47	97	47	97
48	98	48	98
49	99 P'11	49	99

PRINT CHARACTER TABLE
80-Column Print Code

NUMERIC	ZONE 00	ZONE 01	ZONE 10	ZONE 11
0111	)	&		
0110	Space	:	*	%
0101	— (Minus)	.	$	' (Comma)
0000	0			+
0001	1	A	J	/
0010	2	B	K	S
0011	3	C	L	T
0100	4	D	M	U
1000	5	E	N	V
1001	6	F	O	W
1010	7	G	P	X
1011	8	H	Q	Y
1100	9	I	R	Z
1111	' (Apostrophe)	#		
1110	;			
1101	(			

SPECIFICATIONS

Physical Dimensions

Dimensions (doors closed)

UNIT	LENGTH	WIDTH	HEIGHT	WEIGHT	
				lb	lb /ft^2
Central Processor	108"	32"	69"	3532	146.8
High-Speed Reader	51"	27"	49"	758	91.8
Read-Punch Unit	49"	27"	54"	1334	134.0
High-Speed Printer	72"	32"	51"	1613	101.4

Space Requirements (doors open)

Central Processor	136"	118"	69"
High-Speed Reader	51"	72"	49"
Read-Punch Unit	75"	74"	79"
High-Speed Printer	128"	68"	51"

Heat Dissipation

		AIR TEMPERATURE AT INTAKES	
	Btu min	MAX.	MIN.
Central Processor	284.0	100F	60F
High-Speed Reader	56.6	100F	60F
Read-Punch Unit	56.6	100F	60F
High-Speed Printer	198.5	100F	60F

Power Requirements

Frequency: 60 cycle ± 0.5% maximum deviation.
Voltage: 230-volt, single phase, 3-wire with grounded neutral (115 v each line to ground).

	RATING	SERVICE LINE FUSE SIZE
Central Processor, Read-Punch Unit, High-Speed Reader	9.8 kva	70 amp
High-Speed Printer	4.7 kva	70 amp

Remington Rand Univac
DIVISION OF SPERRY RAND CORPORATION

U1742.2 REV. 2 PRINTED IN U.S.A.