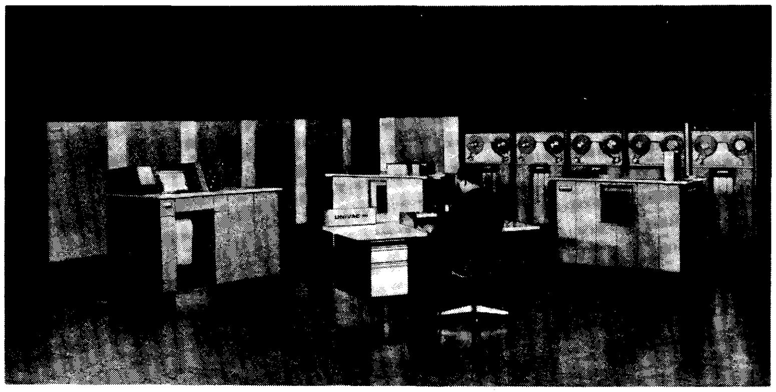


REFERENCE MANUAL

UNIVAC® III DATA PROCESSING SYSTEM

REFERENCE MANUAL

UNIVAC® III DATA PROCESSING SYSTEM

PREFACE

The purpose of this manual is to provide a comprehensive source of information concerning the UNIVAC III System: its basic units, their performance, the performance of the system as a whole, and the instruction list. The manual is addressed to those who have a good general knowledge of electronic computers and therefore can appreciate the significance of the uniquely powerful features of the system without detailed explanations or examples of the use of instructions or programming aids. Descriptions of software packages are available in separate publications.

This manual consists of seven principal sections: a general description, five sections which describe the nature and use of each of the basic units, and appendices containing information on such subjects as modulo-3 checking and timing of multiplication and division. Each section is designed to convey essential information to the informed reader in complete but concise form.

UNIVAC III DATA PROCESSING SYSTEM

For years the goal of designers of electronic data-processing equipment has been to achieve a system the units of which could operate not only simultaneously but also at their full rated speeds.

Now, with the UNIVAC III System, this goal has been attained. This system is capable of performing calculations and data manipulations and operating its peripherals at full speed, under the control of several concurrently running programs.

Briefly summarized here are some of the principal characteristics of the system which account for its unusual merit in general data-processing applications.

HIGH-SPEED MAGNETIC TAPES—Numeric read-write speed, 200,000 digits per second—alphanumeric read-write speed, 133,000 characters per second. A 3600-foot reel of tape can be rewound in 125 seconds. Scatter-read and gather-write operations permit direct transfer of magnetic-tape data to and from nonadjacent areas of magnetic-core storage. This obviates the need for many data-transfers and thus saves large amounts of time and storage space.

HIGH-SPEED PROCESSING—4-microsecond memory cycle. The system executes most instructions in only 8 microseconds. In 12 milliseconds the UNIVAC III system can add 1000 12-digit numbers. Multiplication and division are performed at correspondingly high speeds.

VERSATILE COMMAND STRUCTURE—The instructions of the UNIVAC III System can address data directly or can indirectly address operands of up to four 6-digit words, thus saving many programming steps and much time on operations such as sorting and table referencing.

Using multiword operands and field selection, instructions operate directly upon fields which may range in size from a single bit, digit, or character to 96 bits, 24 digits, or 16 characters, eliminating the need for isolation of packed fields or separately processing each word of multiword fields. The effect of this is a large economy in programming and operating time otherwise required for shifting, extracting, and data transfers.

The system may be provided with either 9 or 15 index registers which automatically modify the operand address as part of the basic instruction execution cycle—no additional time need be allocated to operate the registers.

SYSTEMS MODULARITY—The UNIVAC III System can be smoothly and efficiently expanded by adding up to four 8192-word core-storage modules to provide storage for up to 32,768 words; up to 32 UNISERVO IIIA magnetic-tape units; and an extensive and versatile array of peripherals.

SIMULTANEOUS OPERATIONS—As many as 13 on-line input-output operations can proceed in parallel with each other and with operations of the Central Processor under the control of several concurrently running programs.

HIGH-SPEED PERIPHERALS—High-Speed Reader—700 cards a minute; Card-Punch Unit—300 cards a minute; High-Speed Printer—922 128-character lines of numeric data per minute or 700 128-character lines of alphanumeric data per minute.

DEPENDABILITY—The circuitry of the UNIVAC III System is based on that proved in use in the famous UNIVAC LARC, one of the fastest, most reliable computers ever built. The UNIVAC III System also contains many automatic self-checking features which ensure dependable operation.

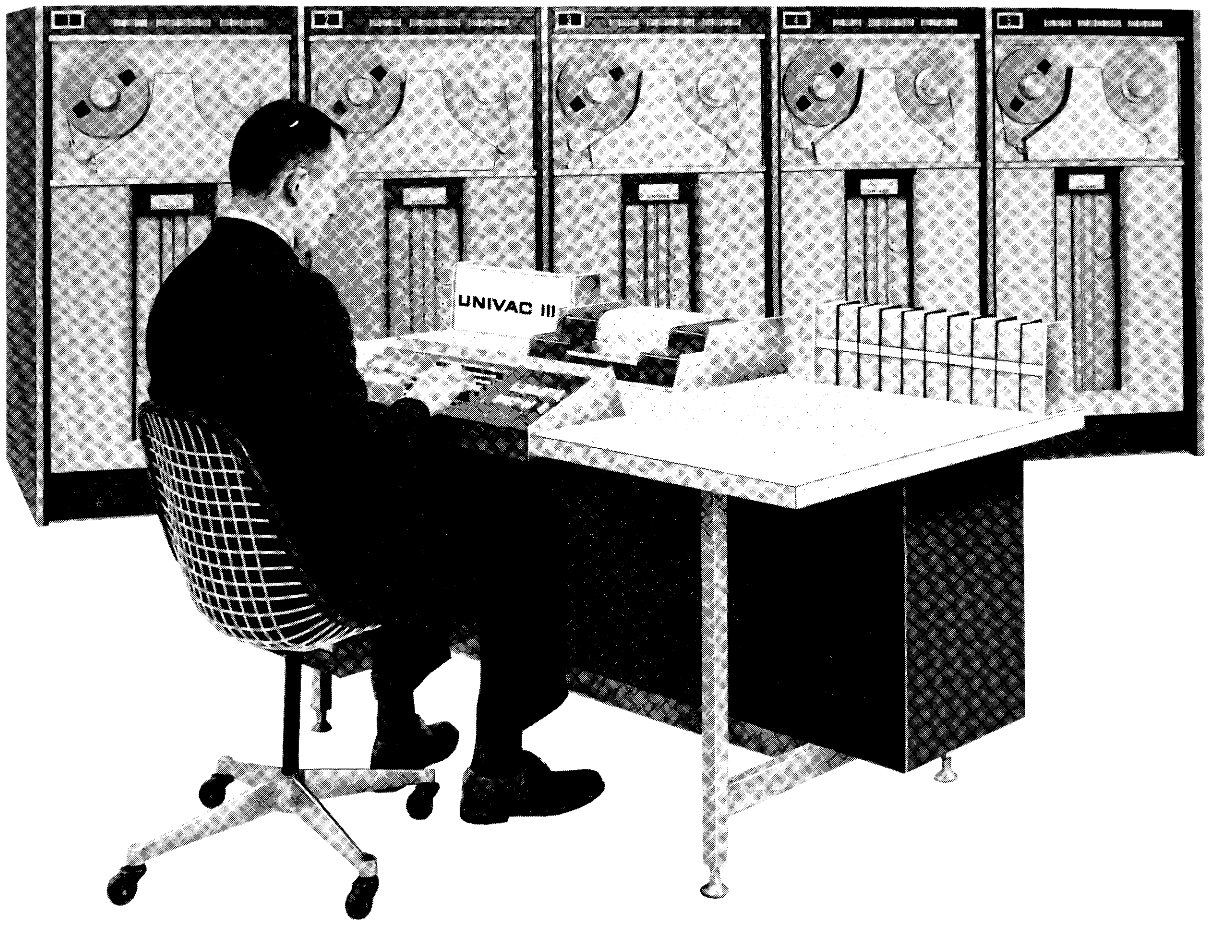
A COMPREHENSIVE AND EFFICIENT PROGRAMMING SYSTEM—Every UNIVAC III System is provided with an assembly system with macroinstruction facilities, routine generators, and the ability to utilize program libraries—an executive monitoring system which controls simultaneous full-speed operation of peripherals and concurrently running programs—a powerful sort/merge generator—utility, service, and program-testing routines—an extensive COBOL compiler; and a FORTRAN compiler which has great speed and is compatible with most other FORTRAN systems.

CONTENTS

APPENDICES

SECTION 1

GENERAL DESCRIPTION



A typical basic UNIVAC III System comprises a Central Processor with magnetic-core storage, a UNISERVO IIIA tape system, a High-Speed Reader, a Card-Punch Unit, and a High-Speed Printer. The basic system may be expanded to include up to 32 UNISERVO IIIA tape units, a UNISERVO IIA tape system, and a variety of peripheral units, as shown in figure 1-1.

■ CENTRAL PROCESSOR

The UNIVAC III Central Processor consists of an expandable magnetic-core storage, arithmetic and control units, and an operator's console and typewriter.

At the user's option, the processor may have either 9 or 15 index registers. It also may have an addressable clock.

MAGNETIC-CORE STORAGE

The UNIVAC III magnetic-core storage consists of ferrite cores arranged in horizontal planes 64 cores wide by 64 long. Twenty-seven of these planes form a stack with a storage capacity of 4096 words; and two stacks form a memory module, which is the primary storage unit of the system. Each memory module has a storage capacity of 8192 words. One, two, or three modules may be added, to increase the core-storage capacity of the system to 16,384, 24,576, or 32,768 words.

The primary unit of information in the UNIVAC III System is a fixed-length word consisting of 27 bits. Twenty-four of these bits represent an instruction, a control word, or data in alphanumeric, decimal, or binary format. The twenty-fifth bit represents the sign in a data word, and control information in an instruction or control word. The remaining two bits are used to check the accuracy of information transfers and arithmetic operations. The memory cycle—selecting, reading, and regenerating the 27 bits of a word in core storage—is four microseconds.

ARITHMETIC UNIT

The arithmetic unit of the Central Processor performs the calculations and data manipulation called for by the instructions. It contains an adder for decimal and binary arithmetic operations, four one-word arithmetic registers, and additional circuitry which provides a wide range of data-handling abilities.

Data is processed in the arithmetic unit bit-parallel, digit-serial. Because the digit rate through the arithmetic unit is less than 0.5 microsecond per digit, the execution time for most instructions is only 8 microseconds: 4 microseconds to access the instruction, and 4 to access and process the operand.

The four arithmetic registers may be used in nearly all data-manipulation instructions to process operands of from one to four words in length. When a field is split across computer words or is packed in a word with other fields, field selection may be used to designate only those bits, digits, or characters to be manipulated. Using these features, operands ranging in length from one data unit (bit, digit, or character) to four words (96 bits, 24 digits, 16 characters) may be operated upon directly, eliminating the need for extra program steps to isolate packed fields, to align fields with computer words, or to process separately the individual words of multiword fields.

Operations of the arithmetic unit are automatically checked by modulo-3 congruence arithmetic.

CONTROL UNIT

The control unit of the Central Processor selects, interprets, and initiates the execution of instructions in the stored programs which govern the operation of the system. Instructions, which are single-address, are executed sequentially.

Address Modification

Operand addresses are indexed automatically as part of the UNIVAC III instruction cycle. As an instruction is being set up for execution, the contents of one of the index registers are used to modify the effective operand address by means of an adder separate from that of the arithmetic unit. By overlapping the indexing step with other phases of the instruction cycle, and by using the special adder, address modification is made a part of the basic instruction cycle and requires no additional time.

Addresses also may be modified in the UNIVAC III System by means of indirect addressing. With indirect addressing, an instruction can specify the location in which the address of the operand is stored, instead of specifying the operand address directly. This capability is valuable in sort-merge procedures, the manipulation of variable lines of coding, table-handling routines, and other program functions that require flexible and efficient techniques of variable addressing.

Automatic Program Interrupt

An important function of the control unit is to detect special conditions in the system which require programmed or operator action outside of the program in progress. When any of these conditions is detected, the program in progress is interrupted automatically. To provide for subsequent re-entry into the interrupted program, the point at which interrupt occurs is recorded in a fixed memory location. Program control is then transferred to another fixed location, where a routine to process the condition causing the interrupt is initiated. There are six fixed memory locations; the two used when an interrupt occurs are selected by the control unit according to the class of special condition detected: input-output, contingency, or processor error. A set of program-testable indicators is also associated with each class of interrupt; these indicators specify the condition or conditions which caused the interrupt.

Input-Output Interrupt. Through the use of input-output interrupt, the input-output equipment can be kept running at full capacity, with relatively small amounts of central-processor time required for input-output control. The operation of each input-output unit is under the direct control of an associated synchronizer. The performance of a particular input-output operation is initiated by a central-processor instruction which transfers a control word (function specification) to a fixed memory location associated

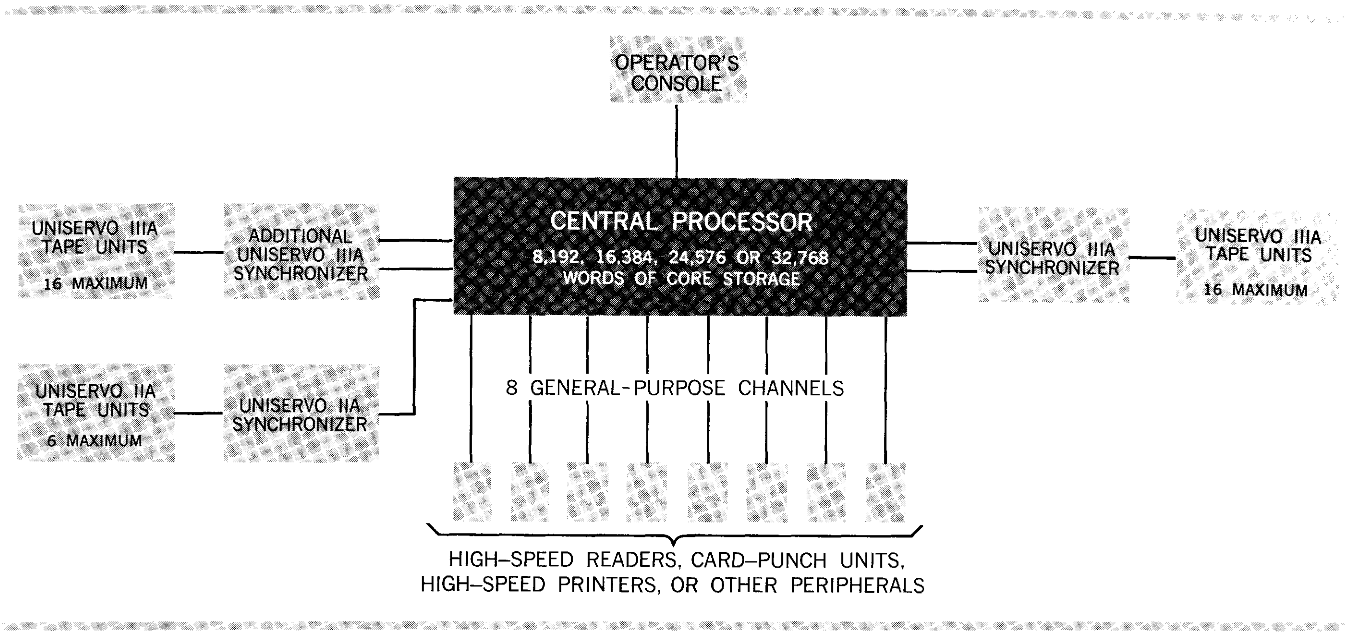

OPERATOR'S CONSOLE
CENTRAL PROCESSOR
8,192, 16,384, 24,576 OR 32,768 WORDS OF CORE STORAGE
UNISERVO IIIA TAPE UNITS 16 MAXIMUM
ADDITIONAL UNISERVO IIIA SYNCHRONIZER
UNISERVO IIA TAPE UNITS 6 MAXIMUM
UNISERVO IIA SYNCHRONIZER
UNISERVO IIIA SYNCHRONIZER
UNISERVO IIIA TAPE UNITS 16 MAXIMUM
8 GENERAL-PURPOSE CHANNELS
HIGH-SPEED READERS, CARD-PUNCH UNITS, HIGH-SPEED PRINTERS, OR OTHER PERIPHERALS

Figure 1-1. UNIVAC III System, Block Diagram

with the input-output synchronizer concerned. After the 12 microseconds required to execute the initiate instruction, the Central Processor is free to initiate the operation of other input-output units or to execute other instructions.

When the input-output unit is ready to perform the specified operation, its synchronizer accesses and decodes the function specification, which causes the unit to begin executing the specified operation. The synchronizer performs all data transfers, data and equipment checking, and other required control functions.

When another operation can be initiated for the particular unit, or if the successful performance of the current operation is blocked by an abnormal condition, program-testable indicators are automatically set. The program in progress is then interrupted and program control is transferred to an input-output interrupt routine. By testing the indicators, the interrupt routine determines the exact cause of the interrupt and either initiates another operation for the unit concerned or, in the case of an abnormal condition, takes the required corrective action. Thus, input-output equipment operates at full capacity, and relatively small amounts of central-processor time are expended for input-output control. In addition, because input-output operations controlled by different synchronizers proceed independently of one another, error-recovery procedures are performed without disrupting the overall operation of the system.

When another input-output operation has been initiated, or when the required corrective action has been taken, program control may be returned to the program that was interrupted, transferred to the program for which the input-output operation was performed, or transferred to another program. Thus, the executive monitoring routines control the use of central-processor time when several programs are running concurrently.

Contingency Interrupt. The program is interrupted if any of the following events occur at the Operator's Console:

The operator requests use of the keyboard to type in information;

A character has been typed in or out;

A type-in has been completed; or

The operator requests a program stop.

Contingency interrupt also occurs if an arithmetic operation resulted in an overflow, an invalid operation code was specified, or the clock was addressed after clock power was removed.

Processor-Error Interrupt. If a word accessed in any part of the system is found to contain incorrect check bits, or if an error occurs in the addressing of a memory location, processor-error interrupt occurs.

An error in the operation of the arithmetic unit also will cause a processor-error interrupt.

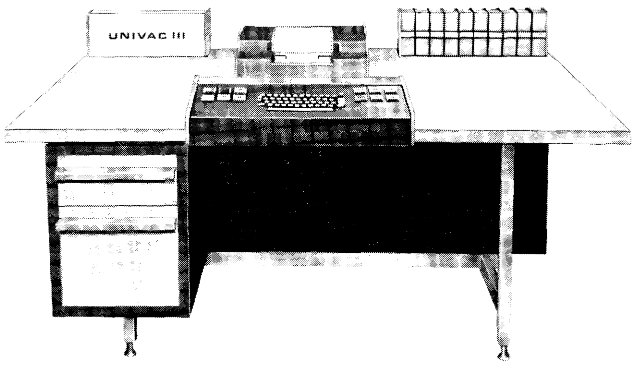
Priority Circuits

The priority circuits of the control unit grant memory access to the various units of the system in a sequence which enables each input-output device and system running time to be used most efficiently. The priority circuits receive, store, and grant requests for access to memory made by the input-output synchronizers and the Central Processor. When simultaneous requests are made, the priority circuits select the synchronizer which is to be granted memory access according to the relative data-transfer rate of the input-output device controlled by each synchronizer. A synchronizer that controls a unit with a relatively slow transfer rate, such as the Card-Punch Unit, requires access to memory less often than a synchronizer which controls a unit with a relatively fast transfer rate, such as a UNISERVO IIIA tape unit; thus the Card-Punch Unit synchronizer has a lower priority than the tape-unit synchronizer.

The Central Processor has the lowest priority, since delaying a central-processor request for memory access will not disrupt the execution cycle or cause loss of information.

The general order of priority is as follows:

1. UNISERVO IIIA synchronizers
2. UNISERVO IIA synchronizer
3. General-purpose input-output channels
4. Central Processor operand access
5. Central Processor instruction access



OPERATOR'S CONSOLE

The operator's console and the console typewriter are the primary media of communication between the operator and the system. The console contains buttons and lights which the operator uses to control and monitor the operation of the Central Processor and the input-output units. The typewriter is used to transmit and record operator-program communications, which include program requests for operator intervention, the detection of machine malfunctions, program running time, and other logging information.

INPUT-OUTPUT EQUIPMENT

The UNIVAC III System has 13 input-output channels, through which data and control information are transferred between the input-output units and the Central Processor. Four channels—two read and two write—are allocated to the UNISERVO IIIA tape system; eight channels are general-purpose input-output channels, allocated to card-punch units, printers, and other peripheral devices; and one channel is the read-write channel for a UNISERVO IIA tape system. As many of the 13 channels as are desired may be used initially; the remaining channels then are available to meet increasing data-processing requirements.

The UNIVAC III System operates outstandingly as a high-speed tape-to-tape data-processing system. In addition, four features make possible economical on-line operation of peripheral units: solid-state components; synchronizer control of all peripheral units; fast processor operating speed compared with peripheral-unit operating speeds; and flexibility of control through the use of executive monitoring routines.

The UNIVAC III System contains solid-state components, which have been successfully tested in hundreds of thousands of hours of use in the UNIVAC LARC and UNIVAC Solid-State Systems. These thoroughly reliable components and the advanced electromechanical design of UNIVAC III peripheral equipment greatly reduce the possibility of system failure due to the malfunction of a peripheral unit.

Because control functions are relegated to individual input-output synchronizers, input-output operations can proceed in parallel with one another and with operations of the Central Processor. As described under the heading *Input-Output Interrupt,* the appropriate input-output routine in the Central Processor is alerted by an automatic interrupt whenever another input-output operation can be initiated.

Because the processor operates much faster than the peripheral devices, there is ample time to process data and to control the simultaneous operation of the peripheral devices.

Since the configuration of peripheral equipment will vary from one installation to the next, the requirements for controlling peripheral operations also will vary. Executive monitoring systems and the input-output subroutines of the assembly systems provide the necessary flexibility to control peripherals; these software systems may be modified readily to meet the requirements of any installation.

To illustrate the efficiency of concurrent on-line peripheral operations, a data-processing example is described below and illustrated in figure 1-2.

> A tape-to-tape file-maintenance run operates concurrently with a tape-to-printer run and a card-to-tape conversion. The primary input to the file-maintenance run is a master file (composed of 250-word records), which is processed against a transaction file. The master file contains 19,800 records, and the transaction file contains 6600 records. The entries in both files have been preordered in ascending sequence and are recorded on tape in 500-word blocks. The output of the file-maintenance run is an updated master file, which is approximately the same size as the master file, and an error file, which lists erroneous transaction records for subsequent printing.
>
> While the file-maintenance run is proceeding, 1000 paychecks and withholding statements are printed from a pre-edited tape file. At the same time, the information from 3500 80-column cards is checked and converted to tape for processing in a subsequent run.

These three runs are completed in approximately five minutes. Of this, 34 seconds of central-processor time is used for the tape-to-printer and card-to-tape runs. The remaining time is available for the file-maintenance run. Thus, the Central Processor controls the concurrent operation of the peripheral devices at the same time that it is processing tape data.

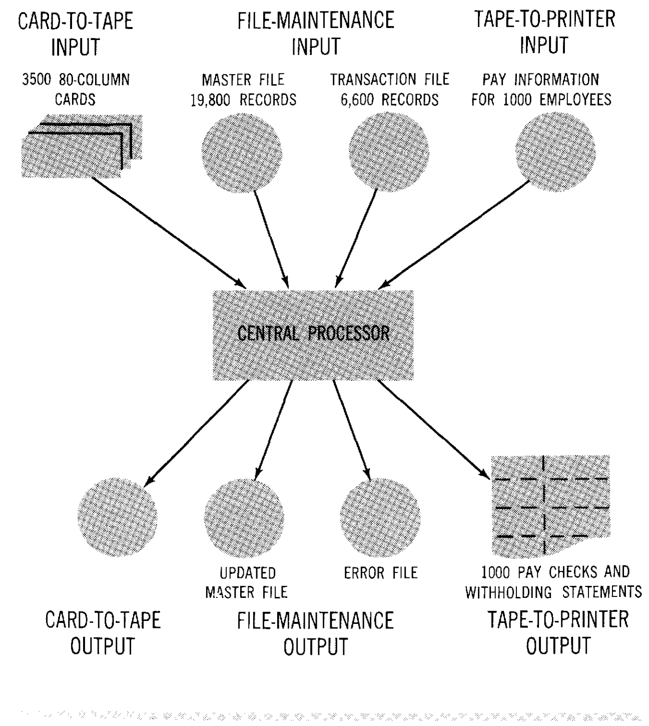
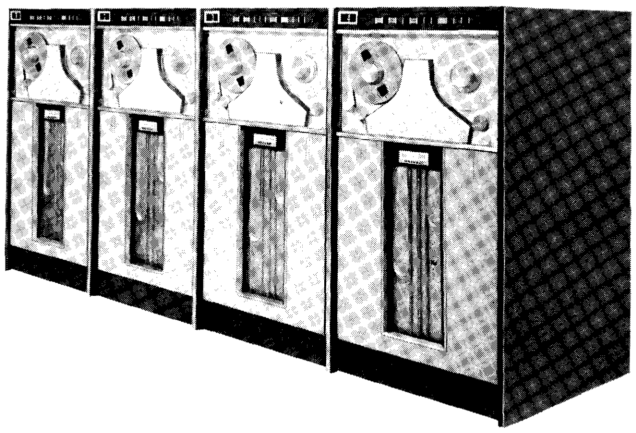


Figure 1-2. Data-Processing Example



UNISERVO IIIA TAPE SYSTEM

A basic UNISERVO IIIA tape system consists of a UNISERVO IIIA synchronizer and up to 16 UNISERVO IIIA tape-handling units. The tape-handling units are controlled by the synchronizer under the direction of function specifications and control words supplied by the program. Data and control information are transferred between the synchronizer and the Central Processor through a write channel and a read channel. The write channel is used for writing tape forward and for reading tape forward and backward; the read channel is used for reading tape forward and backward. Since the two channels transfer data independently of one another, two read operations, or a read operation and a write operation, can be performed simultaneously.

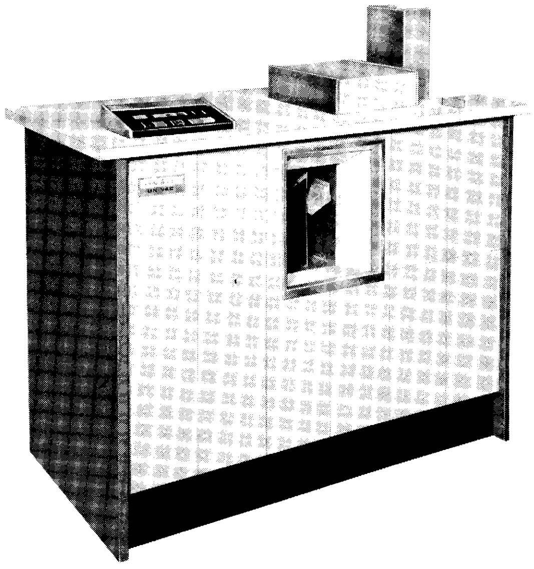
Information is recorded on tape in variable-length blocks at a density of 1333 characters (2000 digits) per inch with an interblock gap of approximately 0.6 inch; tape moves past the read-write heads at a rate of 100 inches per second. Thus, each tape-handling unit transfers 133,300 characters (200,000 digits) per second. Rewind time is 125 seconds for a 3600-foot reel of tape.

Scatter-read and gather-write functions permit tape data to be transferred to and from nonadjacent areas of memory under the direction of control words supplied by the program. This means that data being read from tape can be dispersed automatically to the areas of memory in which it will be processed by the program. During write operations, data can be transferred directly to tape from the areas of memory in which it is computed, and automatically assembled into blocks by the synchronizer.

All data written onto tape is automatically check-read. If an error occurs, the synchronizer marks the incorrectly written block with a bad-spot pattern and alerts the program so that the erroneous block may be rewritten. When the tape is read later, the synchronizer will automatically detect the bad-spot pattern and alert the program so that it can bypass the erroneous block. In addition to the automatic check-read, a number of other checks (described in detail in table 3-2) are made on the information transferred to and from tape, on the operation of the synchronizer, and on the operation of the individual tape-handling units. If errors are detected by any of these checks, program-testable indicators are set and the program is alerted by means of an interrupt.

As a no-cost option, the synchronizer and tape-handling units may be modified so that the system can read and write tapes which are compatible with the UNIVAC 1107 Thin-Film Memory Computer and the UNIVAC 490 Real-Time System.

A second UNISERVO IIIA synchronizer may be added to the system so that up to 32 UNISERVO IIIA tape-handling units may be used. With this configuration, data is transferred between the synchronizers and the Central Processor through two read channels and two write channels, so that four read operations, or two reads and two writes, can be performed simultaneously.



HIGH-SPEED READER

A UNIVAC III System may contain one or more 80- or 90-column High-Speed Readers, each of which is controlled by an individual synchronizer which transfers data and control information to and from the Central Processor through one of the eight general-purpose channels.

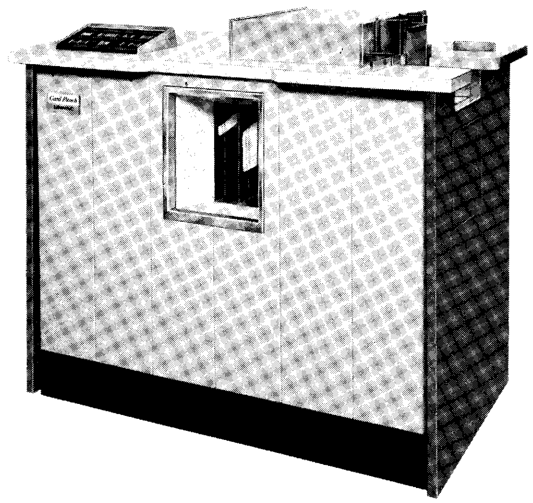
Each reader has a 2000-card input magazine, two read stations in which the cards are brush-sensed, and three 1000-card output stackers. Cards are fed from the input magazine under program control at a rate of 700 cards per minute and committed to continuously moving rollers which advance the cards through the two read stations. The number of holes sensed at the first read station is transferred to the synchronizer, where it is later compared with the number of holes sensed at the second read station; no data sensed at the first read station is transferred to memory. The card image from the second read station is transferred to the area of memory designated by the program.

If specified by the program, the card image from the second read station will be translated automatically by the synchronizer from Hollerith or Remington Rand 90-column card code* to the UNIVAC III character code as it is transferred to memory.

*The Hollerith (80-column) card code used with UNIVAC III Systems is given in table 4-1; the Remington Rand (90-column) card code is given in table 4-2.

After a card has been sensed at both read stations and its image has been transferred to memory, it is deposited in the output stacker designated by the program.

In addition to the hole-count check, a modulo-3 check is made on each word of the card image as it is received from the synchronizer by the Central Processor, and various other checks are made on the data sensed, on the operation of the reader, and on the operation of the synchronizer. These checks are described in detail in table 4-4. When an error occurs, program-testable indicators are set and the program is alerted by means of automatic program interrupt.



CARD-PUNCH UNIT

A Univac III System may contain one or more 80- or 90-column Card-Punch Units, each of which is controlled by an individual synchronizer which transfers data and control information to and from the Central Processor through one of the eight general-purpose channels.

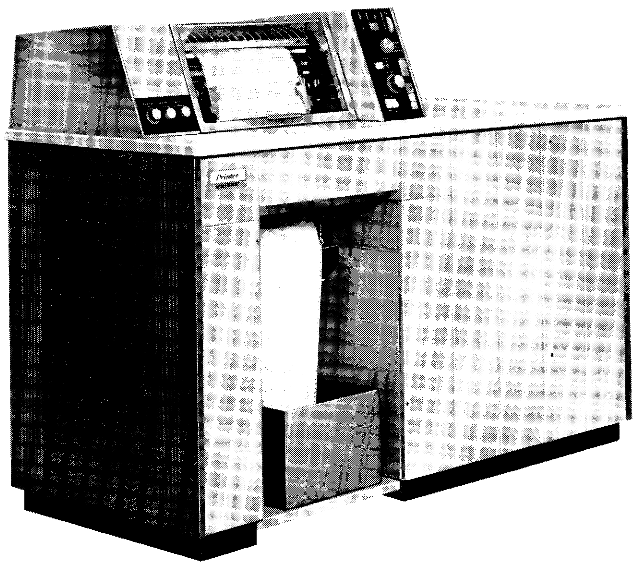
Each unit has a 1000-card input magazine, a punching mechanism, check-read brushes, and two 1000-card output stackers. Cards are fed from the input magazine under program control at a rate of 300 cards per minute. The cards are transported by clutched rollers past the punching mechanism, which punches the data from the area of memory designated by the program into the card.

If specified by the program, the data will be translated automatically from Univac III character code to Hollerith or Remington Rand 90-column card code before it is punched.

After the card is punched, it is moved past the check-read brushes where a hole-count check is made by the synchronizer; no data sensed by the check-read brushes is transferred to memory.

After the card has been brush-sensed, it is committed to continuously moving rollers and is deposited in the output stacker designated by the program.

In addition to the check-read operation, other checks are made on the data, and on the operation of the synchronizer and punch. These checks are described in detail in table 5-4. If any error or abnormal condition is detected, program-testable indicators are set and an input-output interrupt occurs.



HIGH-SPEED PRINTER

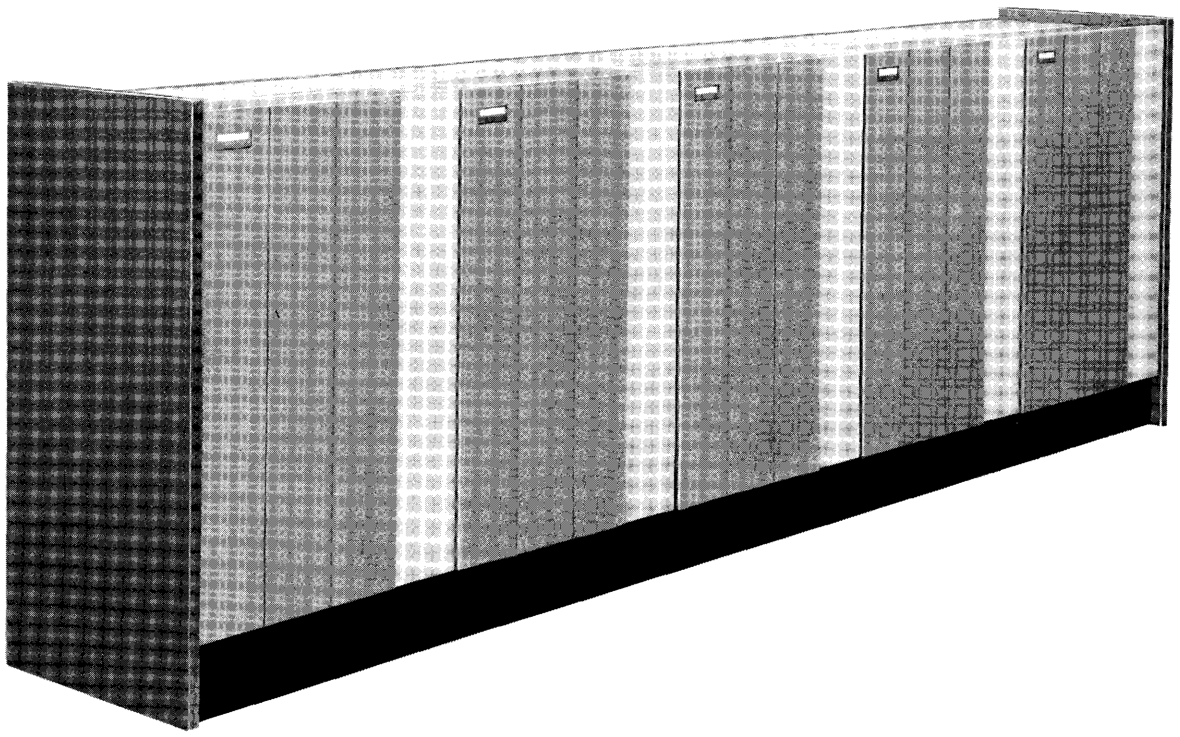
A Univac III System may contain one or more High-Speed Printers, each of which is controlled by an individual synchronizer which transfers data and control information to and from the Central Processor through one of the eight general-purpose channels.

Under program control, pre-edited data from memory is printed at a rate of 700 lines per minute for alphanumeric data, and 922 lines per minute for numeric data. Each line of print is 128 characters long, with a horizontal spacing of 10 characters per inch. Vertical spacing, which may be either 6 or 8 lines per inch, is controlled by the operator. The character set consists of 51 printing characters: A through Z, 0 through 9, and 15 punctuation marks and special symbols. Up to five carbon copies may be produced, using continuous sprocket-fed paper stock from 4 to 22 inches wide.

A number of automatic checks are made on the data, on the operation of the printer, and on the operation of the synchronizer. These checks are described in detail in table 6-3. If errors are detected by any of these checks, program-testable indicators are set and program interrupt occurs automatically.

SECTION II

CENTRAL PROCESSOR



The Central Processor controls the overall operation of the UNIVAC III System, under the direction of a stored program or programs. The programs consist of UNIVAC III words—the primary units of information with which the Central Processor operates.

Following a description of the UNIVAC III word, this section presents central-processor control elements and programming features, and a detailed description of each instruction in the central-processor list. Input-output instructions are described in the sections of this manual that pertain to the input-output units.

■ UNIVAC III WORD

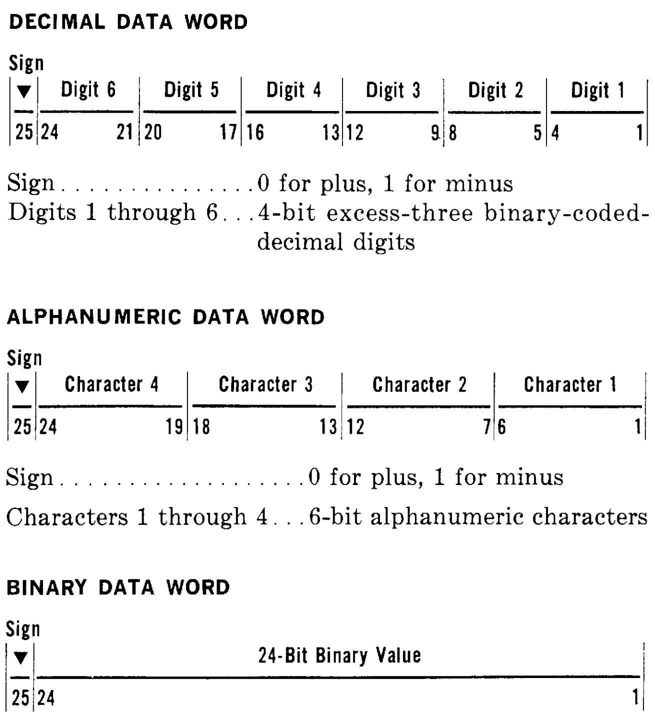
The UNIVAC III word, which is the basic unit of information in the system, is fixed in length and consists of 27 bits. Bit positions 1 through 24 represent data or an instruction in one of the formats described below. Bit position 25 of an instruction word indicates whether or not the instruction is to be executed with indirect addressing or field selection of the operand. Bit position 25 of a data word indicates the sign. Bit positions 26 and 27 are modulo-3 check bits used by the control circuitry to check word transfers and arithmetic operations; these bit positions contain the bits required to make the word, considered to be a 27-bit binary value, an integral multiple of 3. (Refer to Appendix A, *Modulo-3 Checking.*) Because the check bits are not program-accessible they are omitted from the descriptions of data and instructions in this manual.

DATA-WORD FORMATS

As shown in figure 2-1, data may be represented in the Central Processor in alphanumeric, decimal, or binary format. The sign of a data word is plus if bit position 25 is 0, and minus if bit position 25 is 1.

Decimal

A decimal data word comprises six decimal digits and a sign bit. Each digit is represented by four bits and is expressed in excess-three (XS-3) binary-coded-decimal code. All decimal arithmetic, shift, and conversion instructions require operands in this format.



Sign . . . . 0 for plus, 1 for minus
Digits 1 through 6 . . . 4-bit excess-three binary-coded-decimal digits

Sign . . . . 0 for plus, 1 for minus
Characters 1 through 4 . . . 6-bit alphanumeric characters

Sign . . . 0 for plus, 1 for minus

Figure 2-1. Data-Word Formats

Alphanumeric

An alphanumeric data word comprises four alphanumeric characters and a sign bit. Each character is represented by six bits: two bits for the zone, followed by four bits for the numeric portion. All alphanumeric shift and conversion instructions require operands in this format. The COBOL-FORTRAN character set for UNIVAC III Systems is shown in table 2-1. Information concerning alternate character sets is available on request.

Binary

A binary data word comprises 24 binary digits and a sign bit, expressing a value in the range plus or minus 2^{24}-1. All binary shift and arithmetic instructions require operands in this format.

Table 2-1. UNIVAC III Character Code

NP indicates a code which is not printed by the High-Speed Printer. Characters and functions in parentheses pertain to the Console Typewriter only.

Numeric Bits	Zone Bits			
	00	01	10	11
0000	Space	+	NP (5)	NP ($)
0001	&	)	*	(
0010	–	.	$	Comma ,
0011	0 (Ø)	NP (Carriage return and line feed)	NP (Ring bell)	' Apostrophe
0100	1	A	J	/
0101	2	B	K	S
0110	3	C	L	T
0111	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
1101	:	=	NP (2)	NP (:)
1110	<	NP (-)	NP (Horizontal tab)	NP (Form feed)
1111	>	NP (Ø)	NP (4)	NP (U)

Table 2-2. Instruction-Word Formats

Instruction	I/A 25	X 24–21	Op Code 20–15	AR/XO 14–11	m 10–1
General	Same for all instructions			AR	Operand address
Shift				AR	Shift count
Index Register				XO	Operand address
Indicator				Class Channel Indicator	Indicators
Initiate I-O				Channel	Function-specification address

INSTRUCTION-WORD FORMATS

Five basic formats are used for UNIVAC III instructions, in each of which the grouping of bits is the same. As shown in table 2-2, the functions of some bit groups vary with the type of instruction, while the function of other groups always is the same.

Bit Position 25: I/A—Indirect-Addressing/Field-Selection Option

Indirect addressing allows instructions to express operand addresses indirectly through intermediate control words. When it is used, the address in the instruction is that of an indirect-address (INAD) control word which contains the actual memory address to be referenced.

Field selection allows instructions to operate directly upon fields which are not isolated within computer words. When field selection is used, the address in the instruction is that of a field-select (FSEL) control word which specifies the boundaries and address of the operand.

Both indirect addressing and field selection are specified by a 1 in bit position 25 of the instruction word. The specific option to be used is determined by the format of the control word addressed.

Bit Positions 21 through 24: X—Index-Register Designation

All instructions are indexed as part of the basic instruction cycle. The contents of the designated index register are added to the *m* portion of the instruction word, giving a 15-bit field which is designated *m'*. The *X* field contains the binary designation, 0001 through 1111, of one of the 15 index registers. If 0000 is specified, no effective indexing takes place and *m'* equals *m*.

The instruction word in memory and the contents of the index register are not altered by the indexing process.

Bit Positions 15 through 20: Op Code—Operation Code

The operation code is designated in the instruction word by a 6-bit binary number, 000000 through 111111. For ease of representation it is expressed in this manual as a two-digit octal number, 00 through 77.

Bit Positions 11 through 14: AR/XO

The function of this field varies with the type of instruction. It is used to designate arithmetic registers, operand index registers, indicators*, groups of indicators, and input-output channels. When it is used to designate arithmetic registers, the number of registers designated indicates the number of words in the operand.

Bit Positions 1 through 10: m

This field is always indexed, giving *m'*, which is a 15-bit field whose function varies with the type of instruction, as indicated in table 2-2. When indirect addressing or field selection is specified, *m'* is the address of an INAD or FSEL control word which, in turn, serves the original function of *m'*.

*As used in this manual, the term *indicator* refers to a circuit element with which an external display may or may not be associated.

■ PROGRAMMING FEATURES

The design of the UNIVAC III System provides a number of programming features which increase the power of each instruction, provide efficient techniques of address modification and variable addressing, and allow more efficient use of internal storage and input-output media. These features are automatic indexing, indirect addressing, multiword operands, and field selection.

AUTOMATIC INDEXING

A UNIVAC III System contains either 9 or 15 index registers whose primary function is the modification of instruction references to memory addresses. As a part of the basic instruction cycle, the memory-address portion of the instruction word and the contents of one of the index registers are added; the result is the memory address to be referenced by the instruction. The address modification requires no additional machine cycles and does not alter the base address reference in the instruction word. By altering the contents of an index register, a routine can be applied to data anywhere in memory. The routine itself may be written with only relative addresses specified so that it may be relocated in memory as operational requirements dictate. The index registers, and the instructions associated with them, also provide efficient methods for program loop control and other counting functions. By means of the index registers and the automatic indexing process, these functions and the primary function of address modification can be achieved with a minimum number of program steps; thus program running times are reduced and the memory is used more efficiently.

Each index register contains a 15-bit unsigned binary number. The index register to be used is specified by its binary designation, 0001 through 1111, in bit positions 21 through 24 of the instruction word. As the instruction is being set up for execution, bit positions 1 through 10 of the instruction word, usually a memory address and referred to as *m*, and the 15-bit value in the specified index register, *X*, are added. The resulting 15-bit value is designated *m′*, and is the effective operand address. Carries beyond bit position 15 are ignored. The instruction word and the contents of the index register are not altered. If *X* is 0000, no effective indexing takes place and *m′* equals *m*.

Specifying one of the optional index registers (10 through 15) when it is not in the particular system in use will not cause an error; all binary 1's will be added to *m*, giving *m′*.

INDIRECT ADDRESSING

In some programming situations it is desirable to specify the location in which the address of an operand or instruction is stored, rather than specifying the address directly. This method of addressing operands or instructions is called indirect addressing. It is especially useful in table-handling functions, the manipulation of variable lines of coding, sorting and merging procedures, and other program functions which require flexible and efficient variable-addressing techniques. In the UNIVAC III System, indirect addressing, together with automatic indexing, provide the means for implementing such addressing techinques.

Indirect addressing (I/A) is specified by placing a 1 in bit position 25 of the instruction word. The address developed from the instruction word is the location of an indirect-address (INAD) control word. The memory address to be referenced by the instruction is then developed from the INAD control word. Expression of this memory address may be deferred through another level of indirect addressing by placing a 1 in bit position 25 of the INAD control word; the address developed from the INAD control word will then be the location of a second INAD control word. In this way, indirect addressing can be extended through as many levels as desired. This cascading process terminates when an INAD control word with a 0 in bit position 25 is accessed. The instruction will then be executed, using the memory address developed from the last INAD control word.

An INAD control word has the following format:

I/A	X	000	Unassigned	L-Address
25	24 21	20 18	17 16	15 1

I/A	Indirect-addressing/field-selection option
X	Binary designation of index register
Bit positions 18 through 20	Binary 0's, indicating an INAD control word
Bit positions 16 and 17	Unassigned
L-Address	If I/A is 1, the L-address is the unindexed address of another INAD control word (or field-select control word). If I/A is 0, the L-address is the unindexed address to be referenced by the instruction.

An instruction using indirect addressing is executed in the following manner:

1. The instruction is accessed and decoded. An indexed address, *m'*, is developed and bit 25 is examined.
2. If bit 25 is a 1, execution of the instruction is delayed and the control word in *m'* is accessed.
3. A new *m'* is developed from the accessed control word and bit position 25 of the control word is examined. If bit position 25 is a 1, steps 2 and 3 are repeated until a control word with a 0 in bit position 25 is accessed.
4. If bit position 25 is a 0, the control word is further examined. If bit positions 18 through 20 are binary 0's, the *m'* which has been developed from the control word is the memory address to be referenced by the instruction, and the instruction is then executed. (If bit positions 18 through 20 are not binary 0's, the control word is a field-select control word.)

An additional memory cycle is added to the instruction execution time for each INAD control word accessed.

MULTIWORD OPERANDS AND FIELD SELECTION

With the multiword-operand and field-selection features of the UNIVAC III System, fields in any of the three data formats, and ranging in size from one data unit (bit, digit, or character) to four words, can be processed directly with single instructions. With this facility, it is unnecessary to isolate packed fields, to align fields with computer words, or to process each word of a multiword field separately. Therefore, input-output media and memory can be used efficiently by packing fields within words and records. In addition, the need for fewer program steps results in reduced program running times and in conservation of memory.

Multiword Operands

The UNIVAC III System contains four 1-word arithmetic registers: AR1, AR2, AR3, and AR4. The arithmetic registers used in the execution of an instruction are designated by the bits in bit positions 11 through 14 of the instruction word as follows:

Register Designated	*Bit Position:* *14*	*13*	*12*	*11*
AR1	1	0	0	0
AR2	0	1	0	0
AR3	0	0	1	0
AR4	0	0	0	1

Through combinations of these designations, operands of from one to four words in length can be processed with a single instruction. The number and position of 1-bits in bit positions 11 through 14 control the size of the operand and its placement within the arithmetic registers. Arithmetic registers not specified will not be affected by the instruction (see figure 2-2).

Whether the arithmetic registers selected are adjacent or nonadjacent, they act as a single extended register. The words of the operand in memory, however, are always in adjacent memory locations. The memory location specified by the instruction (*m'*) contains the least significant word of the operand, which is transferred to or from the highest numbered arithmetic register designated, or is combined or

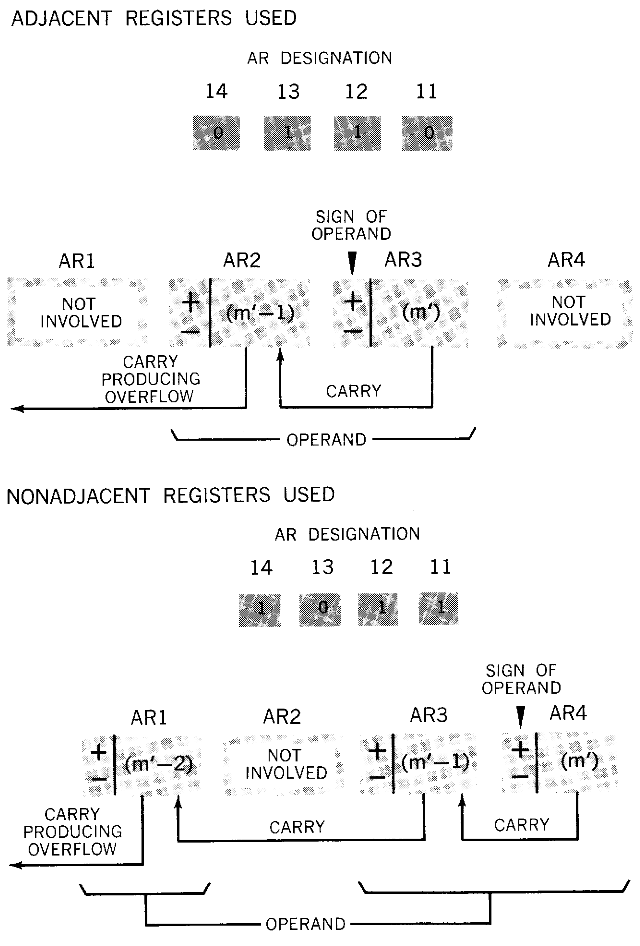


Figure 2-2. Examples of Multiword Operands

compared with the contents of that register. The balance of the operand in the lower ordered memory locations is processed similarly with the lower numbered arithmetic registers designated.

The sign of the least significant word of a multiword operand is taken as the sign of the entire operand; the signs of the other words are ignored. Each word of the result of an arithmetic operation has the same sign as that of the least significant word.

Carries or borrows between designated arithmetic registers are allowed to propagate. Only when there is a carry or borrow beyond the most significant arithmetic register designated will the arithmetic-overflow indicator be set.

Generally, when a multiword operand is specified, an additional machine cycle for each word beyond one should be added to the basic execution time.

Field Selection

When a field is split across computer words or is packed in a word with other fields, field selection may be used to designate only those bits, digits, or characters to be manipulated (see figure 2-3). The address and boundaries of the field are defined by a field-select (FSEL) control word.

Field selection is called for by a 1 in bit position 25 of the instruction word. The effective address specified by the instruction, *m′*, is the address of the FSEL control word. The format of the FSEL control word is as follows:

0	X	Left Boundary Bit	Right Boundary Bit	m
25	24 21	20 16	15 11	10 1

Bit 25	Always 0. The FSEL control word may not reference an indirect-address control word.
X	Binary designation of an index register
Left Boundary Bit	Position of the most significant bit of the field to be selected, expressed in excess-three binary code
Right Boundary Bit	Position of the least significant bit of the field to be selected, expressed in excess-three binary code
m	Unindexed memory address of the least significant portion of the field

NOTES

1. The most significant bit position of a word is designated 24 (11011); the least significant bit position is designated 1 (00100). The sign, bit position 25, may not be designated.
2. If the operand is multiword, the left boundary bit is in the most significant word, and the right boundary bit is in the least significant word. The arithmetic registers designated need not be adjacent, but the operand will always be selected from contiguous memory locations.
3. Sign bits are not selected, and the sign of each word of the operand is positive.
4. Portions of the words beyond the boundaries specified are treated as binary 0's. In decimal add and subtract operations they are treated as decimal 0's.
5. Carries and borrows are allowed to propagate up to bit position 24 of the most significant arithmetic register designated. An attempted carry or borrow beyond this bit position will set the arithmetic-overflow indicator.
6. If only one bit is to be selected, its position should be designated as the left and right boundary bits.
7. The FSEL control word may itself be indirectly addressed but does not use indirect addressing to express an operand address; therefore, bit position 25 must always be 0.
8. One machine cycle is added to the instruction execution time.

■ AUTOMATIC PROGRAM INTERRUPT

Automatic program interrupt is a method used to signal a program in progress that special conditions have arisen in the system which require action outside of the main processing chain. The three classes of special conditions which cause interrupt are, in ascending order of priority, *Input-Output, Contingency,* and *Processor Error.*

Examples of these are as follows:

Input-Output: Availability of a magnetic-tape unit for further use after successful completion of a write operation.

Contingency: A type-in by the operator.

Processor Error: Incorrect addressing of a word in memory.

When the conditions calling for interrupt are detected by the control circuitry, the contents of the control counter are stored in a fixed memory location associated with the particular class of interrupt. (Fixed memory locations are summarized in Appendix H.) Control is then transferred to another fixed location, the contents of which initiate a routine to determine the cause of the interrupt and to take suitable action. Following such action, the stored control-counter reading returns the program to the point at which interrupt occurred.

When a condition calling for interrupt arises, the following action takes place within the Central Processor. A program-testable indicator, or group of indicators, is set to identify the specific condition calling for interrupt. For each class of interrupt there is an *Interrupt-Mode Indicator (IMI)*. These indicators are automatically set, reset, and tested by the control circuitry and are not accessible to the program. When an IMI is set, interrupts of the same class or of classes with a lower priority are inhibited. Interrupts of a higher class, and the setting of specific indicators when interrupt conditions arise, are not inhibited.

As shown in figure 2-4, at the completion of each instruction, classes of interrupt indicators are automatically probed in descending order of priority. If

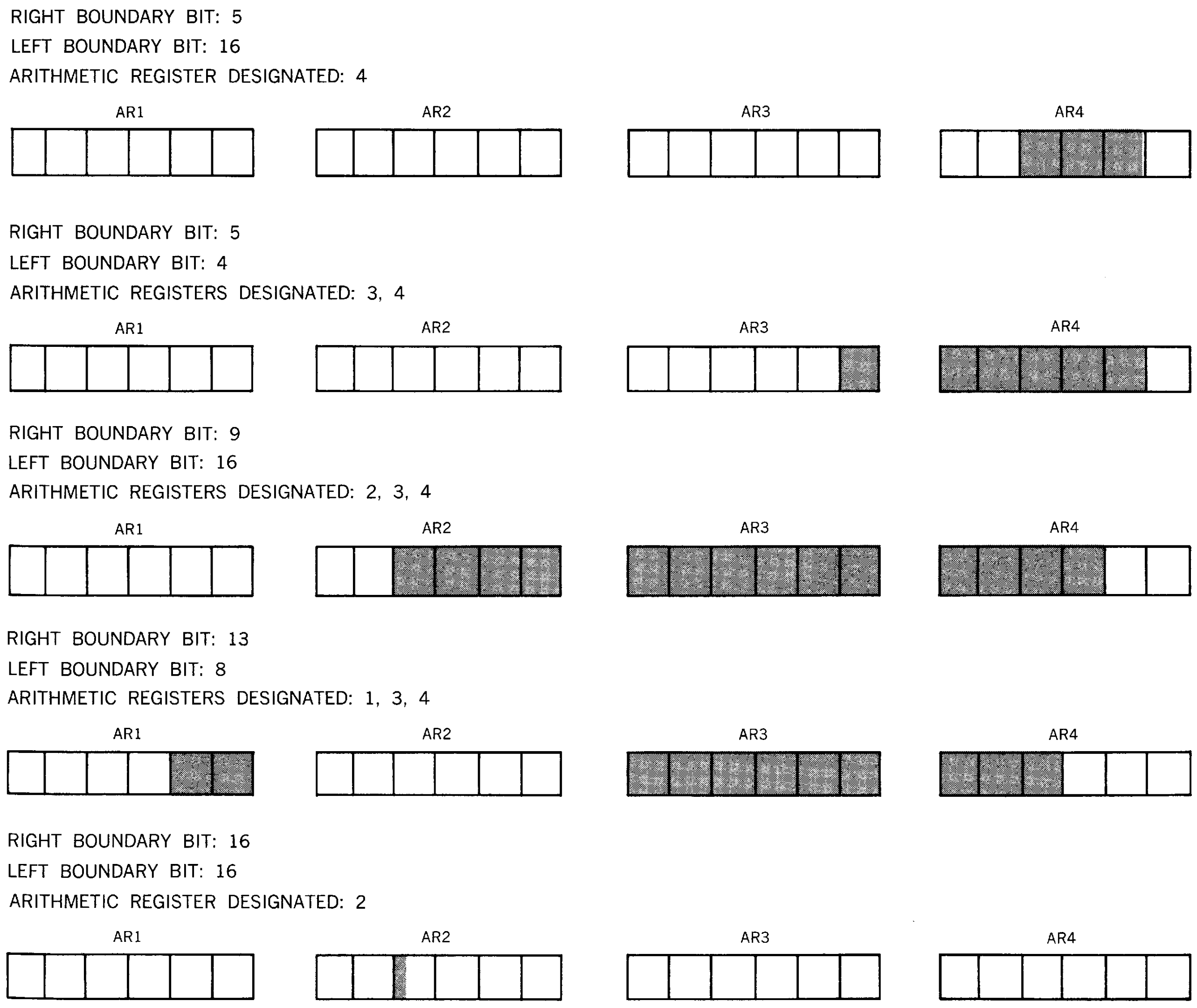


Figure 2-3. Examples of Field-Selected Operands

any specific indicator is found to be set, and if the IMI for its class or for a class of higher priority is not set, interrupt takes place. At this time the IMI for the class of interrupt occurring is automatically set in order to prevent subsequent interrupts of the same or lower classes from taking place while the current interrupt is being processed.

The control-counter reading, which is the address of the next instruction in the interrupted program, is stored in a fixed memory location associated with the class of interrupt taking place. Bit positions 1 through 15 of the memory location contain the control-counter reading; bit positions 16 through 25 are cleared to binary 0's. This memory location may be used later as an indirect-address (INAD) control word to return control to the interrupted program.

Control is transferred to another fixed memory location associated with the class of interrupt. The locations associated with each class of interrupt are as follows:

Class of Interrupt	*Control-Counter Reading Stored in*	*Control Transferred to*
Processor Error	0016	0017
Contingency	0018	0019
Input-Output	0020	0021

Control is thus transferred to one of three locations where a routine to determine the exact nature of the interrupt is initiated. This determination is made by testing the specific indicators associated with the class of interrupt. When the cause of the interrupt has been identified and appropriate action taken, the specific indicators may be reset and control transferred to the address specified by the stored control-counter reading. The instruction that resets the specific indicators (RPE, RCI, or RIO) automatically resets the interrupt-mode indicator for the class of interrupt involved. Interrupts of all classes are then inhibited until the instruction following the reset instruction is completed.

The processing of all interrupts and interrupt conditions is one of the functions of the executive monitoring routines, and of the input-output subroutines provided by the assembly systems. Detailed descriptions of these routines are provided in separate manuals.

INPUT-OUTPUT INTERRUPT

At the completion of each instruction, if no processor error or contingency indicators are found to be set, the input-output interrupt indicators are automatically tested as a group. If any input-output indicator is set, and no interrupt-mode indicator is set, input-output interrupt will occur. The input-output interrupt-mode indicator (IOIMI) is then set; the contents of the control counter are stored in memory location 0020; and control is transferred to memory location 0021.

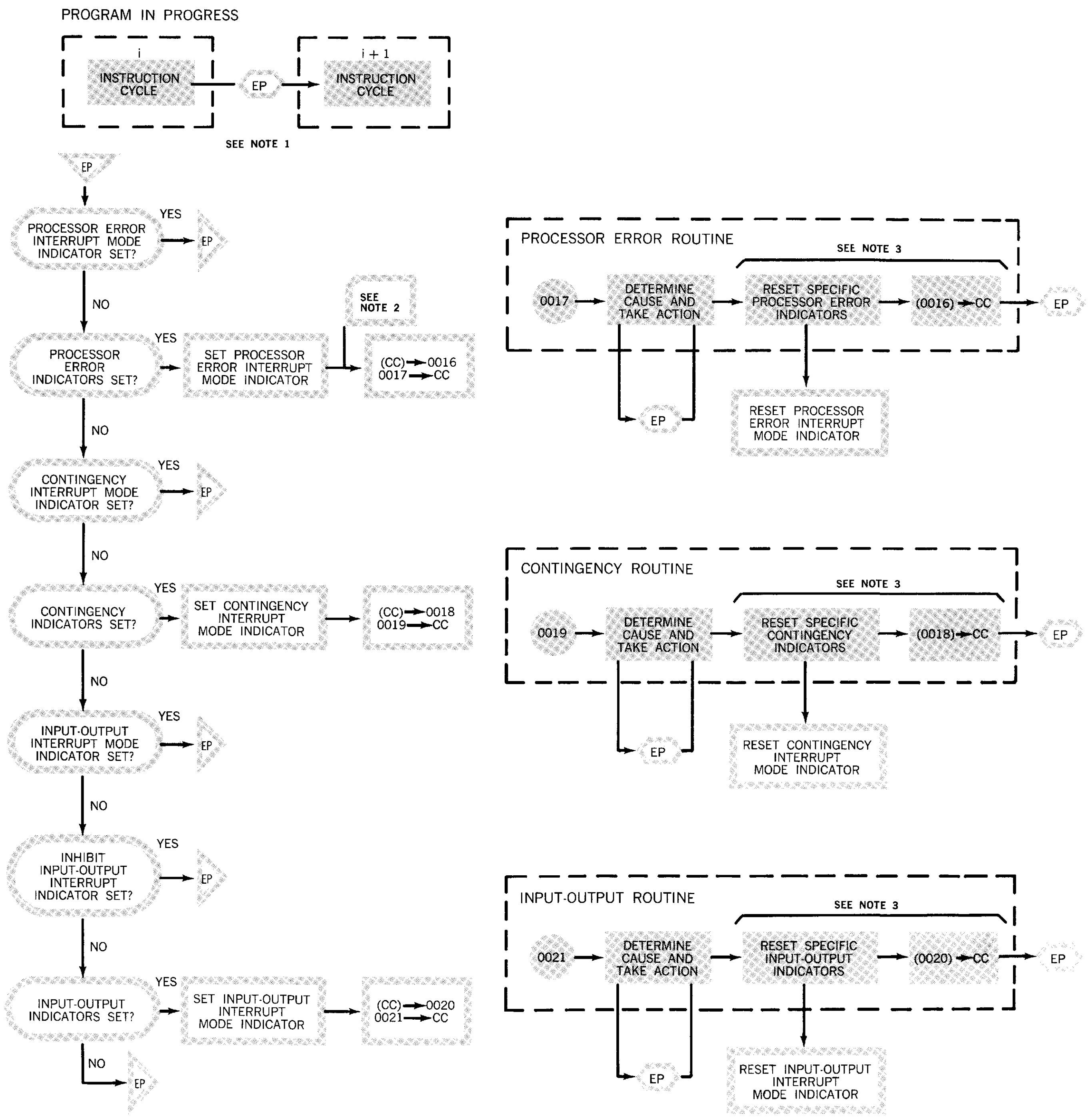
When the IOIMI is set, the further setting of specific input-output indicators will not be inhibited; however, a second input-output interrupt will not occur until the completion of the instruction that follows the reset input-output indicators (RIO) instruction. If any processor error or contingency conditions arise while the IOIMI is set, interrupts of these classes will occur.

Input-output interrupt occurs when input-output interrupt indicators have been set due to one of the following conditions:

Error or fault conditions have prevented the proper execution of an input-output operation and corrective action, either by program steps or operator intervention, must be taken.

An input-output operation has been initiated in the High-Speed Reader or Card-Punch Unit, or has been successfully completed, in magnetic-tape units or the High-Speed Printer. Interrupt upon initiation occurs when a function specification has been accessed from the standby location of the channel by the synchronizer for the input-output device; interrupt upon successful completion can occur only when an input-output operation has been completed, and no error or fault indicators have been set. Interrupt upon initiation or successful completion indicates that the input-output synchronizer concerned is ready to accept another function specification; this interrupt occurs only if bit position 16 of the function specification contains a 1.

If interrupt is not called for in the function specification, the initiation or successful-completion indicator will not be set upon initiation or successful completion, but interrupt will occur if one or more error or fault indicators are set and no interrupt-mode indicator is set. If desired, all input-output interrupts may be inhibited by setting the inhibit input-output-interrupt indicator (PIO instruction). While this indicator is set, specific indicators may be set, but no actual input-output interrupts can occur until it is reset (AIO instruction).


PROGRAM IN PROGRESS
i
INSTRUCTION CYCLE
EP
i + 1
INSTRUCTION CYCLE
SEE NOTE 1
EP
PROCESSOR ERROR INTERRUPT MODE INDICATOR SET?
YES
EP
NO
PROCESSOR ERROR INDICATORS SET?
YES
SET PROCESSOR ERROR INTERRUPT MODE INDICATOR
SEE NOTE 2
(CC)→0016
0017→CC
NO
CONTINGENCY INTERRUPT MODE INDICATOR SET?
YES
EP
NO
CONTINGENCY INDICATORS SET?
YES
SET CONTINGENCY INTERRUPT MODE INDICATOR
(CC)→0018
0019→CC
NO
INPUT-OUTPUT INTERRUPT MODE INDICATOR SET?
YES
EP
NO
INHIBIT INPUT-OUTPUT INTERRUPT INDICATOR SET?
YES
EP
NO
INPUT-OUTPUT INDICATORS SET?
YES
SET INPUT-OUTPUT INTERRUPT MODE INDICATOR
(CC)→0020
0021→CC
NO
EP
PROCESSOR ERROR ROUTINE
SEE NOTE 3
0017
DETERMINE CAUSE AND TAKE ACTION
RESET SPECIFIC PROCESSOR ERROR INDICATORS
(0016)→CC
EP
EP
RESET PROCESSOR ERROR INTERRUPT MODE INDICATOR
CONTINGENCY ROUTINE
SEE NOTE 3
0019
DETERMINE CAUSE AND TAKE ACTION
RESET SPECIFIC CONTINGENCY INDICATORS
(0018)→CC
EP
EP
RESET CONTINGENCY INTERRUPT MODE INDICATOR
INPUT-OUTPUT ROUTINE
SEE NOTE 3
0021
DETERMINE CAUSE AND TAKE ACTION
RESET SPECIFIC INPUT-OUTPUT INDICATORS
(0020)→CC
EP
EP
RESET INPUT-OUTPUT INTERRUPT MODE INDICATOR

NOTES:

1. Operations within the dotted lines are performed by program steps. Operations outside of the dotted lines are automatically performed by the control circuitry. EP is performed by the control circuitry when the ending pulse is generated at the completion of each instruction (i). If interrupt does not occur, the instruction (i+1) in the location specified by the contents of the control counter is executed. If interrupt occurs, the address of i+1 (contents of the control counter) is stored and control is transferred to the appropriate interrupt routine.
2. If further processor errors occur after the processor error interrupt mode indicator has been set, the Central Processor will stop.
3. Interrupts of all classes are inhibited during this time.

Figure 2-4. Automatic Program Interrupt, Functional Diagram

CONTINGENCY INTERRUPT

At the completion of each instruction, if no processor-error indicators are found to be set, the contingency indicators are automatically tested as a group. If any contingency indicator is set, and neither the processor-error interrupt-mode indicator nor the contingency interrupt-mode indicator (CIMI) is set, contingency interrupt will occur. The CIMI is then set, the contents of the control counter are stored in memory location 0018, and control is transferred to memory location 0019.

When the CIMI is set, the setting of specific contingency or input-output indicators will not be inhibited; however, contingency or input-output interrupt cannot occur until the instruction that follows the reset-contingency-indicators (RCI) instruction is completed. If any processor-error condition arises while the CIMI is set, a processor-error interrupt will occur.

The conditions which call for contingency interrupt are of two classes, external and internal. External contingency conditions, which are associated with the operator's console and console typewriter, facilitate programmed operator-system communication:

Typewriter Interrupt: A character has been typed in or typed out.

Keyboard Request: The operator is requesting that the console-typewriter keyboard be activated so that a message may be typed.

Keyboard Release: The operator has completed a type-in and the keyboard is de-activated.

Contingency Stop: The operator has pressed the PROGRAM STOP button, requesting that the program bring the system to a halt.

Internal contingency conditions are as follows:

Arithmetic Overflow: An add or subtract operation has resulted in a carry beyond the most significant arithmetic register designated, or a divide operation has been attempted in which the absolute value of the divisor is not greater than the absolute value of the dividend.

Clock Power Disrupted: The addressable clock has been accessed (LT instruction) while in a non-ready condition due to a previous removal of clock power.

Invalid Op Code: The execution of an instruction whose operation code is not in the repertoire for the system has been attempted.

PROCESSOR-ERROR INTERRUPT

At the completion of every instruction, the processor-error indicators are automatically tested as a group. If any indicator is set, and the processor-error interrupt-mode (PEIMI) indicator is not set, interrupt will occur. The PEIMI is then set, the control-counter reading is stored in memory location 0016, and control is transferred to memory location 0017. If further processor errors occur after the PEIMI has been set, the Central Processor will stop.

When the PEIMI is set, the setting of specific contingency and input-output indicators will not be inhibited. However, interrupts of these classes cannot occur until the instruction that follows the reset-processor-errors (RPE) instruction has been completed.

There are two types of processor errors: memory-address errors and modulo-3 errors. A memory-address error indicator is set when the memory location accessed is not that which was addressed, or when a location in a memory module not included in the particular system in use is addressed.

A modulo-3 error indicator is set when a word being accessed does not have the correct modulo-3 check bits (bit positions 26 and 27), or if the results of certain instructions in the adder fail to pass the modulo-3 check. (Refer to Appendix A, *Modulo-3 Checking.*) Attempting to access a location in a memory module not included in the particular system in use will also set a modulo-3 error indicator.

■ CONTROL UNIT

This section describes the main functions of those components of the control unit which are involved in the execution of instructions. The execution cycles of a single-word operand instruction, a multiword-operand instruction, and an instruction using field selection or indirect addressing, are also described.

CENTRAL-PROCESSOR REGISTER

The central-processor register is the primary working register used in central-processor control operations; it performs all shifting and data manipulation required by the control circuitry. Data and instructions read from memory by the Central Processor are stored in this register; during arithmetic operations it contains the secondary operand from memory (addend, subtrahend, multiplicand, or divisor).

MEMORY-ADDRESS ADDER

The memory-address adder, which is separate from the arithmetic adder, is used by the control circuitry to compute the memory addresses needed to transfer information between memory, the Central Processor, and the input-output synchronizers. The adder is also used as a data-transfer path during the execution of certain instructions.

STORAGE-ADDRESS UNIT

The storage-address unit consists of several registers which supply addressing information for central-processor and input-output operations; these registers are described in the following paragraphs.

Index Registers (X)

The 9 or 15 index registers in the system are used primarily to modify instruction references to memory addresses. The index registers and the indexing process are described in detail under the heading *Automatic Indexing.*

Control Counter (CC)

This register is used by the control circuitry to sequence instructions. It contains a 15-bit value which is the address of the instruction being executed. During the final execution phase of the instruction, the contents of the control counter are normally incremented by 1 in the memory-address adder to give the address of the next instruction. Certain instructions will cause the address in the control counter to be incremented by 2 or to be replaced with a new address from the instruction word.

Memory-Address Register (MAR)

The memory-address register contains the address of the memory location to or from which information is to be transferred.

Memory-Address Counters (MAC)

Each input-output synchronizer has an associated 15-bit MAC which contains the memory-address information required by the synchronizer to perform input-output operations.

Tape Control-Word Registers (TCWR)

A one-word TCWR, which holds the tape control words used in scatter-read and gather-write (SCAT) operations, is associated with each UNISERVO IIIA channel.

INDICATOR UNIT

The indicator unit contains the program-testable indicators described below. When the indicator tested is found to be reset, the next instruction in sequence is accessed. When the indicator tested is found to be set, control is transferred to the address specified by the instruction.

Sense Indicators 1 through 8

Each of the sense indicators, which may be used for program control, may be tested, set, or reset by the program.

Inhibit-Input-Output-Interrupt Indicator

This indicator may be set, reset, or tested by the program. When it is set, input-output interrupt is inhibited; however, the setting of specific input-output indicators is not inhibited.

Comparison Indicators

The three comparison indicators, high, low, and equal, are set on the basis of comparisons of operands in the arithmetic or index registers with operands in memory. The equal indicator is also set and reset by add and subtract instructions. If the result of an addition or subtraction is zero, the equal indicator is set. If the result is not zero, the equal indicator is reset.

Arithmetic-Register-Sign Indicators

These indicators, one for the sign of the contents of each arithmetic register, may be tested by the program. The sign indicator is set when the contents of the arithmetic register are positive, and reset when the contents of the arithmetic register are negative.

EXECUTION CYCLE WITH SINGLE-WORD OPERAND

During the final execution phase of an instruction, the instruction address is transferred from the control counter to the memory-address adder. Depending on the type of instruction and its result, the address in the memory-address adder is incremented by 1, incremented by 2, or replaced by a new address from the instruction word. The value thus developed, which is the address of the next instruction to be executed, is transferred to the control counter and the memory-address register.

The instruction word in the location specified by the memory-address register is read from memory into the central-processor register. At the same time, the

X and *AR* fields of the instruction word are interpreted to select the index register and arithmetic register to be used in executing the instruction.

In the next phase of the instruction cycle, the contents of the selected index register and the *m* field of the instruction word are added in the memory-address adder; the result is *m'*, the 15-bit memory address to be referenced by the instruction. As *m'* is being developed, the op-code field generates the function signals necessary to execute the instruction.

The *m'* value is transferred to the memory-address register. If the instruction is a register-to-memory transfer, the contents of the selected arithmetic register are transferred to the location specified by the memory-address register. If the instruction requires an operand from memory, the contents of the location specified by the memory-address register are transferred through the central-processor register to the arithmetic unit where they are processed. At the same time, the address of the next instruction is being computed from the contents of the control counter.

EXECUTION CYCLE WITH MULTIWORD OPERAND

If the operand is multiword, the cycle described above is executed for the least significant word of the operand, which is in either *m'* or the highest designated arithmetic register. However, development of the next instruction address is deferred until the most significant word of the operand is being transferred to or from memory.

While the least significant word of the operand is being processed, the *m'* address in the memory-address register is decremented by 1 in the memory-address adder, and the result is returned to the memory-address register. The value thus developed is the address of the next operand word.

If the instruction is a register-to-memory transfer, the contents of the next lower designated arithmetic register are transferred to the location specified by the memory-address register. Otherwise, the contents of the location specified by the memory-address register are transferred through the central-processor register to the arithmetic unit, where they are processed with the contents of the next lower designated arithmetic register.

This process is repeated until the last word of the operand is being transferred to or from memory. At this time, the address of the next instruction is computed.

EXECUTION CYCLE WITH FIELD SELECTION AND INDIRECT ADDRESSING

After the instruction word has been accessed from memory, but before *m'* has been developed, bit position 25 of the instruction word is examined. If it contains a 0-bit, *m'* is an operand address; if it contains a 1-bit, *m'* is the address of a field-select (FSEL) control word or an indirect-address (INAD) control word. The control word is transferred from memory to the central-processor register and its *X* field is interpreted to select the index register to be used.

Bit position 25 of the control word then is examined. If it contains a 1-bit, the control word is an INAD and the next word to be accessed is a control word which must in turn be analyzed. If bit position 25 contains a 0-bit, bit positions 18 through 20 of the control word are examined. If they do not contain all 0-bits, the control word is a FSEL and the next word to be accessed is an operand on which field selection will be performed as it is transferred to the arithmetic unit. If bit positions 18 through 20 contain all 0-bits, the control word is an INAD and the next word to be accessed is an operand.

The address portion of the control word in the central-processor register and the contents of the selected index register are now added in the memory-address adder, to produce *m'*, the address of the next word to be accessed.

If field selection was specified, the contents of *m'* are transferred through the central-processor register to the arithmetic unit, the portions of the operand outside the boundaries designated by the control word being replaced with binary 0's.

If the original control word was an INAD calling for another control word, the contents of *m'* are analyzed and a new *m'* is developed. This process is repeated until a control word with a 0 in bit position 25 is accessed and the operand address can be developed.

If the control word was an INAD with a 0 in bit position 25, *m'* is an operand address. If the instruction is a register-to-memory transfer, the contents of the selected arithmetic register are transferred to the memory location specified by *m'*. If the instruction requires an operand from memory, the contents of *m'* are transferred through the central-processor register to the arithmetic unit.

In the final phase of transferring the operand to or from memory, the address of the next instruction is computed from the contents of the control counter.

UNIVAC III INSTRUCTION LIST

The instruction list of the UNIVAC III System is arranged in the following pages by functional category. Each category is introduced by a brief description of the general characteristics of the instructions in that category.

The function specifications associated with each input-output device are listed in table 2-3 (page 2-30). For a detailed description of each function specification, refer to the section of this manual that describes the associated input-output device.

INSTRUCTION-LIST FORMAT

Each instruction is described in accordance with the following format:

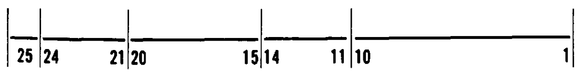
FUNCTION—Mnemonic Code

Operation: Symbolic representation of instruction description

Op Code: Operation code, expressed as two octal digits

Cycles: Basic execution time, expressed in 4-microsecond memory cycles

Description: Definition of function of the instruction



Instruction format: Explanation of the function of each part of the instruction word

NOTES

Considerations in instruction use and further definition of function.

SYMBOLS AND ABBREVIATIONS

The following symbols and abreviations are used in the instruction formats:

( )	the contents of
| |	the absolute value of
→	transfer to
:	compare with
AR	arithmetic register(s)
ARi	arithmetic register(s) designated
CC	control counter
I/A	indirect-addressing/field-selection option
LSB	least significant bits
m	unindexed 10-bit memory address
m′	15-bit memory address developed from *m* by indexing and by indirect addressing
Op Code	operation code
XO	index register
XOi	designated index register to be modified
X	index register used to modify *m*

Other abbreviations will be defined as they are introduced.

■ OPERAND-TRANSFER INSTRUCTIONS

The following instructions transfer data between memory and the four arithmetic registers. The contents of the sending memory location or arithmetic register are not altered by the transfer, and the original contents of the receiving location or register are replaced with the operand transferred.

All of these instructions may use indirect addressing and multiword operands, but only those which transfer data from memory to the arithmetic registers may use field selection.

LOAD—L

Operation: (m′) → ARi
Op Code: 12
Cycles: 2

Description: Transfer the operand from the indexed memory location(s) to the arithmetic register(s) designated.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing/field-selection option
X . . . . . Binary designation of index register
AR . . . Designation of arithmetic register(s)
m . . . . . Unindexed address of operand

NOTES

1. The contents of the memory location(s) accessed are not altered.
2. Indirect addressing, field selection, and multiword operands may be used.

LOAD WITH CHANGED SIGN—LCS

Operation: −(m′) → ARi
Op Code: 13
Cycles: 2

Description: Transfer the operand from the indexed memory location(s) to the designated arithmetic register(s), reversing the sign of each word.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing/field-selection option. See note 3.
X . . . . . Binary designation of index register
AR . . . Designation of arithmetic register(s)
m . . . . . Unindexed address of operand

NOTES

1. The contents of the memory location(s) accessed are not altered.
2. The sign of each word of a multiword operand is treated separately.
3. When field selection is used with this instruction, the contents of the designated arithmetic register(s) will always be negative because field selection normally makes the sign of the operand from memory positive.
4. Indirect addressing, field selection, and multiword operands may be used.

EXTRACT INTO REGISTER—EXT

Operation: Extract (m′) → ARi
Op Code: 14
Cycles: 3 (includes 1 cycle for field selection)

Description: Replace consecutive bits of the designated arithmetic register(s) with bits from the corresponding positions of the memory location(s) specified.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A . . . Should be 1 to specify field selection. See note 3.
X . . . . . Binary designation of index register
AR . . . Designation of arithmetic register(s)
m . . . . . Unindexed address of operand

NOTES

1. The bit positions to be replaced and the operand address are specified in a field-select (FSEL) control word.
2. The sign of the arithmetic register(s) will not be affected.
3. Without field selection, the instruction will function as a load (L) instruction, except that the sign of the arithmetic register(s) will not be affected.
4. The contents of the memory location(s) accessed, and bits of the operand in the arithmetic register(s) outside the limits specified, remain unchanged.
5. Indirect addressing and multiword operands may be used.

STORE—ST

Operation: (ARi) → m′
Op Code: 10
Cycles: 2

Description: Transfer the contents of the arithmetic register(s) designated to the indexed memory location(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing option

X . . . . . Binary designation of index register

AR . . . Designation of arithmetic register(s)

m . . . . . Unindexed address of operand

NOTES

1. Contents of the arithmetic registers are not altered.
2. Indirect addressing and multiword operands, but not field selection, may be used.

STORE WITH CHANGED SIGN—STCS

Operation: $-(\text{ARi}) \rightarrow \text{m}'$

Op Code: 11

Cycles: 2

Description: Transfer the contents of the designated arithmetic register(s) to the indexed memory location(s), reversing the sign of each word of the operand.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing option

X . . . . . Binary designation of index register

AR . . . Designation of arithmetic register(s)

m . . . . . Unindexed address of operand

NOTES

1. Contents of the arithmetic registers are not altered.
2. Indirect addressing and multiword operands, but not field selection, may be used.

■ ARITHMETIC INSTRUCTIONS

The following instructions perform arithmetic operations on signed binary or decimal values in the adder. The primary input to the adder comes from one or more of the arithmetic registers, and the secondary input is taken from the memory location, or locations, specified by the instruction. The result of the operation appears in one place only: it either replaces the primary operand in the arithmetic registers, or it is placed in a higher designated arithmetic register, leaving the primary operand undisturbed.

If the result of an arithmetic operation is multiword, the signs of the individual words will be identical. The exception to this is division, in which the sign of the remainder will be that of the dividend, regardless of the sign of the quotient.

The arithmetic-overflow indicator will be set, causing contingency interrupt, if an add or subtract operation results in a carry beyond the most significant arithmetic register designated, or if a divide operation is attempted with a divisor which is not greater in absolute value than the dividend.

All arithmetic instructions, with the exception of multiplication and division, may use field selection and multiword operands.

ADD (DECIMAL)—A

Operation: $(\text{ARi}) + (\text{m}') \rightarrow \text{ARi}$

Op Code: 20

Cycles: 2

Description: Algebraically add in decimal (4-bit) code the operand in the indexed memory location(s) to the operand in the designated arithmetic register(s), placing the sum in the same arithmetic register(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing/field-selection option

X . . . . . Binary designation of index register

AR . . . Designation of arithmetic register(s)

m . . . . . Unindexed address of operand

NOTES

1. Binary 0's (0000) in either operand will be treated as decimal 0's (0011). All 0's in the sum will be decimal. (Refer to Appendix D for a description of the treatment of other non-numeric codes.)
2. The equal comparison indicator (ECI) will be set if the sum is zero. If the sum is not zero, the ECI will be reset.
3. Indirect addressing, field selection, and multiword operands may be used.

ADD HIGHER (DECIMAL)—AH

Operation: $(\text{ARi}) + (\text{m}') \rightarrow \text{ARi}'$, where $\text{i}' > \text{i}$

Op Code: 22

Cycles: 2

Description: Algebraically add in decimal (4-bit) code the operand in the indexed memory location(s) to the operand in the lower designated arithmetic register(s), placing the sum in the higher designated arithmetic register(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . .	Indirect-addressing/field-selection option
X.	Binary designation of index register
AR. . .	Designation of arithmetic registers. See note 3.
m.	Unindexed address of operand

NOTES

1. The contents of ARi will be undisturbed except for the replacement of binary 0's (0000) with decimal 0's (0011).
2. Binary 0's in either operand will be treated as decimal 0's. All 0's in the sum will be decimal. (Refer to Appendix D for a description of the treatment of other non-numeric codes.)
3. For single-word operands, the designations of *i* and *i′* are as follows:

i	*i′*
1	2, 3, or 4
2	3 or 4
3	4
Cannot be 4	—

   For multiword operands, *i* must be 1 and 2, and *i′* must be 3 and 4.
4. The equal comparison indicator (ECI) will be set if the sum is zero. If the sum is not zero, the ECI will be reset.
5. Indirect addressing, field selection, and multiword operands may be used.

SUBTRACT (DECIMAL)—S

Operation: (ARi) − (m′) → ARi

Op Code: 21

Cycles: 2

Description: Algebraically subtract in decimal (4-bit) code the operand in the indexed memory location(s) from the operand in the designated arithmetic register(s), placing the difference in the same arithmetic register(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . .	Indirect-addressing/field-selection option
X.	Binary designation of index register
AR. . .	Designation of arithmetic register(s)
m.	Unindexed address of operand

NOTES

1. Binary 0's (0000) in either operand will be treated as decimal 0's (0011). All 0's in the difference will be decimal. (Refer to Appendix D for a description of the treatment of other non-numeric codes.)
2. The equal comparison indicator (ECI) will be set if the difference is zero; if the difference is not zero, the ECI will be reset.
3. Indirect addressing, field selection, and multiword operands may be used.

SUBTRACT HIGHER (DECIMAL)—SH

Operation: (ARi) − (m′) → ARi′, where i′ > i

Op Code: 23

Cycles: 2

Description: Algebraically subtract in decimal (4-bit) code the operand in the indexed memory location(s) from the operand in the lower designated arithmetic register(s), placing the difference in the higher designated arithmetic register(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . .	Indirect-addressing/field-selection option
X.	Binary designation of index register
AR. . .	Designation of arithmetic registers. See note 3.
m.	Unindexed address of operand

NOTES

1. The contents of ARi will be undisturbed except for the replacement of binary 0's (0000) with decimal 0's (0011).
2. Binary 0's in either operand will be treated as decimal 0's. All 0's in the difference will be decimal. (Refer to Appendix D for the description of the treatment of other non-numeric codes.)
3. For single-word operands, the designations of *i* and *i′* are as follows:

i	*i′*
1	2, 3, or 4
2	3 or 4
3	4
Cannot be 4	—

   For multiword operands, *i* must be 1 and 2, and *i′* must be 3 and 4.
4. The equal comparison indicator (ECI) will be set if the difference is zero; if the difference is not zero, the ECI will be reset.
5. Indirect addressing, field selection, and multiword operands may be used.

MULTIPLY (DECIMAL)—M

Operation: (m′) × (AR1) → AR2, AR3

Op Code: 30

Cycles: 19 to 31, depending on multiplier digits

Description: Algebraically multiply in decimal (4-bit) code the contents of the indexed memory location (multiplicand) by the contents of arithmetic register 1 (multiplier), placing the six most significant digits of the product in arithmetic register 2 and the six least significant digits in arithmetic register 3.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing option

X. . . . . Binary designation of index register

AR. . . 1110—Designation of arithmetic registers 1, 2, and 3

m. . . . . Unindexed address of multiplicand

NOTES

1. The multiplier, multiplicand, and arithmetic register 4 remain unchanged at the completion of the operation.
2. Zeros in both operands must be decimal (0011).
3. Multiword operands may not be used; however, a 12-digit product is developed.
4. Refer to Appendix B for a description of the method of computing execution time.
5. Indirect addressing, but not field selection, may be used.

DIVIDE (DECIMAL)—D

Operation: (AR1, AR2) ÷ (m′); quotient → AR2
remainder → AR1

Op Code: 31

Cycles: 17 to 35, depending on quotient digits

Description: Algebraically divide in decimal code the contents of arithmetic registers 1 and 2 (dividend) by the contents of the indexed memory location (divisor), placing the quotient in arithmetic register 2 and the remainder in arithmetic register 1.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing option

X. . . . . Binary designation of index register

AR. . . 1100—Designation of arithmetic registers 1 and 2

m. . . . . Unindexed address of divisor

NOTES

1. Zeros in the divisor and the dividend must be decimal (0011).
2. If the absolute value of the divisor (m′) is not greater than that of the contents of arithmetic register 1, the arithmetic-overflow indicator will be set and a contingency interrupt will result. The contents of arithmetic register 1 will be unusable, and an attempt to transfer them to memory may result in a modulo-3 processor error.
3. The sign of the remainder will be that of the dividend.
4. Refer to Appendix C for a description of the method of computing the execution time.
5. Indirect addressing, but not field selection, may be used.

BINARY ADD—BA

Operation: (ARi) + (m′) → ARi

Op Code: 24

Cycles: 2

Description: Algebraically add in binary code the operand in the indexed memory location(s) to the operand in the designated arithmetic register(s), placing the sum in the same arithmetic register(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing/field-selection option

X. . . . . Binary designation of index register

AR. . . Designation of arithmetic register(s)

m. . . . . Unindexed address of operand

NOTES

1. The equal comparison indicator (ECI) will be set if the sum is zero. If the sum is not zero, the ECI will be reset.
2. Indirect addressing, field selection, and multiword operands may be used.

BINARY ADD HIGHER—BAH

Operation: (ARi) + (m′) → ARi′, where i′ > i

Op Code: 26

Cycles: 2

Description: Algebraically add in binary code the operand in the indexed memory location(s) to the operand in the lower designated arithmetic register(s), placing the sum in the higher designated arithmetic register(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing/field-selection option

X. . . . . Binary designation of index register

AR. . . Designation of arithmetic registers. See note 2.

m. . . . . Unindexed address of operand

NOTES

1. The contents of ARi remain unchanged at the end of the operation.
2. For single-word operands, the designations of *i* and *i'* are as follows:

i	*i'*
1	2, 3, or 4
2	3 or 4
3	4
Cannot be 4	–

   For multiword operands, *i* must be 1 and 2, and *i'* must be 3 and 4.
3. The equal comparison indicator will be set if the sum is zero. If the sum is not zero, the ECI will be reset.
4. Indirect addressing, field selection, and multiword operands may be used.

BINARY SUBTRACT—BS

Operation: (ARi) $-$ (m') $\rightarrow$ ARi

Op Code: 25

Cycles: 2

Description: Algebraically subtract in binary code the operand in the indexed memory location(s) from the operand in the designated arithmetic register(s), placing the difference in the same arithmetic register(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing/field-selection option

X. . . . . Binary designation of index register

AR. . . Designation of arithmetic register(s)

m. . . . . Unindexed address of operand

NOTES

1. The equal comparison indicator (ECI) will be set if the difference is zero. If the difference is not zero, the ECI will be reset.
2. Indirect addressing, field selection, and multiword operands may be used.

BINARY SUBTRACT HIGHER—BSH

Operation: (ARi) $-$ (m') $\rightarrow$ ARi', where $i' > i$

Op Code: 27

Cycles: 2

Description: Algebraically subtract in binary code the operand in the indexed memory location(s) from the operand in the lower designated arithmetic register(s), placing the difference in the higher designated arithmetic register(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing/field-selection option

X. . . . . Binary designation of index register

AR. . . Designation of arithmetic registers. See note 2.

m. . . . . Unindexed address of operand

NOTES

1. The contents of ARi will not be disturbed.
2. For single-word operands, the designations of *i* and *i'* are as follows:

i	*i'*
1	2, 3, or 4
2	3 or 4
3	4
Cannot be 4	–

   For multiword operands, *i* must be 1 and 2, and *i'* must be 3 and 4.
3. The equal comparison indicator (ECI) will be set if the difference is zero. If the difference is not zero, the ECI will be reset.
4. Indirect addressing, field selection, and multiword operands may be used.

■ COMPARISON INSTRUCTIONS

Comparison instructions, which are used in conjunction with logical branching instructions, compare an operand in the arithmetic registers with an operand in memory. As a result of the comparison, one of the three comparison indicators, high, low, or equal, is set, and the other two are reset. The condition of the indicators may then be determined by a logical branching instruction.

All comparison instructions may use field selection and multiword operands.

COMPARE ALGEBRAIC—C

Operation: (ARi) : (m′)

Op Code: 54

Cycles: 2

Description: Compare algebraically the operand in the designated arithmetic register(s) with the operand in the indexed memory location(s). Set the appropriate comparison indicator as follows:

Condition	*Comparison Indicator Set*
(ARi) > (m′)	High
(ARi) < (m′)	Low
(ARi) = (m′)	Equal

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing/field-selection option

X. . . . . Binary designation of index register

AR. . . Designation of arithmetic register(s)

m. . . . . Unindexed address of operand

NOTES

1. Comparison is based on the binary value of the operands regardless of word format. When an operand is in alphanumeric format, the two most significant bits of each six-bit character represent the zone. Thus, blank or space, 00 0000, is the lowest character in order of magnitude and 11 1111 is the highest. Refer to table 2-1, *UNIVAC III Character Code.*
2. Only the sign of the least significant word of a multiword operand is considered. The signs of the more significant words are ignored.
3. When field selection is used, only the selected bit positions in memory and the corresponding bit positions of the arithmetic registers are compared.The sign of the operand from memory is considered to be positive.
4. The operands are not altered.
5. Plus zero will compare greater than minus zero.
6. Indirect addressing, multiword operands, and field selection may be employed.

COMPARE ABSOLUTE—CA

Operation: | (ARi) | : | (m′) |

Op Code: 55

Cycles: 2

Description: Compare the absolute value of the operand in the designated arithmetic register(s) with the absolute value of the operand in the indexed memory location(s). Set the appropriate comparison indicator as follows:

Condition	*Comparison Indicator Set*
| (ARi) | > | (m′) |	High
| (ARi) | < | (m′) |	Low
| (ARi) | = | (m′) |	Equal

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing/field-selection option

X. . . . . Binary designation of index register

AR. . . Designation of arithmetic register(s)

m. . . . . Unindexed address of operand

NOTES

1. All sign bits are ignored.
2. Comparison is based on the binary value of the operands, regardless of word format. When an operand is in alphanumeric format, the two most significant bits of each six-bit character represent the zone. Thus, blank or space, 00 0000, is the lowest character in order of magnitude and 11 1111 is the highest. Refer to table 2-1, *UNIVAC III Character Code.*
3. When field selection is used, only the selected bit positions in memory and the corresponding bit positions of the arithmetic registers are compared.
4. The operands are not altered.
5. Indirect addressing, multiword operands, and field selection may be used.

COMPARE ONE-BITS—CONE

Operation: 1-bits (ARi) : 1-bits (m′)

Op Code: 57

Cycles: 2

Description: Compare the 1-bits of the arithmetic register(s) designated with the 1-bits of an operand in memory. If the operand in memory contains a 1-bit in every position in which the operand in the arithmetic register(s) contains a 1-bit, set the equal comparison indicator; otherwise, set the high comparison indicator.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing/field-selection option
X . . . . . Binary designation of index register
AR . . . Designation of arithmetic register(s)
m . . . . . Unindexed address of operand

NOTES

1. Sign bits are included in the comparison unless field selection is specified.
2. When field selection is specified, only the selected bit positions in memory and the corresponding bit positions of the arithmetic register(s) are compared. All sign bits are ignored.
3. If the operand in the arithmetic register(s) is all 0's, the equal comparison indicator is set.
4. The operands are not altered.
5. Indirect addressing, multiword operands, and field selection may be used.

COMPARE ZERO-BITS—CZRO

Operation: 1-bits (ARi) : 0-bits (m′)
Op Code: 56
Cycles: 2

Description: Compare the 1-bits of the arithmetic register(s) designated with the 0-bits of an operand in memory. If the operand in memory contains a 0-bit in every position in which the operand in the arithmetic register(s) contains a 1-bit, set the equal comparison indicator; otherwise, set the high comparison indicator.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing/field-selection option
X . . . . . Binary designation of index register
AR . . . Designation of arithmetic register(s)
m . . . . . Unindexed address of operand

NOTES

1. Sign bits are included in the comparison unless field selection is specified.
2. When field selection is specified, only the selected bit positions in memory and the corresponding bit positions of the arithmetic register(s) are compared. All sign bits are ignored.
3. If the operand in the arithmetic register(s) is all 0's, the equal comparison indicator is set.
4. The operands are not altered.
5. Indirect addressing, multiword operands, and field selection, may be used.

■ LOGICAL BRANCHING INSTRUCTIONS

Six logical branching instructions are included in the UNIVAC III repertoire. Four of these instructions transfer control conditionally according to the setting of indicators by previous instructions; the remaining two transfer control unconditionally.

The instructions that transfer control conditionally test a comparison indicator or an arithmetic register sign indicator. If the indicator tested is reset, the next instruction in sequence is accessed. If the indicator is set, control is transferred to the memory location specified by the instruction. The condition of the tested indicator is not affected.

The two instructions that transfer control unconditionally are the TUN instruction and the TR instruction.

The TUN transfers control to the memory address specified by the instruction. The TR records the address of the next instruction in sequence (contents of the control counter) in the memory location specified by the instruction and transfers control to the following memory location. The stored control-counter reading, as the indirect-address control word of a TUN or some other transfer-of-control instruction, subsequently may be used as a return line. The TR may also be used to record the contents of the memory-address counters associated with the input-output channels.

TRANSFER IF EQUAL—TEQ

Operation: Test ECI: if set, m′ → CC
if reset, (CC) + 1 → CC
Op Code: 60
Cycles: 1 if set, 2 if reset

Description: Test the equal comparison indicator (ECI). If the indicator is set, access the instruction contained in the indexed memory address; otherwise, access the next instruction in sequence.

I/A	X	Op Code	Indicator	m
25	24 21	20 15	14 11	10 1

I/A . . . . Indirect-addressing option
X . . . . Binary designation of index register
Indicator . . . 0110—Equal comparison indicator (ECI)
m . . . . Unindexed address of next instruction if ECI is set

NOTES

1. The condition of the ECI will not be changed by the test.
2. The ECI is affected by add and subtract instructions, as well as by comparison instructions. If the result is zero, the indicator will be set; if the result is not zero, the indicator will be reset.
3. Indirect addressing may be used.

TRANSFER IF HIGH—THI

Operation: Test HCI : if set, m′ → CC
if reset, (CC) + 1 → CC

Op Code: 60

Cycles: 1 if set, 2 if reset

Description: Test the high comparison indicator (HCI). If the indicator is set, access the instruction in the indexed memory location; otherwise, access the next instruction in sequence.

I/A	X	Op Code	Indicator	m
25	24 21	20 15	14 11	10 1

I/A........	Indirect-addressing option
X.........	Binary designation of index register
Indicator...	0111—High comparison indicator (HCI)
m.........	Unindexed address of next instruction if HCI is set

NOTES

1. The condition of the indicator will not be changed by the test.
2. Indirect addressing may be used.

TRANSFER IF LOW—TLO

Operation: Test LCI : if set, m′ → CC
if reset, (CC) + 1 → CC

Op Code: 60

Cycles: 1 if set, 2 if reset

Description: Test the low comparison indicator (LCI). If the indicator is set, access the instruction contained in the indexed memory location; otherwise, access the next instruction in sequence.

I/A	X	Op Code	Indicator	m
25	24 21	20 15	14 11	10 1

I/A........	Indirect-addressing option
X.........	Binary designation of index register
Indicator...	0101—Low comparison indicator (LCI)
m.........	Unindexed address of next instruction if LCI is set

NOTES

1. The condition of the indicator will not be changed by the test.
2. Indirect addressing may be used.

TRANSFER IF POSITIVE—TPOS

Operation: Test sign of AR : if +, m′ → CC
if −, (CC) + 1 → CC

Op Code: 60

Cycles: 1 if positive, 2 if negative

Description: Test the designated arithmetic-register sign indicator. If the indication is positive, access the instruction in the indexed memory location; if negative, access the next instruction in sequence.

I/A	X	Op Code	Indicator	m
25	24 21	20 15	14 11	10 1

I/A........	Indirect-addressing option
X.........	Binary designation of index register
Indicator...	Arithmetic-register sign indicator. See note 2.
m.........	Unindexed address of next instruction if the sign of the arithmetic register is positive.

NOTES

1. The condition of the indicator will not be changed by the test.
2. The sign indicators are designated as follows:

Sign of	*Bits 14—11*	*Mnemonic Code*
AR1	0001	1
AR2	0010	2
AR3	0011	3
AR4	0100	4

3. Indirect addressing may be used.

TRANSFER CONTROL UNCONDITIONALLY—TUN

Operation: m′ → CC

Op Code: 06

Cycles: 1

Description: Replace the contents of the control counter (CC) with the indexed memory address.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A...	Indirect-addressing option
X.....	Binary designation of index register
AR...	Not relevant
m.....	Unindexed address of next instruction

NOTE

Indirect addressing may be used.

TRANSFER RETURN—TR

Operation: (CC/MAC) + 1 → m′
m′ + 1 → CC

Op Code: 07

Cycles: 3

Description: Transfer the contents of the control counter (CC) or designated memory-address counter (MAC), incremented by 1, into bit positions 1 through 15 of the indexed memory location and replace the contents of the control counter with the indexed memory address incremented by 1.

I/A	X	Op Code	CC/MAC	m
25	24 21	20 15	14 11	10 1

I/A.	Indirect-addressing option
X.	Binary designation of index register
CC/MAC. . .	Designation of control counter or memory-address counter. See note 2.
m.	Unindexed address of location in which CC/MAC reading is to be stored, and unindexed address, minus 1, of next instruction.

NOTES

1. Bit positions 16 through 25 of the indexed memory location will be cleared to binary 0's.
2. The control-counter and memory-address-counter designations are as follows:

Storage-Address Register	*Designation: Bits 14—11*	*Mnemonic Code*
Control Counter	0001	14
Memory-Address Counter		
UNISERVO IIIA, Basic Write	0011	1
UNISERVO IIIA, Basic Read	0100	2
General Purpose 1	0101	3
General Purpose 2	0110	4
General Purpose 3	0111	5
General Purpose 4	1000	6
General Purpose 5	1001	7
General Purpose 6	1010	8
General Purpose 7	1011	9
General Purpose 8	1100	10
UNISERVO IIA, Read-Write	1101	11
UNISERVO IIIA, Additional Write	1110	12
UNISERVO IIIA, Additional Read	1111	13

3. The contents of the memory-address register, incremented by 1, also may be transferred to memory by this instruction. Bit positions 14 through 11 of the instruction should be 0010 (mnemonic code, 15).
4. Indirect addressing may be used.

SENSE-INDICATOR INSTRUCTIONS

The sense-indicator instructions set, reset, and test the condition of eight program-control indicators in the Central Processor. If the indicator tested is reset, the next instruction in sequence is accessed. If the indicator is set, control is transferred to the memory location specified by the instruction.

SET SENSE INDICATOR—SSI

Op Code: 62

Cycles: 2

Description: Set the sense indicator specified by bits 11 through 14 of the instruction word.

I/A	X	Op Code	Indicator	m
25	24 21	20 15	14 11	10 1

I/A.	0
X.	Not relevant
Indicator. . .	Designation of indicator to be set. See note 1
m.	Not relevant

NOTES

1. The sense indicators are designated as follows:

Sense Indicator	*Designation: Bits 14—11*	*Mnemonic Code*
1	1000	1
2	1001	2
3	1010	3
4	1011	4
5	1100	5
6	1101	6
7	1110	7
8	1111	8

2. Attempting to set an indicator that is already set will not result in an error, and the condition of the indicator will remain unchanged.
3. Indirect addressing and field selection are not applicable.

RESET SENSE INDICATOR—RSI

Op Code: 61

Cycles: 2

Description: Reset the sense indicator specified by bits 11 through 14 of the instruction word.

I/A	X	Op Code	Indicator	m
25	24 21	20 15	14 11	10 1

I/A........0

X.........Not relevant

Indicator...Designation of indicator to be reset. See note 1, SSI for sense-indicator designations.

m.........Not relevant

NOTES

1. Attempting to reset an indicator that is already reset will not result in an error, and the condition of the indicator will remain unchanged.
2. Indirect addressing and field selection are not applicable.

TRANSFER IF SENSE INDICATOR SET—TSI

Operation: Test sense indicator: if set, m′ → CC
if reset, (CC) + 1 → CC

Op Code: 60

Cycles: 1 if set, 2 if reset

Description: Test the sense indicator specified by bits 11 through 14 of the instruction word. If the indicator is set, access the instruction in the indexed memory address; otherwise, access the next instruction in sequence.

I/A	X	Op Code	Indicator	m
25	24 21	20 15	14 11	10 1

I/A........Indirect-addressing option

X.........Binary designation of index register

Indicator...Designation of indicator to be tested. See note 1, SSI for sense-indicator designations.

m.........Unindexed address of next instruction if indicator is set

NOTES

1. The condition of the indicator will not be changed by the test.
2. Indirect addressing may be used.

■ SHIFT INSTRUCTIONS

Data in the arithmetic registers may be altered by means of the shift instructions for each of the three data formats: alphanumeric, decimal, and binary. The amount of shift is specified by the shift-count field, which is indexed and may be indirectly addressed. If index-register modification of the shift count is not desired, the index-register designation should be 0000.

A maximum of a two-word operand, in adjacent or nonadjacent arithmetic registers, may be shifted. The result always appears in the registers designated, and the contents of the other registers are unchanged.

SHIFT RIGHT (DECIMAL)—SR

Op Code: 40

Cycles: 4, one-word operand; 7, two-word operand*

Description: Shift right the contents of the designated arithmetic register(s) the number of decimal digit positions specified by the indexed shift count.

I/A	X	Op Code	AR	Shift Count
25	24 21	20 15	14 11	10 1

I/A..........Indirect-addressing option

X............Binary designation of index register

AR..........Designation of arithmetic register(s)

Shift Count...Unindexed number of decimal digit positions to be shifted, expressed in binary

NOTES

1. Sign bits are not shifted.
2. Digits shifted past the least significant digit position of the operand are lost, and decimal 0's (0011) are inserted from the left.
3. A maximum of a two-word operand, in adjacent or nonadjacent arithmetic registers, may be shifted. The results in either case will appear in the registers designated; the contents of the other registers are unchanged.
4. A shift count greater than the size of the operand will cause an erroneous shift, which would remove the contents of the designated arithmetic registers. A shift count of zero is allowable and will leave the contents of the arithmetic registers unchanged except for the conversion of non-numeric digit codes.
5. For a description of the treatment of non-numeric digit codes, refer to Appendix D.
6. Indirect addressing, but not field selection, may be used.

SHIFT LEFT (DECIMAL)—SL

Op Code: 41

Cycles: 3, one-word operand; 6, two-word operand†

Description: Shift left the contents of the arithmetic register(s) designated the number of decimal digit positions specified by the indexed shift count.

*Shifting a two-word operand from 7 to 12 decimal digit positions requires 4 cycles.

†Shifting a two-word operand from 7 to 12 decimal digit positions requires 3 cycles.

I/A	X	Op Code	AR	Shift Count
25	24 21	20 15	14 11	10 1

I/A..........	Indirect-addressing option
X............	Binary designation of index register
AR..........	Designation of arithmetic register(s)
Shift Count...	Unindexed number of decimal digit positions to be shifted, expressed in binary

NOTES

1. Sign bits are not shifted.
2. Digits shifted past the most significant digit position of the operand are lost and decimal 0's (0011) are inserted from the right.
3. A maximum of a two-word operand, in adjacent or nonadjacent arithmetic registers, may be shifted. The results in either case will appear in the registers designated; the contents of the other registers are unchanged.
4. A shift count greater than the size of the operand will cause an erroneous shift, which would remove the contents of the designated arithmetic registers. A shift count of zero is allowable and will not change the contents of the arithmetic registers except for the conversion of non-numeric digit codes.
5. For a description of the treatment of non-numeric digit codes, refer to Appendix D.
6. Indirect addressing, but not field selection, may be used.

SHIFT ALPHANUMERIC RIGHT—SAR

Op Code: 42

Cycles: 4, one-word operand; 9, two-word operand*

Description: Shift right the contents of the designated arithmetic register(s) the number of alphanumeric character positions specified by the indexed shift count.

I/A	X	Op Code	AR	Shift Count
25	24 21	20 15	14 11	10 1

I/A..........	Indirect-addressing option
X...........	Binary address of index register
AR..........	Designation of arithmetic register(s)
Shift Count...	Unindexed number of alphanumeric character positions to be shifted, expressed in binary

*Shifting a two-word operand from 5 to 8 alphanumeric character positions requires 4 cycles.

NOTES

1. Sign bits are not shifted.
2. Characters shifted past the least significant character position of the operand are lost, and spaces (00 0000) are inserted from the left.
3. A maximum of a two-word operand, in adjacent or nonadjacent arithmetic registers, may be shifted. The results in either case will appear in the registers designated; the contents of the other registers are unchanged.
4. A shift count greater than the size of the operand will cause an erroneous shift, which would remove the contents of the designated arithmetic registers. A shift count of zero is allowable and will not change the contents of the arithmetic registers.
5. Indirect addressing, but not field selection, may be used.

SHIFT ALPHANUMERIC LEFT—SAL

Op Code: 43

Cycles: 3, one-word operand; 8, two-word operand*

Description: Shift left the contents of the designated arithmetic register(s) the number of alphanumeric character positions specified by the indexed shift count.

I/A	X	Op Code	AR	Shift Count
25	24 21	20 15	14 11	10 1

I/A..........	Indirect-addressing option
X............	Binary designation of index register
AR..........	Designation of arithmetic register(s)
Shift Count...	Unindexed number of alphanumeric character positions to be shifted expressed in binary

NOTES

1. Sign bits are not shifted.
2. Characters shifted past the most significant character position of the operand are lost, and spaces (00 0000) are inserted from the right.
3. A maximum of a two-word operand, in adjacent or nonadjacent arithmetic registers, may be shifted. The results in either case will appear in the registers designated; the contents of the other registers are unchanged.
4. A shift count greater than the size of the operand will cause an erroneous shift, which would remove the contents of the designated arithmetic registers. A shift count of zero is allowable and will not change the contents of the arithmetic registers.
5. Indirect addressing, but not field selection, may be used.

*Shifting a two-word operand from 5 to 8 alphanumeric character positions requires 3 cycles.

SHIFT BINARY CIRCULAR (RIGHT)—SBC

Op Code: 44

Cycles: 4, shift count 0 through 7; 5, shift count 8 through 16; 6, shift count 17 through 25

Description: Shift right circularly the contents of the arithmetic register designated the number of bit positions specified by the indexed shift count.

I/A	X	Op Code	AR	Shift Count
25	24 21	20 15	14 11	10 1

I/A.......... Indirect-addressing option
X............ Binary designation of index register
AR.......... Designation of arithmetic register
Shift Count... Unindexed number of binary bit positions to be shifted, expressed in binary

NOTES

1. Bits shifted beyond the least significant bit position of the designated arithmetic register re-enter at the most significant bit position of the same register.
2. The sign bit is shifted.
3. The size of the operand is one word.
4. A shift count of zero is allowable and will not change the contents of the arithmetic registers.
5. A shift count greater than 25 will not cause an erroneous shift. Bit positions 6 through 15 of the indexed shift count will be ignored and a normal circular shift to the right will be performed. Therefore, the number of bit positions the operand will be effectively shifted is the value specified in bit positions 1 through 5 of the indexed shift count, minus 25. For example, if the indexed shift count is 127, and bit positions 6 through 15 are ignored, a value of 31 is obtained. When 25 is subtracted, the result, 6, is the number of bit positions the operand will be effectively shifted.
6. Indirect addressing, but not field selection, may be used.

■ CONVERSION INSTRUCTIONS

Conversion instructions may be used to convert numeric data from decimal to alphanumeric format or from alphanumeric to decimal format, and to replace leading zeros, commas, and other characters with nonprinting space codes.

CONVERT ALPHANUMERIC TO DECIMAL—ATD

Operation: (m′−2, m′−1, m′) → ARi, ARj

Op Code: 72

Cycles: 8

Description: Transfer the contents of three consecutive memory locations to two arithmetic registers; convert the operand from alphanumeric format (6-bit code) to decimal format (4-bit code) by removing the zone bits.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A... Indirect-addressing option
X..... Binary designation of index register
AR... Designation of two arithmetic registers
m..... Unindexed address of least significant word of operand

NOTES

1. The arithmetic registers designated need not be adjacent.
2. It is assumed that the operand in memory is a numeric field represented in 6-bit code. There is no check for non-numeric characters.
3. The signs of the two-word result in the arithmetic registers will be that of the least significant word of the operand in memory.
4. The operand in memory will not be altered.
5. Indirect addressing, but not field selection, may be used.

CONVERT DECIMAL TO ALPHANUMERIC—DTA

Operation: (ARi, ARj) → m′−2, m′−1, m′

Op Code: 71

Cycles: 8

Description: Transfer the contents of two arithmetic registers to three consecutive memory locations, converting the operand from decimal format (4-bit code) to alphanumeric format (6-bit code) by inserting zero zone bits.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A... Indirect-addressing option
X..... Binary designation of index register
AR... Designation of two arithmetic registers
m..... Unindexed address of least significant word of operand

NOTES

1. The arithmetic registers designated need not be adjacent.

2. It is assumed that the operand in the arithmetic registers is a numeric field represented in 4-bit code. No check is made for non-numeric digits.
3. The signs of the result in memory will be that of the least significant word of the operand in the arithmetic registers.
4. The operand in the arithmetic registers will not be altered.
5. Indirect addressing, but not field selection may be used.

ZERO SUPPRESS—ZUP

Op Code: 73

Cycles: 2

Description: Transfer the operand in the indexed memory location(s) to the arithmetic register(s) designated; replace the following leading alphanumeric characters with space codes (00 0000):

;	Semicolon	00 0001
—	Minus	00 0010
0	Zero	00 0011
,	Comma	11 0010

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing option

X. . . . . Binary designation of index register

AR. . . Designation of arithmetic register(s)

m. . . . . Unindexed address of most significant word of operand

NOTES

1. The replacement starts with the most significant character of the operand and stops when a character other than a space or one of the four given in the description is encountered.
2. Because the suppression proceeds from the most significant to the least significant character, when the operand is multiword the indexed memory location must be the address of its *most* significant word.
3. A multiword operand must be located in consecutive memory locations, but the suppressed result may be in nonadjacent arithmetic registers.
4. The operand in memory is not altered.
5. The original signs of the operand words are retained.
6. Indirect addressing, but not field selection may be used.

LOGICAL INSTRUCTIONS

Two instructions perform the logical AND and OR operations for each bit of an operand in the arithmetic registers and the corresponding bit of an operand in memory. The result of the operation appears in the arithmetic registers, leaving the operand in memory unaltered.

The operation table for each bit of the operand in the arithmetic registers has the following form:

	Initial	*Final*
(m′)	*(ARi)*	*(ARi)*
Bit position in memory	Bit position in ARi before execution	Bit position in ARi after execution

SUPERIMPOSE—SUP

Operation: 1-bits (m′) → ARi

Op Code: 15

Cycles: 2

Description: Transfer all the 1-bits of the operand in the indexed memory location(s) to the corresponding bit positions of the operand in the designated arithmetic register(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing/field-selection option

X. . . . . Binary designation of index register

AR. . . Designation of arithmetic register(s)

m. . . . . Unindexed address of operand

NOTES

1. A logical OR operation is performed on each bit position of the operands, including the signs, and the result is placed in the designated arithmetic register(s). The operation table for each bit position is as follows:

	Initial	*Final*
(m′)	*(ARi)*	*(ARi)*
0	0	0
0	1	1
1	0	1
1	1	1

2. The operand in memory is not altered.
3. When field selection is used bit positions of the operand in the arithmetic registers outside the portion selected, and sign bits are not affected.
4. Indirect addressing and multiword operands may be used.

ERASE—ERS

Operation: 0-bits (m′) → ARi

Op Code: 16

Cycles: 2

Description: Transfer all the 0-bits of the operand in the indexed memory location(s) to the corresponding bit positions of the operand in the designated arithmetic register(s).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing/field-selection option
X . . . . . Binary designation of index register
AR . . . Designation of arithmetic register(s)
m . . . . . Unindexed address of operand

NOTES

1. A logical AND operation is performed on each bit position of the operands, including the signs, and the result is placed in the designated arithmetic register(s). The following table shows the operation for each bit position:

(m')	*Initial* (ARi)	*Final* (ARi)
0	0	0
0	1	0
1	0	0
1	1	1

2. The operand in memory is not altered.
3. When field selection is used, bit positions of the operand in the arithmetic registers outside the portion selected, and sign bits are not affected.
4. Indirect addressing and multiword operands may be used.

■ INDEX-REGISTER INSTRUCTIONS

Four instructions are used to manipulate the nine or fifteen index registers. The contents of an index register may be incremented, decremented, incremented and compared, decremented and compared, stored, or replaced with a new value from memory. The index register to be manipulated is specified by its binary designation in bit positions 11 through 14 of the instruction word.

If index register 0000 is designated in these instructions, the result of the operation will, in general, be the same as that obtained by addressing an actual index register containing all binary 0's. Addressing an optional index register (10 to 15) not in the particular system in use will not cause an error. In general, the result will be the same as if a legitimate index register containing all binary 1's had been addressed.

LOAD INDEX REGISTER—LX

Operation: 15 LSB (m′) → XOi

Op Code: 51

Cycles: 3

Description: Transfer bits 1 through 15 of the indexed memory location to the index register specified by bit positions 11 through 14 of the instruction word.

I/A	X	Op Code	XO	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing option
X . . . . . Binary designation of index register modifying *m* address
XO . . . Binary designation of index register being loaded
m . . . . . Unindexed address of value being loaded into XO

NOTES

1. If XO is 0000 (or 1010 through 1111, when these index registers are not part of the system), no operation will be performed and the next instruction in sequence will be accessed.
2. Indirect addressing may be used; field selection and multiword operands are not applicable.

STORE INDEX REGISTER—STX

Operation: (XOi) → 15 LSB m′

Op Code: 50

Cycles: 3

Description: Transfer the contents of the index register specified by bit positions 11 through 14 of the instruction word to bit positions 1 through 15 of the indexed memory location.

I/A	X	Op Code	XO	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing option
X . . . . . Binary designation of index register modifying *m* address
XO . . . Binary designation of index register whose contents are being stored
m . . . . . Unindexed address of location in which (XO) are being stored

NOTES

1. Bit positions 16 through 25 of the indexed memory location are cleared to binary 0's.

2. If XO is 0000, bit positions 1 through 25 of the indexed memory location will contain binary 0's. If XO is 1010 through 1111, and these index registers are not part of the system, bit positions 1 through 16 will contain binary 1's, and bit positions 17 through 25 will contain binary 0's.
3. Indirect addressing may be used; field selection and multiword operands are not applicable.

INCREMENT INDEX REGISTER—IX

Operation: (XOi) + 9 LSB (m') → XOi

Op Code: 52

Cycles: 3

Description: Algebraically add, in binary, bit positions 1 through 9 of the indexed memory location to the contents of the index register designated by bit positions 11 through 14 of the instruction word, placing the sum in the same index register.

I/A	X	Op Code	XO	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing option

X . . . . . Binary designation of index register modifying *m* address

XO . . . Binary designation of index register being incremented

m . . . . . Unindexed address of increment

NOTES

1. When the sign of the indexed memory location is negative, the contents of the index register will be decremented. If the absolute value of a negative increment is greater than the absolute value of the index register contents, the result placed in the index register will be the binary complement of their difference.
2. Any carry beyond the most significant bit position of the index register will be ignored.
3. If bit positions 11 through 14 of the instruction word are 0000, or 1010 through 1111 when these index registers are not part of the system, no operation is performed.
4. Indirect addressing may be used; field selection and multiword operands are not applicable.

INCREMENT AND COMPARE INDEX REGISTER—ICX

Operation: (XOi) + 9 LSB (m') → XOi
|(XOi)| : |(m')| bits 10 through 24

Op Code: 53

Cycles: 4

Description: Algebraically add in binary the increment amount (bit positions 1 through 9) of the index-register-modification control word (XMOD) in the indexed memory location to the contents of the index register designated by bit positions 11 thorough 14 of the instruction word; place the result in the same index register. Compare the new contents of the index register with the absolute value of the comparison amount of the XMOD (bit positions 10 through 24) and set the appropriate comparison indicator as follows:

Condition	*Comparison Indicator Set*
|(XOi)| > |(m')|	High
|(XOi)| < |(m')|	Low
|(XOi)| = |(m')|	Equal

I/A	X	Op Code	XO	m
25	24 21	20 15	14 11	10 1

I/A . . . Indirect-addressing option

X . . . . . Binary designation of index register modifying *m* address

XO . . . Binary designation of index register being incremented and compared

m . . . . . Unindexed address of XMOD

NOTES

1. The index-register-modification control word (XMOD) has the following format:

±	Comparison Amount	Increment Amount
25	24 10	9 1

2. When the sign of the XMOD is negative, the contents of the index register will be decremented. If the absolute value of a negative increment is greater than the absolute value of the index-register contents, the result in the index register will be the binary complement of their difference.
3. Any carry beyond the most significant bit position of the index register will be ignored.
4. If bit positions 11 through 14 of the instruction word are 0000, the index register is not incremented. For purposes of comparison, the contents of the index register are considered to be binary 0's. If bit positions 11 through 14 of the instruction word are 1010 through 1111 when these index registers are not in the system, incrementing does not take place. For purposes of comparison, the contents of the index register are considered to be binary 1's.
5. The XMOD may be indirectly addressed; field selection and multiword operands are not applicable.

■ INITIATE-INPUT-OUTPUT-FUNCTION INSTRUCTION

All input-output operations, except those of the console typewriter, are initiated by the initiate-input-output-function instruction. An information word (function specification), containing the detailed information necessary to execute an input-output operation, is placed in a fixed memory location associated with the input-output channel and an indicator is set, alerting the synchronizer that an operation is to be performed. Once the operation has been initiated in this manner, subsequent control and monitoring of the input-output device is relegated to the synchronizer, and the Central Processor can continue with the execution of further instructions.

Function specifications are summarized in table 2-3. A detailed description of each function specification is contained in the section dealing with the pertinent input-output device.

INITIATE INPUT-OUTPUT FUNCTION—IOF

Operation: (m′) → channel standby location; set channel-standby-location interlock indicator

Op Code: 70

Cycles: 3

Description: Transfer the function specification from the indexed memory location to the fixed standby location associated with the channel designated by bit positions 11 through 14 of the instruction word, and set the channel standby-location interlock indicator.

I/A	X	Op Code	Channel	m
25	24 21	20 15	14 11	10 1

I/A.......	Indirect-addressing option
X.........	Binary designation of index register
Channel...	Channel designation. See note 2.
m.........	Unindexed address of function specification

NOTES

1. All input-output operations, except those pertaining to the console typewriter, are executed by means of this instruction and associated function specifications. A function specification is a control word which contains the detailed information necessary for the execution of an input-output operation. The IOF places the function specification in the standby memory location associated with an input-output channel. To alert the synchronizer that a function specification is available for execution, the IOF sets the standby-location interlock indicator associated with the channel.

   When the channel is free to perform an input-output operation, the function specification is accessed by the synchronizer for that channel, decoded, and executed. During magnetic tape or printer operations, the interlock indicator will be reset when the operation is successfully initiated and its execution has begun. For card reader or punch operations, the interlock indicator is unconditionally reset when the function specification is received by the synchronizer.

   If the standby-location interlock indicator of a channel is set, and an IOF instruction is executed for that channel, the associated function specification will replace the one already in the standby location. In normal use, the indicator should be tested (TIO instruction) and found reset before an IOF instruction is executed.

2. The channel designations are as follows:

Channel	*Designation: Bits 14—11*	*Mnemonic Code*
UNISERVO IIIA, Basic Write	0011	1
UNISERVO IIIA, Basic Read	0100	2
General Purpose 1	0101	3
General Purpose 2	0110	4
General Purpose 3	0111	5
General Purpose 4	1000	6
General Purpose 5	1001	7
General Purpose 6	1010	8
General Purpose 7	1011	9
General Purpose 8	1100	10
UNISERVO IIA, Read-Write	1101	11
UNISERVO IIIA, Additional Write	1110	12
UNISERVO IIIA, Additional Read	1111	13

3. The binary value of the channel designation is the address of the standby location associated with the channel.

4. Indirect addressing, but not field selection, may be used.

■ INPUT-OUTPUT INTERRUPT INSTRUCTIONS

Input-output interrupt instructions are used to determine the input-output unit originating an interrupt and the cause of the interrupt by testing the indicators associated with each input-output channel.

When the cause of the interrupt has been determined and corrective action, if required, has been taken, the indicators may be reset and the interrupted program resumed.

These instructions also provide the facility for setting, resetting, and testing the inhibit-input-output interrupt indicator.

Table 2-3. Summary of Function Specifications

Mnemonic Code	Function-Specification Operation	Function Code							
		Bit Position							
		24	23	22	21	20	19	18	17
	UNISERVO IIIA Tape Units								
GWT	Gather-write Tn					0	0	0	1
OWT	Overlay patterns ⟶ Tn, gather-write Tn					0	1	0	1
FSR	Forward scatter-read Tn					0	0	0	0
BSR	Backward scatter-read Tn					0	0	1	0
FBR	Forward-block-read, Tn ⟶ L...(L + W) — 1					0	1	0	0
BBR	Backward-block-read, Tn ⟶ L...(L — W) + 1	Tape-unit number (0 through 15)				0	1	1	0
FCSR	Forward contingency scatter-read Tn					1	0	0	0
BCSR	Backward-contingency scatter-read Tn					1	0	1	0
FCBR	Forward-contingency block-read, Tn ⟶ L...(L + W) — 1					1	1	0	0
BCBR	Backward-contingency block-read, Tn ⟶ L...(L — W) + 1					1	1	1	0
RWI	Rewind Tn with interlock (L is ignored)					0	1	1	1
RW	Rewind Tn without interlock (L is ignored)					0	0	1	1
LPT	Test Tn for load point					1	1	1	1
	W = number of words in a block								
	UNISERVO IIA Tape Units								
CWRT	Compatible write, L...L + 179 ⟶ Tn					0	0	0	1
CWSD	Compatible write subdivided, L...L + 179 ⟶ Tn					0	1	0	1
CFRH	Compatible forward-read high gain, Tn ⟶ L...L + 179					1	0	0	0
CFRL	Compatible forward-read low gain, Tn ⟶ L...L + 179	Tape-unit number (0 through 5)				0	1	0	0
CFRN	Compatible forward-read normal gain, Tn ⟶ L...L + 179					0	0	0	0
CBRH	Compatible backward-read high gain, Tn ⟶ L...L + 179					1	0	1	0
CBRL	Compatible backward-read low gain, Tn ⟶ L...L — 179					0	1	1	0
CBRN	Compatible backward-read normal gain, Tn ⟶ L...L — 179					0	0	1	0
CRWI	Compatible rewind with interlock					1	0	1	1
CRW	Compatible rewind					0	0	1	1
	High-Speed Reader*								
FCT	Feed card, select stacker 0, image with translation ⟶ L...L + 19					0	0	1	1
FCTS1	Feed card, select stacker 1, image with translation ⟶ L...L + 19					0	1	1	1
FCTS2	Feed card, select stacker 2, image with translation ⟶ L...L + 19					1	0	1	1
FC	Feed card, select stacker 0, image without translation ⟶ L...L + 39					0	0	0	1
FCS1	Feed card, select stacker 1, image without translation ⟶ L...L + 39					0	1	0	1
FCS2	Feed card, select stacker 2, image without translation ⟶ L...L + 39	All 0's				1	0	0	1
CT	Select stacker 0, image with translation ⟶ L...L + 19					0	0	1	0
CTS1	Select stacker 1, image with translation ⟶ L...L + 19					0	1	1	0
CTS2	Select stacker 2, image with translation ⟶ L...L + 19					1	0	1	0
CAD	Select stacker 0, image without translation ⟶ L...L + 39					0	0	0	0
CS1	Select stacker 1, image without translation ⟶ L...L + 39					0	1	0	0
CS2	Select stacker 2, image without translation ⟶ L...L + 39					1	0	0	0
	Card-Punch Unit*								
PC	Feed card, select stacker 0, image without translation L...L + 39 ⟶ punch					0	0	0	1
PCT	Feed card, select stacker 0, image with translation L...L + 19 ⟶ punch	All 0's				0	0	1	1
PCS	Feed card, select stacker 1, image without translation L...L + 39 ⟶ punch					0	1	0	1
PCTS	Feed card, select stacker 1, image with translation L...L + 19 ⟶ punch					0	1	1	1
CCS	Select stacker 1					0	1	0	0
	High-Speed Printer								
PRT	Advance paper and print L...L + 31	Number of lines						1	0
PAD	Advance paper (L is ignored)							0	0

* Transfer addresses given here are for 80-column card only. Both translated and untranslated 90-column card images are transferred to and from L...L + 23.

TEST INPUT-OUTPUT INDICATORS—TIO

Operation: Test I-O indicators: if set, (CC) + 1 → CC; if reset, (CC) + 2 → CC

Op Code: 64

Cycles: 2

Description: Test the input-output indicators specified by bit positions 1 through 10 for the channel designated by bit positions 11 through 14. If one or more of the indicators tested is set, access the next instruction in sequence; otherwise, skip the next instruction in sequence.

I/A	X	Op Code	Channel	Indicators
25	24 21	20 15	14 11	10 1

I/A........Indirect-addressing option

X..........Binary designation of index register

Channel.....Input-output channel designation. See note 6.

Indicators...Input-output indicator designations. See note 7.

NOTES

1. Any number of indicators may be tested simultaneously. If one or more of the indicators tested is set, the next instruction in sequence will be accessed.
2. The condition of the indicators is unchanged by the test.
3. Testing an indicator which is undefined for the channel designated will not cause an error. The result of the test will be the same as if a defined indicator in a reset condition had been tested.
4. The location immediately following this instruction should contain an instruction which will transfer control to the appropriate input-output routine for the channel designated.
5. Indirect addressing may be used.
6. The channel designations are as follows:

Channel	*Designation: Bits 14—11*	*Mnemonic Code*
UNISERVO IIIA, Basic Write	0011	1
UNISERVO IIIA, Basic Read	0100	2
General Purpose 1	0101	3
General Purpose 2	0110	4
General Purpose 3	0111	5
General Purpose 4	1000	6
General Purpose 5	1001	7
General Purpose 6	1010	8
General Purpose 7	1011	9
General Purpose 8	1100	10
UNISERVO IIA, Read-Write	1101	11
UNISERVO IIIA, Additional Write	1110	12
UNISERVO IIIA, Additional Read	1111	13

7. The indicators associated with each input-output device are designated by 1-bits in bit positions 1 through 10 of the instruction as shown in table 2-4.

Table 2-4. Input-Output Indicators

Condition	*1 in Bit Position(s) (Mnemonic Code)*
UNISERVO IIIA TAPE UNIT	
Standby-location interlock	1
Successful completion	2
Error A	3
Busy	4
Error B (tape-read error)	5
FS-call error	145
End-of-tape warning	6
Fault	7
Fault (occurring during attempted execution of instruction)	37
UNISERVO IIA TAPE UNIT	
Standby-location interlock	1
Successful completion	2
Data-transfer error	5
FS-call error	157
Greater-than-720 error	6
Fault	7
HIGH-SPEED READER CARD-PUNCH UNIT	
Standby-location interlock	1
Initiation	2
Data error	5
Fault	7
Operator oversight	6
HIGH-SPEED PRINTER	
Standby-location interlock	1
Successful completion	2
Data error	5
FS-call error	15
Out-of-paper warning	6
Fault	7

RESET INPUT-OUTPUT INDICATORS—RIO

Op Code: 65

Cycles: 2

Description: Reset the input-output indicators specified by bit positions 1 through 10 for the channel designated by bit positions 11 through 14. Reset the input-output interrupt-mode indicator.

I/A	X	Op Code	Channel	Indicators
25	24 21	20 15	14 11	10 1

I/A	Indirect-addressing option
X	Binary designation of index register
Channel	Input-output channel designation. See TIO, note 6 for channel designations.
Indicators	Input-output indicator designations. See TIO, note 7 for indicator designations.

NOTES

1. Any number of indicators may be reset simultaneously.
2. Attempting to reset an indicator which is undefined for the channel designated or which is already reset, will not result in an error.
3. Resetting any indicator that is set automatically resets the input-output interrupt-mode indicator, and all interrupts are inhibited until after the next instruction is executed.
4. Indirect addressing may be used.

PREVENT INPUT-OUTPUT INTERRUPT—PIO

Op Code: 62

Cycles: 2

Description: Set the inhibit-input-output interrupt indicator, preventing input-output interrupt until the indicator is reset.

I/A	X	Op Code	Indicator	m
25	24 21	20 15	14 11	10 1

I/A	0
X	Not relevant
Indicator	0000—Inhibit-input-output interrupt indicator
m	Not relevant

NOTES

1. The setting of specific input-output interrupt indicators is unchanged by this instruction.
2. Indirect addressing and field selection are not applicable.

ALLOW INPUT-OUTPUT INTERRUPT—AIO

Op Code: 61

Cycles: 2

Description: Reset the inhibit-input-output interrupt indicator.

I/A	X	Op Code	Indicator	m
25	24 21	20 15	14 11	10 1

I/A	0
X	Not relevant
Indicator	0000—Inhibit-input-output interrupt indicator
m	Not relevant

NOTES

1. All interrupts are inhibited until after the next instruction is executed.
2. Indirect addressing and field selection are not applicable.

TRANSFER IF INPUT-OUTPUT INTERRUPT PREVENTED—TIOP

Operation: Test inhibit-input-output interrupt indicator: if set, $m' \rightarrow CC$
if reset, $(CC) + 1 \rightarrow CC$

Op Code: 60

Cycles: 1 if set; 2 if reset

Description: Test the inhibit-input-output interrupt indicator. If the indicator is set, access the instruction in the indexed memory location; otherwise, access the next instruction in sequence.

I/A	X	Op Code	Indicator	m
25	24 21	20 15	14 11	10 1

I/A	Indirect-addressing option
X	Binary designation of index register
Indicator	0000—Inhibit-input-output interrupt indicator
m	Unindexed address of next instruction if input-output interrupt is inhibited

NOTES

1. The condition of the indicator is unchanged by the test.
2. Indirect addressing, but not field selection, may be used.

■ PROCESSOR-ERROR AND CONTINGENCY INTERRUPT INSTRUCTIONS

Processor-error and contingency interrupt instructions are used to determine the specific cause of interrupt by testing the appropriate indicators. When the condition which caused the interrupt has been acted upon, the indicators may be reset and the interrupted program resumed.

TEST CONTINGENCY INDICATORS—TCI

Operation: Test contingency indicators:
if set, $(CC) + 1 \rightarrow CC$
if reset, $(CC) + 2 \rightarrow CC$

Op Code: 64

Cycles: 2

Description: Test the contingency indicators specified by bit positions 1 through 10 of the instruction word. If any is set, access the next instruction in sequence; otherwise skip the next instruction in sequence.

I/A	X	Op Code	Class	Indicators
25	24 21	20 15	14 11	10 1

I/A	Indirect-addressing option
X	Binary designation of index register
Class	0010—Contingency
Indicators	Contingency-indicator designations. See note 3.

NOTES

1. Any number of indicators may be tested simultaneously. If any indicator tested is set, the next instruction in sequence will be accessed.
2. The condition of the indicators is unchanged by the test.
3. The contingency indicators are designated by 1-bits in the following bit positions:

Contingency	*Bit Position and Mnemonic Code**
Arithmetic overflow/clock power disrupted	1
Invalid op code	2
Typewriter interrupt	3
Keyboard request	4
Keyboard release	5
Contingency stop	6

*Bit positions 7 through 10 should be 0's.

4. The location immediately following the TCI should contain transfer of program control to a contingency routine which will process the condition specified by the indicators.
5. Testing an undefined indicator will not result in an error. The result of the test will be the same as if a defined indicator in a reset condition had been tested.
6. Indirect addressing, but not field selection, may be used.

RESET CONTINGENCY INDICATORS—RCI

Op Code: 65

Cycles: 2

Description: Reset the contingency indicators specified by bit positions 1 through 10 of the instruction word. Reset the contingency interrupt-mode indicator (CIMI).

I/A	X	Op Code	Class	Indicators
25	24 21	20 15	14 11	10 1

I/A	Indirect-addressing option
X	Binary designation of index register
Class	0010—Contingency
Indicators	Contingency-indicator designations. See TCI, note 3 for indicator designations.

NOTES

1. Any number of indicators may be reset simultaneously.
2. Attempting to reset an undefined indicator or an indicator already reset will not result in an error.
3. Resetting any indicator that is set automatically resets the contingency interrupt-mode indicator. All interrupts are then inhibited until after the next instruction is executed.
4. Indirect addressing, but not field selection, may be used.

TEST PROCESSOR-ERROR INDICATORS—TPE

Operation: Test processor-error indicators:
if set, $(CC) + 1 \rightarrow CC$
if reset, $(CC) + 2 \rightarrow CC$

Op Code: 64

Cycles: 2

Description: Test the processor-error indicators specified by bit positions 1 through 10 of the instruction word. If any is set, access the next instruction in sequence; otherwise, skip the next instruction in sequence.

I/A	X	Op Code	Class	Indicators
25	24 21	20 15	14 11	10 1

I/A	Indirect-addressing option
X	Binary designation of index register
Class	0001—Processor error
Indicators	Processor-error-indicator designations. See note 5.

NOTES

1. Any number of indicators may be tested simultaneously. If any indicator tested is set, the next instruction in sequence will be accessed.

2. The condition of the indicators will be unchanged by the test.
3. Indirect addressing, but not field selection, may be used.
4. The location immediately following the TPE should contain a transfer of program control to a subroutine which will process the condition specified by the indicators.
5. The indicators are designated by 1-bits in bit positions 1 through 10 as shown in table 2-5.

Table 2-5. Processor-Error Indicators

Condition	*1 in Bit Position(s) (Mnemonic Code)*
MEMORY-ADDRESS ERROR	
Transferring Operand to Memory	
Central Processor	2
Input-Output Channel	
UNISERVO IIIA, Basic Write	12
UNISERVO IIIA, Basic Read	3
General Purpose 1	13
General Purpose 2	23
General Purpose 3	123
General Purpose 4	4
General Purpose 5	14
General Purpose 6	24
General Purpose 7	124
General Purpose 8	34
UNISERVO IIA Synchronizer	134
UNISERVO IIIA, Additional Write	234
UNISERVO IIIA, Additional Read	1234
Reading operand from memory	9
Reading instruction from memory	1
MODULO-3 ERROR	
Reading operand from memory	5
Reading TUN, TR, TIOP, or WT instruction from memory	58
Transferring operand to or from memory	6
Adder output	7

RESET PROCESSOR-ERROR INDICATORS—RPE

Op Code: 65

Cycles: 2

Description: Reset the processor-error indicators specified by bit positions 1 through 10 of the instruction word. Reset the processor-error interrupt-mode indicator (PEIMI).

I/A	X	Op Code	Class	Indicators
25	24 21	20 15	14 11	10 1

I/A.........Indirect-addressing option

X..........Binary designation of index register

Class........0001—Processor error

Indicators...Processor-error-indicator designations. See TPE, note 5 for indicator designations.

NOTES

1. Any number of indicators may be reset simultaneously.
2. Attempting to reset an undefined indicator or an indicator already reset will not result in an error.
3. Resetting any indicator that was set automatically resets the PEIMI. All interrupts are then inhibited until after the next instruction is executed.
4. Indirect addressing, but not field selection, may be used.

■ CONSOLE-TYPEWRITER INSTRUCTIONS

When the console typewriter is on-line, its operation is controlled by three console-typewriter instructions. The manner in which they are used to transmit information between the operator and programs is described under the heading *Console Typewriter.*

ACTIVATE KEYBOARD—ACT

Op Code: 66

Cycles: 2

Description: Activate the typewriter keyboard, so that one alphanumeric character can be typed into the typewriter-buffer register (TBR).

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A...0

X.....0000

AR...0000

m.....Binary 0's

NOTES

1. The KEYBOARD ACTIVE light on the operator's console lights when this instruction is executed.
2. When a character is typed into the typewriter-buffer register, or when the KEYBOARD RELEASE button is pressed, the keyboard is deactivated and the KEYBOARD ACTIVE light extinguished. Typing in a character sets the console-typewriter-interrupt indicator; pressing the KEYBOARD RELEASE button sets the keyboard-release interrupt indicator. In either case, a contingency interrupt will occur.

3. Typing a character into the TBR does not cause a character to be printed or any typewriter action to be performed.
4. The Central Processor can execute other instructions while the character is being typed in.

READ TYPEWRITER CHARACTER—RT

Operation: (ARi) + (TBR) → ARi

Op Code: 01

Cycles: 2

Description: Add the alphanumeric character in the typewriter-buffer register (TBR) to the least significant character (bit positions 1 through 6) of the arithmetic register designated.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . .0

X. 0000

AR. . .Designation of arithmetic register

m. . . . .Binary 0's

NOTES

1. Because the character in the TBR is transferred to the designated arithmetic register by binary addition, bit positions 1 through 6 of the arithmetic register should be preset to binary 0's.
2. The operation is a normal binary addition except that no carries will pass from bit position 6 to bit position 7.
3. Bit positions 7 through 25 of the arithmetic register are unchanged.
4. Indirect addressing, field selection, and multiword operands are not applicable.

WRITE TYPEWRITER CHARACTER—WT

Operation: Typewriter on-line: 1 character (m')→TBR; (CC) + 2 → CC
Typewriter off-line: (CC) + 1 → CC

Op Code: 02

Cycles: 2

Description: If the typewriter is on-line, transfer the character specified by bit positions 11 through 14 of the instruction from the indexed memory location to the typewriter-buffer register (TBR); initiate a print cycle, and skip the next instruction in sequence. If the typewriter is off-line, execute the next instruction in sequence.

I/A	X	Op Code	Character	m
25	24 21	20 15	14 11	10 1

I/A. . . . . . . .Indirect-addressing option

X. . . . . .Binary designation of index register

Character. . .Character position to be printed. See note 1.

m. . . . . .Unindexed address of character to be printed

NOTES

1. The position of the character to be transferred from the indexed memory location to the TBR is designated in bit positions 11 and 12 of the instruction word as shown below; bit positions 13 and 14 are not significant:

Positions (m')		*Desig-nation*	
Character	*Bits*	*12*	*11*
4	24 through 19	1	1
3	18 through 13	1	0
2	12 through 7	0	1
1	6 through 1	0	0

2. When the character in the TBR has been printed, or the nonprinting function has been performed, the console-typewriter-interrupt indicator is set, causing a contingency interrupt.
3. If the typewriter is off-line, the instruction is not executed and the next instruction in sequence is accessed. The console-typewriter-interrupt indicator is not set.
4. Indirect addressing, but not field selection, may be used.

■ MISCELLANEOUS INSTRUCTIONS

Several instructions, classified as miscellaneous, are included under this heading; these are the no-operation, store tape control-word register, wait (and transfer), display, and load-time instructions.

NO OPERATION—NOP

Op Code: 00

Cycles: 2

Description: No operation is performed. Access the next instruction in sequence.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . .Not relevant

X. . . . .Not relevant

AR. . .Not relevant

m. . . . .Not relevant

NOTES

1. The contents of memory and the arithmetic registers, and the condition of indicators are unchanged.

STORE MEMORY-ADDRESS COUNTER—STMC

Operation: (MACi) → m′

Op Code: 04

Cycles: 3

Description: Transfer the contents of the memory-address counter (MAC) specified by bit positions 11 through 14 of the instruction word to bit positions 1 through 15 of the indexed memory location. The contents of the control counter may also be specified.

I/A	X	Op Code	MAC/CC	m
25	24 21	20 15	14 11	10 1

I/A	Indirect-addressing option
X	Binary designation of index register
MAC/CC . . .	Designation of memory-address counter or control counter. See notes 2 through 5.
m	Unindexed address of location in which MAC/CC reading is to be stored

NOTES

1. Bit positions 16 through 25 of the indexed memory location are cleared to binary 0's.
2. At the time of transfer the memory-address counter designated will contain the following:

Unit	*MAC Contents*
UNISERVO IIIA (using tape control words)	Address of tape control word currently in UNISERVO IIIA synchronizer channel
UNISERVO IIIA (not using tape control words)	Address to or from which previous data transfer took place
UNISERVO IIA General Purpose 1 through 8	Address to or from which previous data transfer took place

3. The contents of the control counter are specified by placing 0001 in bit positions 14 through 11 of the instruction word (mnemonic designation 14). The control counter at the time of transfer will contain the address of the STMC instruction.
4. The contents of the memory-address register (MAR) may also be transferred to memory by placing 0010 in bit positions 14 through 11 of the instruction word (mnemonic designation 15). The MAR at the time of transfer will contain the address of the location to which its contents are to be transferred.
5. The memory-address-counter designations are as follows:

Channel	*Designation: Bits 14—11*	*Mnemonic Code*
UNISERVO IIIA, Basic Write	0011	1
UNISERVO IIIA, Basic Read	0100	2
General Purpose 1	0101	3
General Purpose 2	0110	4
General Purpose 3	0111	5
General Purpose 4	1000	6
General Purpose 5	1001	7
General Purpose 6	1010	8
General Purpose 7	1011	9
General Purpose 8	1100	10
UNISERVO IIA, Read-Write	1101	11
UNISERVO IIIA, Additional Write	1110	12
UNISERVO IIIA, Additional Read	1111	13

6. Indirect addressing, but not field selection, may be used.

STORE TAPE CONTROL-WORD REGISTER—STCR

Operation: (TCWRi) → m′

Op Code: 05

Cycles: 3

Description: Transfer the contents of the tape control-word register (TCWR) for the UNISERVO IIIA synchronizer channel specified by bit positions 11 through 14 of the instruction word to the indexed memory location.

I/A	X	Op Code	TCWR	m
25	24 21	20 15	14 11	10 1

I/A	Indirect-addressing option
X	Binary designation of index register
TCWR . . .	Designation of tape control-word register. See note 2.
m	Unindexed address of location in which (TCWR) are to be stored

NOTES

1. The indexed memory location will contain the following information:

   Bits 1 through 15: Binary address of the last word transferred to or from the synchronizer channel.

   Bits 16 through 24: Original count as contained in the scatter-read/gather-write (SCAT) control word, decremented by 1 for each word transferred.

   Bit 25: Sign, which always is positive.

2. The UNISERVO IIIA synchronizer-channel designations for this instruction only are as follows:

Channel	*Designation: Bits 14—11*	*Mnemonic Code*
Basic Write	1000	4
Basic Read	0100	3
Additional Write	0010	2
Additional Read	0001	1

3. Indirect addressing, but not field selection, may be used.

WAIT (AND TRANSFER)—WAIT

Operation: m′ → CC; Stop Central Processor

Op Code: 77

Cycles: 2

Description: Replace the contents of the control counter with the indexed memory address specified; then stop the Central Processor.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing option

X. . . . . Binary designation of index register

AR. . . Not relevant

m. . . . . Unindexed address at which program is to be resumed when the PROGRAM RUN button on the operator's console is pressed

NOTES

1. The Central Processor will stop accessing instructions after the indexed memory address has been transferred to the control counter. All tape and peripheral operations in progress will continue until they have been completed, and any function specifications in standby locations will be executed.
2. When the PROGRAM RUN button on the operator's console is pressed, the program is resumed at the address specified by the control counter. Interrupt will occur at this time if any interrupt indicators were set after the execution of the WAIT instruction.
3. Indirect addressing, but not field selection, may be used.

DISPLAY—DIS

Operation: (m′) → Display

Op Code: 03

Cycles: 2

Description: Transfer the 27 bits of the indexed memory location to the memory-information display on the engineer's maintenance panel.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . Indirect-addressing option

X. . . . . Binary designation of index register

AR. . . 0000

m. . . . . Unindexed address of operand to be displayed

NOTES

1. The DISPLAY SELECTOR switch on the engineer's mantenance panel must be set to position 0 to light the display indicators.
2. Indirect addressing, but not field selection or multiword operands, may be used.

LOAD TIME—LT

Operation: (Clock) → ARi

Op Code: 76

Cycles: 2

Description: Transfer the clock reading to the arithmetic register designated.

I/A	X	Op Code	AR	m
25	24 21	20 15	14 11	10 1

I/A. . . 0

X. 0000

AR. . . Designation of arithmetic register

m. . . . . Binary 0's

NOTES

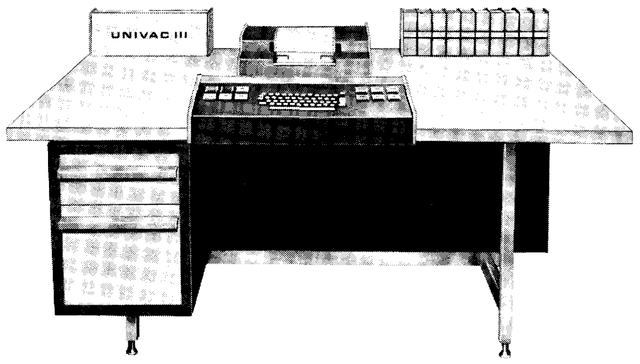
1. Every six seconds, one-half second is required to change contact settings within the clock. If the LT instruction is executed during this period, an invalid time is transferred to the designated arithmetic register and the next instruction in sequence is accessed: (CC) + 1 → CC.
2. If the contact settings are not changing when the LT instruction is executed, a valid time is transferred to the designated arithmetic register and the next instruction in sequence is skipped: (CC) + 2 → CC.
3. The valid time is expressed as five decimal digits in the following format:

25 21	20 17	16 13	12 9	8 5	4 1

Bit Positions	*Valid Time*
1 through 4	Tenth of minute (0 through 9)
5 through 12	Minute (00 through 59)
13 through 20	Hour (00 through 23)
21 through 25	— Filled with binary 0's

4. If more than one arithmetic register is designated, the clock reading will be transferred to the highest arithmetic register designated.
5. If the UNIVAC III system does not include an addressable clock and the instruction is executed, the arithmetic register will receive binary 0's and the next instruction in sequence will be accessed: (CC) $+ 1 \rightarrow$ CC.
6. If power to the clock has been temporarily removed, the execution of an LT instruction will set the arithmetic-overflow indicator and a contingency interrupt will result. To prevent further contingency interrupts of this nature the operator must reset the clock by pressing the reset button on the clock housing.
7. Indirect addressing, field selection, and multiword operands are not applicable.

OPERATOR'S CONSOLE



The UNIVAC III Operator's-Console control panel (figure 2-5) contains buttons and lights used to control the Central Processor and monitor the peripheral equipment, and a typewriter for program-operator communication. The functions of these buttons and lights are described in table 2-6.

CONSOLE TYPEWRITER

The console typewriter, functioning on-line under program control, is used to transmit and record all operator-program communications, which include program requests for operator intervention, operator replies to such requests, detection of machine malfunctions, program running times, and other logging information. The typewriter may be used off-line as a conventional electric typewriter.

The console typewriter has the following characteristics:

SPEED:
10 characters per second

CHARACTER SET:
51 (0 through 9, A through Z, and 15 special symbols)

FORMAT:
Program-controlled through use of three nonprinting characters: carriage return and line feed; horizontal tab; and form feed (advances paper to first printing line of next 5½- or 11-inch form).

SPACING:
Horizontal—10 characters per inch
Vertical— 6 lines per inch

FORM FEEDING:
Sprocket-fed continuous forms

PAPER WIDTH:
8½ inches, including sprocket holes

NUMBER OF COPIES:
An original and at least 5 carbon copies

Character codes for the UNIVAC III console typewriter are shown in table 2-7.

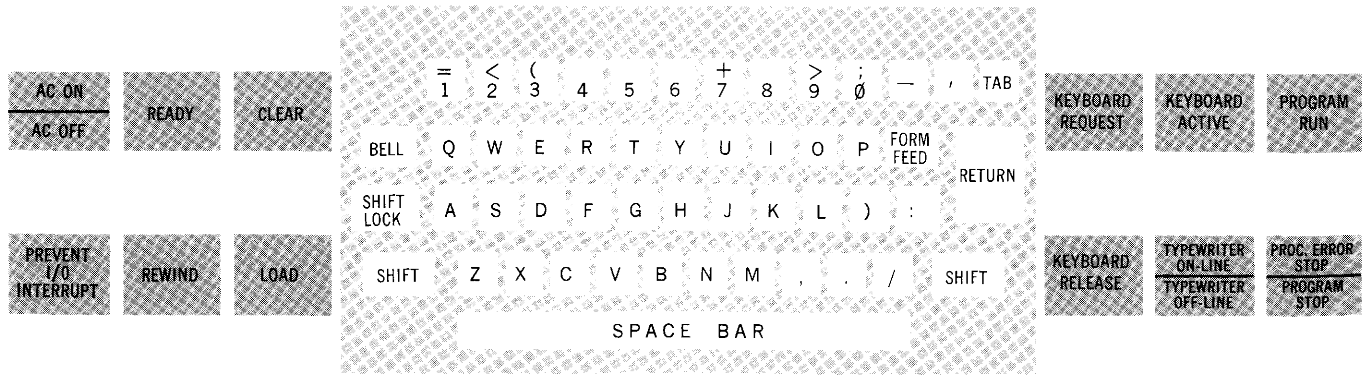


Figure 2-5. UNIVAC III Operator's-Console Control Panel

Table 2-6. Buttons and Lights on Operator's Console

BUTTONS AND BUTTON-LIGHTS

Button Marking	Function
AC ON	Turns on a-c power to system when power is off; upper half lights.
AC OFF	Turns off a-c power to system when power is on; lower half lights. This button is inoperative unless key-operated switch under console is turned on.
CLEAR*	Resets error, contingency, all input-output, sense, and interrupt-mode indicators. Clears control counter, index and arithmetic registers, and memory-address counters to binary 0's.
LOAD†	Causes UNISERVO IIIA tape unit designated 0 at plugboard to read forward one block without control words. Base address of data transferred to memory is specified by memory-address counter of basic read channel (normally set to binary 0's). Lights if Error A occurs during read operation‡. Turned off when CLEAR button is pressed.
REWIND†	Causes UNISERVO IIIA tape unit designated 0 at plugboard to rewind without interlock. Lights if Error B is detected during read operation initiated by pressing LOAD button‡. Turned off when CLEAR button is pressed.
PROGRAM RUN	Causes Central Processor to begin executing instructions, beginning with address in control counter. Remains lit while instructions are being executed.
PROC. ERROR STOP / **PROGRAM STOP**	Sets contingency-stop indicator, causing contingency interrupt. Upper half lights when Central Processor has been stopped automatically by processor errors after processor-error interrupt-mode indicator is set. Lower half lights when Central Processor is brought to programmed stop.
KEYBOARD REQUEST*	Sets keyboard-request indicator in Central Processor, causing contingency interrupt. Inactive when typewriter is off-line.
KEYBOARD RELEASE*	Sets keyboard-release indicator in Central Processor, causing contingency interrupt and turning off KEYBOARD ACTIVE light.
TYPEWRITER ON-LINE	Places typewriter on-line if it is off-line when pressed; upper half lights.
TYPEWRITER OFF-LINE	Places typewriter off-line if it is on-line when pressed; lower half lights.
Signal buzzer* (under console apron)	Used to signal persons at other units of the system; primarily a safety device used by customer engineers.

LIGHTS

Light Marking	Conditions Under Which Lighted
READY	Power on and Central Processor ready to operate. NOTE: Several minutes elapse before button lights after AC ON button is pressed.
PREVENT I/O INTERRUPT	Prevent-input-output-interrupt (PIO) instruction has been executed and inhibit-input-output-interrupt indicator is set.
Monitor panel (10 lights mounted on top of console)	Upper half: Abnormal condition in peripheral unit connected to general-purpose channels, in UNISERVO power supply, or in Central Processor. Signal buzzer, which sounds together with lighting of button, may be overridden by pressing buzzer-override button (on left side of monitor panel). Light goes out and buzzer stops sounding when abnormal condition is corrected. Lower half: Peripheral unit connected to a general-purpose channel is off-line.
KEYBOARD ACTIVE	Activate-keyboard (ACT) instruction has been executed. Turns off when a character is typed in, or when KEYBOARD RELEASE button is pressed.

* Button only.
† Ineffective unless PROGRAM STOP button is lit.
‡ Error A and Error B described in detail in Section III, under "Interrupt Indicators."

On-Line Operation

All typewriter input to and output from the typewriter is transferred character-by-character through the six-bit typewriter-buffer register (TBR) under program control. Information is typed out under program control in the following manner.

A write-typewriter-character (WT) instruction is executed, transferring one character from memory to the TBR and initiating a print cycle. Once the print cycle is initiated, the Central Processor is free to execute further instructions until the character has been printed or the function performed. When the print cycle is completed, the typewriter interrupt indicator is set, causing a contingency interrupt. The typewriter interrupt indicator is reset by the program and another WT instruction is executed; this instruction causes the next character of the information to be printed. This process is continued until all of the information has been printed out.

Information is typed in under program control in the following manner. The operator presses the KEYBOARD REQUEST button, thereby setting the keyboard-request indicator and causing a contingency interrupt. The program resets the keyboard-request indicator and executes an activate-keyboard (ACT) instruction, which activates the keyboard so that one character can be typed into the TBR. Once the ACT instruction has been executed, the Central Processor is free to continue with further instructions until a character is typed in.

When a character is typed into the TBR, the keyboard is deactivated and the typewriter interrupt indicator is set, causing a contingency interrupt. The program resets the typewriter interrupt indicator and executes a read-typewriter (RT) instruction, transferring the character in the TBR to the designated arithmetic register. The program may then transfer the character to memory and perform any desired operations upon it.

When the typewriter is on-line, pressing on a character key inserts the character into the TBR but does not cause it to be printed. So that the operator can verify the character just typed in, and to provide a printed record of the information, the program executes a WT instruction, transferring the character from memory to the TBR and initiating a print cycle. When the print cycle is completed, another activate-keyboard instruction is executed, and the next character may be typed in.

This process is repeated until the operator presses the KEYBOARD RELEASE button, setting the keyboard-release indicator and causing a contingency interrupt. When this indicator is set, the program is alerted that all of the information has been typed in.

Control of on-line typewriter operations is one of the functions of the contingency-control subroutines of the executive monitoring systems. These subroutines are used to record and control all typewriter communication between programs and the operator, and to record program running times, the occurrence of equipment faults, and other logging information by means of the typewriter.

Table 2-7. UNIVAC III Console-Typewriter Code

Numeric Bits	Zone Bits			
	00	01	10	11
0000	Space	+	5	$
0001	;	)	*	(
0010	–	.	$	Comma ,
0011	Ø	Carriage return and line feed	Ring bell	' Apostrophe
0100	1	A	J	/
0101	2	B	K	S
0110	3	C	L	T
0111	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
1101	:	=	2	:
1110	<	–	Horizontal tab	Form feed
1111	>	Ø	4	U

SECTION III

TAPE SYSTEM

The primary tape system of the UNIVAC III is composed of either one or two UNISERVO IIIA synchronizers each with up to 16 UNISERVO IIIA tape units. Data and control information are transferred between the Central Processor and the individual tape units through the synchronizer.

Each synchronizer has two communication channels—one for reading and the other for either writing or reading; because these channels function independently, many tape-unit operations can proceed simultaneously. For example, the tape on one unit can be read while the tape on another unit is being written and while the Central Processor is continuing to execute the instructions in the program. In addition, other input-output operations can proceed at the same time through other channels. If a second UNISERVO IIIA synchronizer is included in the system, the number of possible simultaneous reads and writes is doubled. With two synchronizers, four tapes can be read simultaneously while central-processor operations continue.

Each UNISERVO IIIA unit may be used for reading forward or backward, or for writing forward. Any number of UNISERVO IIIA tape units may rewind simultaneously without affecting the use of others; the rewind instructions, however, are given separately. Gather-write and scatter-read features obviate the need for many internal data transfers associated with writing or reading a tape. When the gather-write feature is used, the computer can write data onto tape from memory areas which need not be adjacent; similarly, the scatter-read feature provides for the transfer of data from tape to areas of memory which also need not be adjacent.

A separate head check-reads data on the tape after it is written to help ensure writing accuracy. If an incorrectly written block is found it is marked automatically with a bad-spot pattern. When this occurs, the program is alerted so that the block in which the error occurred can be rewritten on the next available area of tape. Later, when the tape is read, the synchronizer detects this bad-spot pattern but does not transfer it to memory. Thus, the program can proceed without processing the invalid data.

The UNISERVO tape units are equipped with a separate power supply which accommodates a maximum of 16 tape units in the same line. Tape units not in the same line require an additional power supply.

■ PHYSICAL CHARACTERISTICS

The tape used with UNISERVO IIIA tape units is 1 mil thick, MYLAR* base, 0.5 inch wide, and 3600 feet long; data can be recorded on 3500 feet. Pulsed-phase modulation recording is used at a density of approximately 1000 frames per inch. Each frame contains nine bits, one in each of the nine recording channels across the tape. Three frames on tape constitute one 27-bit UNIVAC III word. The tape speed is 100 inches per second; thus six-bit alphanumeric characters can be transferred at the rate of 133,300 characters per second, while four-bit decimal digits can be transferred at the rate of 200,000 digits per second. A full 3600-foot reel of tape is rewound in 125 seconds.

The beginning of the usable portion of the tape (load point) and a point just before the end of the tape each has a clear area or window along the edge, 1⅛ inches long by ⅛ inch wide. Through these windows photoelectric cells detect the load point and the end-of-tape warning point. The load-point window is about 30 feet from the beginning of the tape along the inner edge. When a reel is mounted on a tape unit and the door is closed, tape is automatically advanced to the load point. The end-of-tape warning window is about 25 feet from the end of the tape along the outer edge.

TAPE LAYOUT

A block of data comprises one or more segments as shown in figure 3-1. When the program continually supplies the synchronizer with write FS's, tapes may be written without stopping the tape drive between blocks. When tapes are written in this continuous manner the interblock gap is approximately 0.4 inch; the gap for tapes which are not continuously written is approximately 0.6 inch.

*Registered trademark of the E. I. du Pont de Nemours & Company, Inc., Wilmington, Delaware.

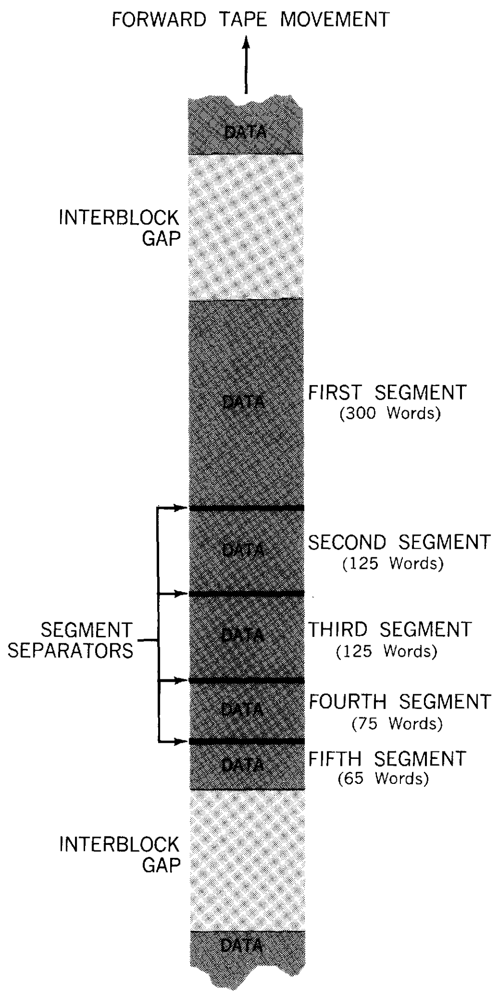


Figure 3-1. Typical Block of Data

The approximate number of blocks of data that can be written on 3500 feet of tape is given by the following expression:

$$\textit{Number of blocks} = \frac{14 \times 10^6}{W + n + 333G - 1}$$

where n = the number of segments in the block,
W = the number of UNIVAC III data words in the block,
and G = the interblock gap.

For example, the 690-word block shown in figure 3-1 contains five segments; the number of blocks of this type that can be continuously written on a reel of tape is

$$\frac{14 \times 10^6}{690 + 5 + 333(0.4) - 1} = 16{,}900 \text{ blocks}$$

The approximate time in milliseconds required to read or write a block (including the interblock gap) is

$$\text{Time in ms} = 0.03\,[W + (n-1)] + T$$

where W = the number of UNIVAC III words in the block,

n = the number of segments in the block,

and T = the time in milliseconds required to transport the interblock gap across the recording heads.

For reading or writing on the same or different tape units with the channel operating at full capacity, T lies in the approximate range of 4 to 8 milliseconds; the exact value depends on the type of tape operation involved.

TAPE PATH

Tape motion is forward when the tape is traveling from the left-hand supply reel (figure 3-2) to the right-hand takeup reel; the motion is backward when the tape is moving from the right-hand reel to the left-hand reel.

The tape clamp holds the beginning of the tape in place while reels are being mounted. Reels are removed from the tape unit simply by pressing the center of the hub. A tape wiper cleans the tape as it is used, and the tape drive mechanism has a vacuum contact which greatly reduces tape wear. The location of the read, write, and erase heads and of the photocells that detect the beginning and end of tape are shown in figure 3-2. The vacuum columns contain a length of slack tape which serves as a buffer between the fast-acting capstan and the slower-acting reel motors. Pressure switches sense the tape position in the vacuum columns to control the reel motors.

■ OPERATING CHARACTERISTICS

Data is transferred between tape units and memory, and tapes are rewound, by three types of operations.

SCATTER-READ AND GATHER-WRITE

Data from tape may be readily transferred from or to nonadjacent memory areas by using the gather-write and scatter-read function specifications (FS's).

When the gather-write feature is used, information stored in a number of nonadjacent groups of consecutive memory locations can be written onto tape in one block. The individual groups of consecutive memory locations, called segments, are separated on tape by an automatically written one-word pattern, called a segment separator. When a tape written in this manner is read, each segment separator is interpreted by the synchronizer but is not transferred to memory. The synchronizer alerts the Central Processor, which handles each segment of data separately, that the data to follow comes from a new segment. A block may contain one or more segments.

Gather-write and scatter-read FS's must be used in conjunction with a consecutive sequence of control words called SCAT control words. The FS contains the address of the first SCAT control word. Each control word specifies the starting location of the segment and the number of words in the segment, and has the following format:

0	Word Count	L-Address
25	24 16	15 1

Bit 25	Must be a 0, except for final (stop-transfer) SCAT control word
Word Count	Specifies, in binary, the number of words to be transferred. If all 0's, 4096 words are specified. If the word is a stop-transfer control word, any number of words may be specified.
L-Address	Specifies, in binary, the starting memory location of the segment. Must always specify a valid address.

When a stop-transfer control word for a scatter-read operation is accessed, reading is completed. If the block contains more data, that data is not transferred to memory; when that block has passed under the read head, the tape is positioned to read the next block.

Gather-Write

The gather-write operation is controlled by a series of SCAT control words located in consecutive memory locations. The control words are arranged in ascending order: the first one accessed is in memory location n, the second is in location $n + 1$, and so on. When the write function specified by one control word is completed, the next one is accessed until a stop-transfer control word is accessed. When a reel of tape is to be written, it must contain a plastic insert ring (enable-write ring); without this ring, the tape cannot be written.

Forward Scatter-Read

The forward scatter-read operation is controlled by a series of SCAT control words which are arranged and accessed in the same sequence as in the gather-write operation. The method of accessing these control words during scatter-read FS's is described below, and applies to both forward and backward read operations.

For scatter-read operations, a new control word is accessed when either a segment separator is detected by the synchronizer, or when all of the required words have been transferred to memory. For example, if the control word specifies 100 words and the segment is 150 words long, only the first 100 words of the segment are transferred to memory. The UNISERVO tape unit reads the last 50 words, but the synchronizer does not transfer them to memory. The tape under the read head is then properly positioned to handle the next segment.

A scatter-read operation ends when a stop-transfer control word (the final SCAT in a sequence of SCAT's) is accessed, or when the end of a block is detected on tape, whichever the synchronizer encounters first. If a stop-transfer control word is accessed before the entire block is read, the remainder of the block is read, but not transferred to memory. If the exact size of the segment varies or is unknown, the word count of the SCAT control word can be greater than the actual number of words on tape.

Backward Scatter-Read

The backward scatter-read operation functions like the forward scatter-read operation, with two exceptions. First, the memory-location sequence of the SCAT control words is reversed; if the first control word is in location n, the second must be in $n-1$, the third in $n-2$, and so on. Second, the L-address in bits 1 through 15 of the control word must specify the final (higher) address of the segment in memory. In this way segment order and word order within segments in memory can be preserved, regardless of the direction of tape read.

BLOCK-READ

Tapes may be read either forward or backward without control words. All segments of a block are read into sequential memory locations; segment separators, if present, are not transferred into memory. For a forward block-read, the L-address is the memory location to which the first word of the block is to be transferred. During a backward block-read, however, the last word written is the first word read and the L-address must be the location to which this word in the tape block is to be transferred. In this way the order of words within a block transferred to memory is the same regardless of the read direction.

There is no corresponding block-write instruction. Either the gather-write FS or the overwrite FS is used to transfer data from memory to tape.

REWIND INSTRUCTIONS

Two rewind FS's are provided, rewind-with-interlock and rewind-without-interlock. Either rewind can be executed through the read or the write channel of the synchronizer.

The channel through which a rewind-without-interlock FS is accessed remains busy while initiating the operation for at least 182 but not more than 357 microseconds. After the operation has been initiated, the channel is free to accept the next FS. Thus, one unit can rewind while other units read and write. However, if a rewind-with-interlock FS is specified, the synchronizer remains busy for about 11 milliseconds. If a rewind FS is accessed on one channel while the other channel is idle, both channels of the synchronizer remain busy for the times mentioned above.

Rewind Without Interlock

The rewind-without-interlock FS causes tape from the takeup reel to be passed to the supply reel until the load point is reached; at this point the rewind is complete, and the UNISERVO tape unit may be addressed. If a UNISERVO tape unit is addressed during a rewind, the busy indicator (table 3-2) is set, and automatic program interrupt occurs.

Rewind With Interlock

The rewind-with-interlock FS causes all tape to be rewound from the takeup reel onto the supply reel so that it can be removed from the UNISERVO tape unit. If a UNISERVO tape unit is addressed during or after rewind with interlock, the fault indicator (table 3-2) is set and automatic program interrupt occurs.

If interrupt is programmed, it will occur as soon as the synchronizer has initiated the rewind FS. All UNISERVO tape units may be rewound simultaneously, but the rewind FS's must be supplied separately.

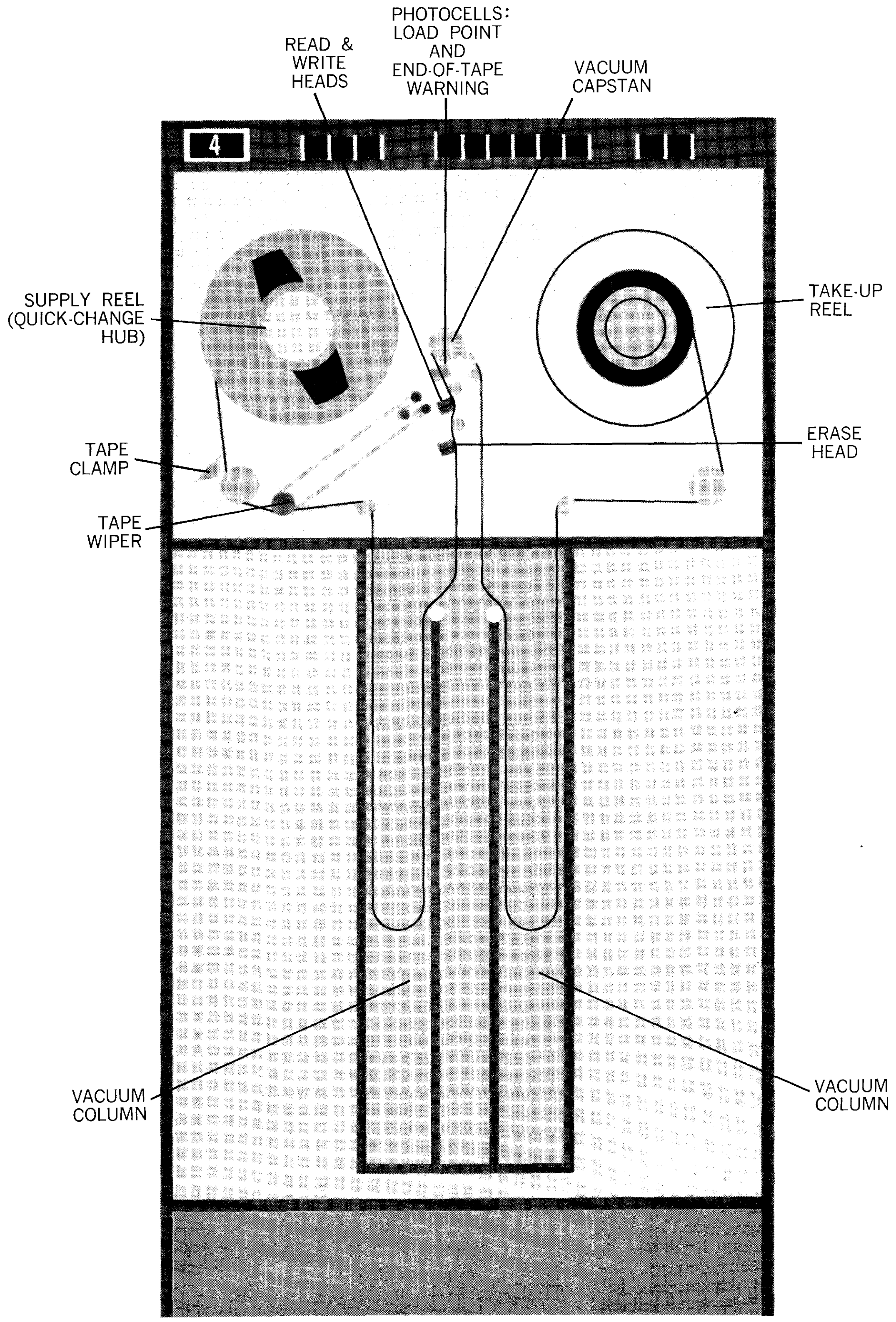


Figure 3-2. Tape Path

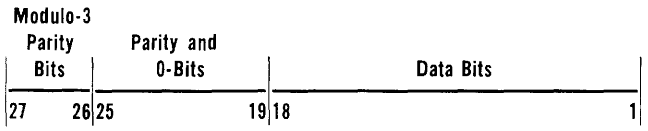
UNISERVO IIIA COMPATIBLE MODE

In addition to the usual UNIVAC III mode, individual bits in the 27-bit word can be arranged so that tapes used with UNISERVO IIIA units can also be used with the UNIVAC 490 Real Time System and the UNIVAC 1107 Thin-Film Memory Computer. This additional bit configuration is handled automatically by the UNISERVO IIIA synchronizer circuitry.

A switch is installed on each tape unit in the UNIVAC III System which is used to process UNISERVO IIIA compatible-mode tapes. This switch has two positions, one for the usual UNIVAC III mode and one for the compatible mode. Memory bit positions for writing and reading compatible-mode tapes are illustrated in figure 3-3.

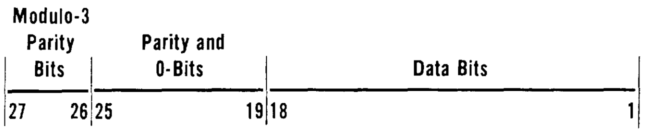
The compatible mode has 18 data bits and the UNIVAC III mode has 25. When writing compatible-

WRITING UNISERVO IIIA COMPATIBLE-MODE TAPES



Modulo-3 Parity Bits . . .	Not transferred to tape
Parity and 0-Bits	Bits 19 through 25 are not transferred to tape. Synchronizer inserts UNISERVO IIIA compatible-mode parity bits and 0-bits on tape.
Data Bits	Bit positions 1 through 18 are transferred to tape.

READING UNISERVO IIIA COMPATIBLE-MODE TAPES



Modulo-3 Parity Bits . . .	Automatically inserted in memory by the synchronizer
Parity and 0-Bits, Data Bits	Bits 1 through 25 are transferred to memory as they appear on tape.

Figure 3-3. Word Formats in Memory for Writing and Reading UNISERVO IIIA Compatible-Mode Tapes

mode tapes, only memory bit positions 1 through 18 are transferred to tape. The synchronizer writes either 0's or compatible-mode parity bits onto tape in the bit positions that correspond to memory positions 19 through 27. When tape is read, bits 1 through 25 are transferred to memory as they appear on tape, and the required UNIVAC III parity bits are inserted automatically in positions 26 and 27.

Compatible-mode blocks must be written from consecutive memory locations using block-read instructions; tape movement must be stopped between blocks.

The following list is a summary of UNISERVO IIIA tape-unit specifications:

PROGRAM-CONTROLLED FUNCTIONS:
Gather-write,
Block- or scatter-read,
Rewind with or without interlock,
Tape-error recovery, and
Load-point test.

TAPE SPEED:
Read or write—100 inches per second
Rewind—125 seconds for a 3600-foot reel of tape

RECORDING:
9 channels; 1000 frames per inch; the 27 bits of a UNIVAC III word occupy 3 frames

CHARACTER-TRANSFER RATES:
Alphanumeric characters—133,000 per second
Decimal digits—200,000 per second

BLOCK LENGTH:
Variable—under program control

INTERBLOCK GAP:
Approximately 0.6 inch

TAPE CHARACTERISTICS:
Base—MYLAR
Coating—oxide
Width—½ inch
Length—3600 feet (3500 feet recordable)
Thickness—1 mil

DIRECTION:
Read—forward or backward
Write—forward

CHECKING:
Information written onto tape is checked by a read head after being written;
Information read from or written onto tape is modulo-3 checked;
Bad spots on tape are marked with special bit configurations which are not transferred to memory; and
Load point and end-of-tape warning are detected photoelectrically.

■ FUNCTION SPECIFICATIONS

To perform a programmed tape operation, the UNISERVO IIIA tape-unit synchronizer requires an initiate-input-output-function (IOF) instruction and a function specification (FS). The IOF instruction identifies the UNISERVO IIIA input-output channel to be used and specifies the address of the FS.

Details of the tape operation are specified in the FS, which has the following format:

	Tape Unit	Function Code	I	L-Address
25	24 21	20 17	16	15 1

Bit 25	Must be a binary 0
Tape Unit	Binary representation of tape unit to be used
Function Code	Specifies operation to be performed on designated tape unit
I	Controls interrupt—if it is a 1-bit, a program interrupt occurs when execution of the FS is completed; otherwise, this interrupt will not occur.
L-Address	For block-reads, used as base address (in binary); for gather-writes and scatter-reads, represents location of first SCAT control word.

The individual function codes with their mnemonics are described under the headings which follow.

GATHER-WRITE—GWT

Operation: Gather-write onto T*n*

Function Code: 0001

Description: Write one block onto the tape on the designated tape unit from the memory locations specified by the SCAT control words.

NOTES

1. The SCAT control words must be in consecutive ascending memory locations, beginning at the location specified by the L-address of the GWT FS.
2. At least one word must be written onto the tape on the designated unit.

EXAMPLE

A sample GWT operation is as follows:

Write the contents of the following memory locations onto the tape on unit 2, with interrupt when execution of the FS is completed:

Memory Locations	*Tape Segment*
1901 through 2200	First
1400 through 1524	Second
1000 through 1124	Third
1602 through 1676	Fourth
1800 through 1864	Fifth

The FS used to gather-write this data is as follows:

	Tape Unit	Function Code	I	L-Address
25	24 21	20 17	16	15 1
0	0010	0001	1	800

The SCAT control words for the preceding GWT FS begin in memory location 800 and contain the following information:

Control-Word Address	*Control-Word Bit Positions*		
	25	*24 through 16 (Word Count)*	*15 through 1 (L-Address)*
800	0	0300	1901
801	0	0125	1400
802	0	0125	1000
803	0	0075	1602
804	0	0065	1800
805	1	0000	0000

The tape block for this example is shown in figure 3-1.

OVERWRITE—OWT

Operation: Overlay pattern → T*n*,
gather-write → T*n*

Function Code: 0101

Description: After a forward read or write FS, write overlay pattern so that erased tape is positioned under the write head before writing; then write the contents of the memory locations specified by the SCAT control words onto the tape on the designated tape unit.

NOTES

1. The SCAT control words must be in consecutive ascending memory locations, beginning with the locations specified by the OWT FS L-address.
2. The area of tape to be overwritten must be a block of at least 150 words, or another overwrite pattern.
3. At least one word must be written onto tape by the OWT FS.
4. Data read from an OWT block originally written after a read FS is treated the same as a bad-spot pattern: the error-A indicator is set and interrupt occurs.
5. Data from an OWT block originally written after a write FS is read normally.

FORWARD SCATTER-READ—FSR

Operation: Forward scatter-read from T*n*

Function Code: 0000

Description: Read forward one block from the designated tape unit into the memory locations specified by SCAT control words.

NOTES

1. The SCAT control words must be in consecutive ascending memory locations, beginning with the location specified by the L-address of the FSR FS.
2. If the first SCAT control word is a stop-transfer control word (a 1 in bit position 25), the entire block is skipped, but the words are modulo-3 checked.

EXAMPLE

The block shown in figure 3-1 is to be read forward from the tape on unit 2 into memory as follows; interrupt is to occur when the execution of the FS has been completed:

Segment	*Memory Locations*
First (first 110 words)	2000 through 2109
Second (all 125 words)	1000 through 1124
Third (first 50 words)	2110 through 2159
Fourth (all 75 words)	3010 through 3084
Fifth—skip all words	—

The FSR FS for this transfer is as follows:

	Tape Unit	Function Code	I	L-Address
25	24 21	20 17	16	15 1
0	0010	0000	1	700

The SCAT control words for the preceding FSR FS begin in memory location 700 and contain the following information:

Control-Word Address	*Control-Word Bit Positions*		
	25	*24 through 16 (Word Count)*	*15 through 1 (L-Address)*
700	0	0110	2000
701	0	0125	1000
702	0	0050	2110
703	0	0075	3010
704	1	0000	0000

BACKWARD SCATTER-READ—BSR

Operation: Backward scatter-read from T*n*

Function Code: 0010

Description: Read backward one block from the designated tape unit into the memory locations specified by SCAT control words.

NOTES

1. The SCAT control words must be in consecutive descending memory locations, beginning with the location specified by the L-address of the BSR FS.
2. If the first SCAT control word is a stop-transfer control word (a 1 in bit position 25), the entire block is skipped, but the words are modulo-3 checked.

EXAMPLE

The block shown in figure 3-1 is to be read backward from tape unit 3 into memory as follows; interrupt is to occur when the FS has been completed:

Segment	*Memory Locations*
Fifth (all 65 words)	1500 through 1564
Fourth (all 75 words)	1700 through 1774
Third (all 125 words)	1800 through 1924
Second (last 25 words)	2000 through 2024
First (all 300 words)	2025 through 2324

The BSR FS used for this transfer is as follows:

	Tape Unit	Function Code	I	L-Address
25	24 21	20 17	16	15 1
0	0011	0010	1	650

Tape control words for the preceding BSR FS begin in memory location 0650 and contain the following information:

Control-Word Address	*Control-Word Bit Positions*		
	25	*24 through 16 (Word Count)*	*15 through 1 (L-Address)*
650	0	0065	1564
649	0	0075	1774
648	0	0125	1924
647	0	0025	2024
646	0	0300	2324
645	1	0000	0000

FORWARD BLOCK-READ—FBR

Operation: Forward block-read 1 block, T*n* → L . . .

Function Code: 0100

Description: Read forward one block from the designated tape unit into consecutive ascending memory locations, beginning with the L-address of the FBR FS.

BACKWARD BLOCK-READ—BBR

Operation: Backward block-read 1 block, T*n* → L . . .

Function Code: 0110

Description: Read backward one block from the designated tape unit into consecutive descending memory locations, beginning with the L-address of the BBR FS.

BACKWARD-CONTINGENCY BLOCK-READ—BCBR

Operation: Backward-contingency block-read, T*n* → L . . .

Function Code: 1110

Description: Position the tape over the erased-tape gap that follows the block in which an error B occurred; then read the error block backward into consecutive descending memory locations beginning at the L-address of the BCBR FS.

NOTES

1. This FS is used for recovering from a condition that set the error-B indicator while tape was moving forward.
2. If the backward read is performed without error, the next read may be a forward-contingency scatter-read or a forward-contingency block-read.

BACKWARD-CONTINGENCY SCATTER-READ—BCSR

Operation: Backward-contingency scatter-read from T*n*

Function Code: 1010

Description: Position the tape over the erased-tape gap that follows the block in which an error B occurred; then read the error block backward into the memory locations specified by the SCAT control words.

NOTES

1. This FS is used for recovering from a condition that set the error-B indicator while tape was moving forward.
2. If the backward read is performed without error, the next read may be a forward-contingency scatter-read or a forward-contingency block-read.
3. The SCAT control words must be in consecutive descending memory locations, beginning with the location specified by the L-address of the BCSR FS.

FORWARD-CONTINGENCY BLOCK-READ—FCBR

Operation: Forward-contingency block-read, T*n* → L . . .

Function Code: 1100

Description: After a successful backward-contingency scatter-read (BCSR) or backward-contingency block-read (BCBR), skip one block forward on the designated tape unit; then read the next block into consecutive ascending memory locations beginning at the L-address of the FCBR FS.

NOTES

1. Normal reads can proceed after this function is completed.
2. This FS is not used after an error-B condition occurred while tape was being read backward, because for backward reads the tape is positioned correctly and recovery can be attempted by a normal forward read.

FORWARD-CONTINGENCY SCATTER-READ—FCSR

Operation: Forward-contingency scatter-read from T*n*

Function Code: 1000

Description: After a successful backward-contingency scatter-read (BCSR) or backward-contingency block-read (BCBR), skip one block forward on the designated tape unit; then read the next block into the memory locations specified by the SCAT control words.

NOTES

1. The SCAT control words must be in consecutive ascending memory locations, beginning with the location specified by the L-address of the FCSR FS.
2. This FS is not used after an error-B condition occurred while tape was being read backward, because for backward reads the tape is positioned correctly and recovery can be attempted by a normal forward read.
3. Normal reads can proceed after this function is completed.

REWIND WITHOUT INTERLOCK—RW

Operation: Rewind T*n* to load point

Function Code: 0011

Description: Rewind the tape on the designated tape unit to the load point, where it is in position for reading or writing the first block.

NOTE

The L-address is ignored.

REWIND WITH INTERLOCK—RWI

Operation: Rewind T*n* with interlock

Function Code: 0111

Description: Rewind the tape on the designated tape unit to the beginning and interlock the tape unit until the reel is changed.

NOTES

1. Pressing the CHANGE TAPE button-light on the tape-unit control panel also will release the interlock.
2. The L-address is ignored.

LOAD POINT TEST—LPT

Operation: Test T*n* for load point

Function Code: 1111

Description: Test the tape on the designated tape unit to determine whether the tape is positioned at the load point. If the tape is at the load point and interrupt is specified, set the successful-completion indicator* and interrupt the program; if the tape is not at the load point, set the error-B and fault indicators*.

NOTES

1. If the designated tape unit is busy when this FS is given, only the busy indicator* is set.
2. If a fault exists in the designated tape unit, only the fault indicator* is set.

Function codes for UNISERVO IIIA tape-unit operations, together with the mnemonic codes for these operations, are summarized in table 3-1.

Table 3-1. Summary of Function Codes for UNISERVO IIIA Tape Units

Function	Function Code: Bit Positions				Mnemonic Code
	20	19	18	17	
Gather write	0	0	0	1	GWT
Overwrite	0	1	0	1	OWT
Forward scatter-read	0	0	0	0	FSR
Backward scatter-read	0	0	1	0	BSR
Forward block-read	0	1	0	0	FBR
Backward block-read	0	1	1	0	BBR
Backward-contingency block-read	1	1	1	0	BCBR
Backward-contingency scatter-read	1	0	1	0	BCSR
Forward-contingency block-read	1	1	0	0	FCBR
Forward-contingency scatter-read	1	0	0	0	FCSR
Rewind without interlock	0	0	1	1	RW
Rewind with interlock	0	1	1	1	RWI
Load-point test	1	1	1	1	LPT

■ INTERRUPT INDICATORS

Each channel of the UNISERVO IIIA tape system is provided with the following set of program-testable interrupt indicators:

Indicator	*Mnemonic Code*
Successful completion	2
Fault	7
Busy	4
End-of-tape warning	6
Error A	3
Error B	5

*These indicators are described under the heading *Interrupt Indicators.*

Specific conditions which set one or more of these indicators are summarized in table 3-2.

Whenever an FS cannot be successfully completed because of an error or fault condition after initiation, the synchronizer does not request memory access to the standby location, regardless of the condition of the standby-location interlock indicator. When a specific channel is inaccessible because of a tape-unit error, the tape unit which caused the error may be addressed through the other channel without resetting the error indicators of the first channel.

When a read-channel error is detected, the indicators for that channel are set at the termination of the execution of the FS during which the error occurred. Write-channel indicators are set when an error is detected.

SUCCESSFUL COMPLETION

When an FS is completed, the successful-completion indicator is set if input-output interrupt is specified and if no errors or other unusual circumstances are associated with the completion of the FS. The synchronizer never sets this indicator in combination with any other indicators, except the end-of-tape-warning indicator.

FAULT

The fault indicator is set mostly by conditions which require corrective action by operator or maintenance personnel. It is also set to indicate that tape is not at its load point and to indicate that an FS that specifies a write order has been given to the read channel.

BUSY

The busy indicator is set to indicate some FS errors (described in table 3-2) or to indicate that the addressed UNISERVO tape unit is either rewinding without interlock or is being used by the other synchronizer channel.

END-OF-TAPE WARNING

The end-of-tape-warning indicator is set to alert the program that the end-of-tape window has been detected by the end-of-tape-warning photocell during the previous FS. Successful completion of the tape operation is not prevented, and the successful-completion indicator is set about 702 microseconds after the end-of-tape-warning indicator is set, unless an error has occurred. Another FS cannot be accessed while the end-of-tape-warning indicator is set.

Table 3-2. Interrupt Conditions, UNISERVO IIIA Tape Unit

Indicator(s) Set	Condition	Description
Fault	UNISERVO tape-unit selection fault*	The UNISERVO tape unit specified cannot be addressed because either the wires of the UNISERVO selector plugboard (described under the heading "Control Features") are not properly connected or a nonexistent unit was addressed.
	FS error*	An FS specifying a write order has been received by a read channel.
	UNISERVO tape unit unavailable*	The UNISERVO tape unit addressed requires manual intervention due to any of the following conditions: Power to the UNISERVO tape unit is off; The unit has been interlocked by a previous FS; or A write operation has been specified on a reel which does not contain an enable-write ring.
Error B, fault	Tape not at load point	Tape is not at the load point when the load-point-test FS is given.
Busy	UNISERVO tape unit busy*	The UNISERVO tape unit addressed is either rewinding without interlock or is being used by the other channel.
Busy, error B	FS memory-address error, write channel*	A memory-address error occurred while attempting to access an FS from memory on a write channel.
	FS memory-address error, read channel*	A memory-address error occurred while attempting to access an FS from memory on a read channel.
	FS modulo-3 check, write channel*	An error was detected during the modulo-3 check while an FS was being read from memory for the write channel.
	FS modulo-3 check, read channel*	An error was detected during the modulo-3 check while an FS was being read from memory for the read channel.
Error B	Beginning of a block cannot be detected, read channel	The synchronizer cannot detect where the data starts.
	Data-word modulo-3 error, read channel	A modulo-3 error was detected when a data word was being transferred to memory.
	Overskew, read channel	The frames of data on tape are not sufficiently close to being parallel to the read head. The synchronizer will automatically correct for up to four frames of skew.
Error A	Bad-spot pattern	During a forward read, a bad-spot pattern was detected at the end of a block, indicating that when the data was originally written it failed the write check, or indicating that when the tape was originally written, an overwrite FS was given after a forward read. (During a backward read, if a bad-spot pattern is detected, the incorrectly written block is bypassed and the next valid block is read; no error indicators are set for the bypassed block.)
	Memory-address error, write-channel data word	An address error occurred when a data word was being accessed from memory.
	Memory modulo-3 error, write channel	A modulo-3 error was detected when a data word was being accessed from memory.
	Modulo-3 error on data that has been written onto tape	A modulo-3 error was detected by the write-check circuits in a data word that has been written onto tape.
	Overskew, write channel	The frames of data on tape are not sufficiently close to being parallel to the read head. The synchronizer will automatically correct for up to four frames of skew.
End-of-tape warning	End-of-tape warning	The end-of-tape-warning photocell in the tape feed path has detected the end-of-tape window, and the next tape FS for the same UNISERVO unit has been completed. (Approximately 702 microseconds later the successful-completion indicator is set.)
Error A, fault	UNISERVO fault	An attempt has been made to write beyond the physical end of tape, or to read backward beyond the load point, or a power loss has occurred.

*When these conditions occur, the attempted execution of the FS does not cause tape movement.

ERROR A AND ERROR B

These two indicators are set by several abnormal conditions as described in table 3-2.

The error-A indicator is used primarily to indicate that a tape has been incorrectly written and has been marked with a bad-spot pattern. The error-A indicator is not set for a backward-read error, except in conjunction with the fault indicator for a tape-unit fault condition. When writing onto a tape, errors may be detected during the writing or during the check-reading which occurs after writing. In either case, when an error is detected, a bad-spot pattern is written and the error-A indicator is set. The program then rewrites the block in which the error occurred. Later, when the tape is read forward, this bad-spot pattern is detected by the synchronizer. If the entire block is read correctly, the synchronizer skips the next valid duplicate block; if the entire block is not read correctly, the error-A indicator is set and interrupt occurs. The next instruction of the program can then specify that the next valid block is to be read. If the bad-spot pattern is detected during a backward read, the synchronizer skips the error block and reads the next valid block of data.

The error-B indicator is used primarily to indicate a read error. The software system attempts to recover from such an error by rereading the block in which the error occurred.

■ EXECUTION OF FUNCTION SPECIFICATIONS

The normal sequence of operations performed during the execution of tape-unit function specifications varies with the type of specification.

For forward scatter-read or gather-write function specifications, the sequence is as follows:

1. The IOF transfers the FS to the standby location for the appropriate channel and sets the standby-location interlock indicator for that channel; this indicator is used by the tape synchronizer and Central Processor to determine whether or not an FS is available for processing. Standby-location addresses for the UNISERVO IIIA tape-unit channels are as follows:

Channel	*Address*
Write channel, basic	0003
Read channel, basic	0004
Write channel, additional	0014
Read channel, additional	0015

   If another IOF addresses the same channel while the standby indicator is set, a second FS could overlay the first FS in the standby location. Thus, the indicator for the channel involved normally is tested to determine whether it is reset before an IOF instruction is issued. If a second FS is completed before the program resets the successful-completion indicator, the synchronizer will not set it again nor will the synchronizer transfer an FS from memory until the program has reset it.

2. After the IOF has set the standby-location interlock indicator and loaded the standby location with an FS, the Central Processor proceeds to the next instruction. At the same time, the synchronizer operation in progress continues. When the synchronizer becomes available after successfully completing the previous FS, and if the standby indicator is set, the FS from the standby location is transferred to the synchronizer; the standby indicator then is reset.

3. The FS from the standby location is decoded by the synchronizer circuitry to designate and specify the function to be performed, and the unit on which to perform it. The L-address of the FS is stored in the memory-address counter (MAC); the SCAT control word is transferred from the memory location specified by the MAC to the tape control-word register (TCWR) associated with the channel being used. Thus, the TCWR contains the number of words to be transferred and the initial address of the segment in memory.

4. The transfer of data is initiated and, as each data word is transferred to memory, the segment base address is incremented so that it represents the address of the next data word to be transferred. The word count is decremented as each word is transferred.

5. The address of the next SCAT control word is stored in the MAC; when the segment separator is sensed, this address is used to access memory for the next SCAT control word. The next SCAT control word is transferred to the TCWR and the MAC is incremented. The process is repeated until all of the data in the block has been transferred.

6. Interrupt occurs if it was specified in the FS (a 1 in bit position 16).

For a backward scatter-read the sequence of operations is the same as for a forward scatter-read or gather-write, except that the MAC containing the address of the next SCAT control word is successively decremented, as is the base-address portion of the TCWR. This dual decrementing requires that for backward scatter-reads the SCAT control words must be accessed in reverse order and the control-word base address must be the highest location in the segment.

For a forward or backward block-read, the sequence is the same as for a forward scatter-read or gather-write, except that, because SCAT control words are not required, the L-address of the FS is the memory base address of the block to be transferred; this address is stored in the MAC during incrementing.

■ CONTROL FEATURES

CONTROL PANEL

Each UNISERVO IIIA unit is equipped with the control panel shown in figure 3-4; the panel is used for indicating the physical condition of the UNISERVO tape unit, for the operational control of the tape, and for power on-off control. The function of each control and light on the panel is given in table 3-3.

SELECTOR PLUGBOARD

A plugboard mounted in a cabinet between the power supply and the first UNISERVO tape unit may be used to change the numeric designation of the individual UNISERVO tape units.

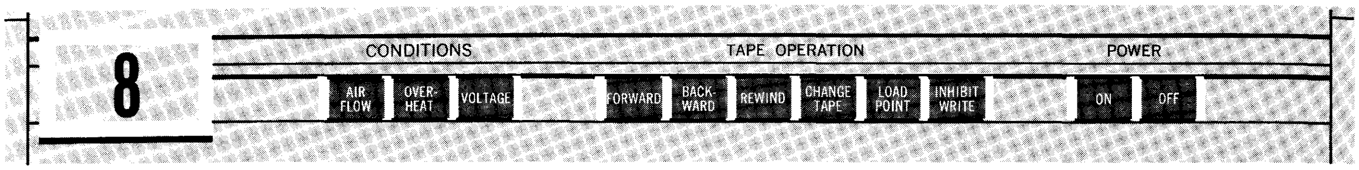


Figure 3-4. Control Panel, UNISERVO IIIA Tape Unit

Table 3-3. Function of Lights and Button-Lights on UNISERVO IIIA Control Panel

LIGHTS

Panel/Light Marking	Conditions Under Which Lighted
CONDITIONS	
AIR FLOW	Insufficient cooling air flowing through tape unit; power is removed from unit.
OVERHEAT	Temperature in tape unit is above normal; power is removed from unit.
VOLTAGE	A circuit breaker has tripped in unit; power is removed from unit.
TAPE OPERATION	
LOAD POINT	Tape is at load point. Turns off after first read or write operation on tape unit is completed.
INHIBIT WRITE	Reel mounted on unit does not contain an enable-write ring.
UNISERVO numeral*	Lights red when either the tape clamp is not positioned correctly, or tape-unit door is not shut. Lights white when unit can be operated. Does not light when power is off.

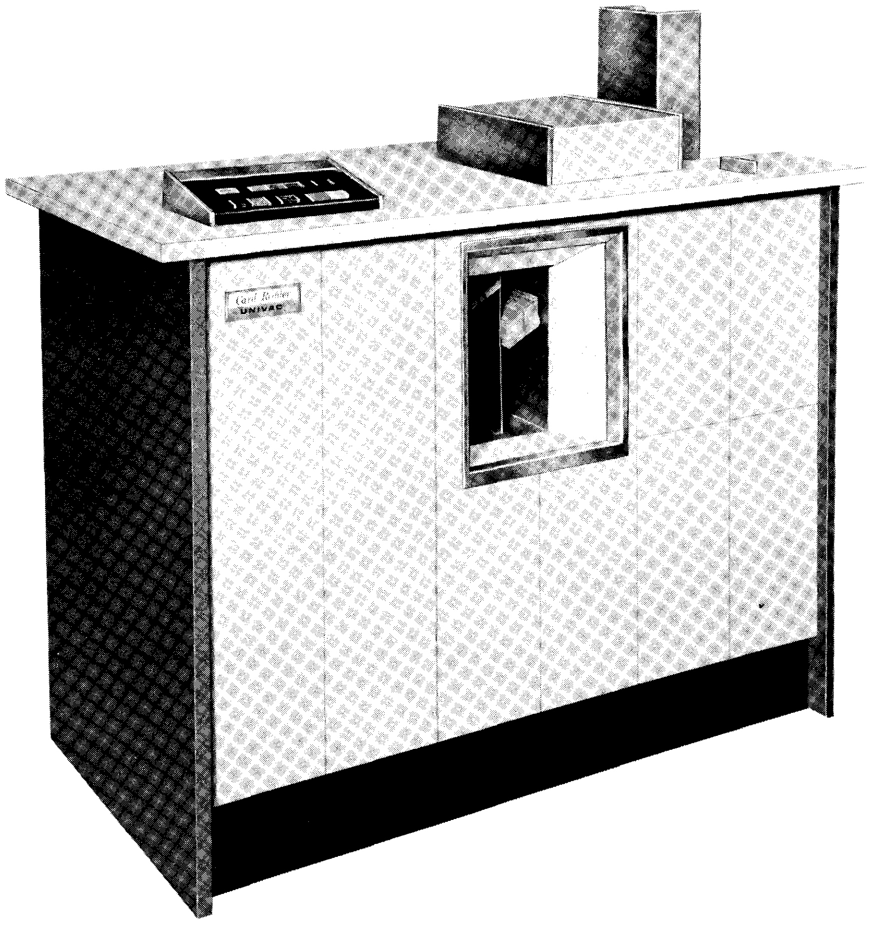
BUTTON-LIGHTS

Panel/Button Marking	Function
TAPE OPERATION	
FORWARD	Manually sets tape unit for forward operation†; button is lit. Also is lit when unit has been set for forward operation by program.
BACKWARD	Manually sets tape unit for backward operation†; button is lit. Also is lit when unit has been set for backward operation by program.
REWIND	Manually causes tape to rewind with interlock; button is lit. Also is lit when tape is being rewound under program control.
CHANGE TAPE	Lights when a tape is rewinding with interlock. If pressed before another tape reel is mounted, tape will be advanced to load point.
UNISERVO IIIA compatible-mode switch	Optional pushbutton switch, installed on tape units that are to handle tapes written in UNISERVO IIIA compatible mode.
POWER	
ON	Turns on power to unit and is lit. Will not be lit if tape loops in vacuum columns are incorrectly positioned.
OFF	Turns off power to unit and is lit. Also is lit if power is removed by abnormal air flow, overheat, or voltage; condition can be determined by examining lights on tape-unit panel.

*Removable plastic numeral block which designates the number of the tape unit, 0 through 15.
†Used primarily for maintenance.

SECTION IV

HIGH-SPEED READER



The High-Speed Reader is available as either an 80-column model or a 90-column model. Both may be included in one UNIVAC III System. The synchronizer and power supply for the reader are housed within the reader cabinet. Cards are read and checked automatically at the rate of 700 cards per minute. The 80-column reader processes standard Hollerith card code and translates it into the UNIVAC III character code; it can also process any other 80-column card code. The 90-column reader processes 90-column Remington Rand card code and translates it into the UNIVAC III character code; it can also process any other 90-column card code.

As with other peripheral units, the automatic program-interrupt feature may be used with the reader. When this feature is used, the reader synchronizer sends a signal to the Central Processor when a function specification (FS) has been initiated. This signal is used to alert the central-processor program that the standby location for the reader is ready to receive another FS. Thus, the operation of the reader is not delayed and it can function at its full capacity of 700 cards per minute. The interrupt feature also is used to alert the Central Processor if an abnormal condition exists in the reader.

The High-Speed Reader receives data and FS's from the Central Processor through one of the eight general-purpose channels.

■ OPERATING CHARACTERISTICS

The card-feed path of the reader (figure 4-1) includes a card-input magazine, four card stations, and three output stackers. In figure 4-1, cards are shown at the start of an FS. The card stations are numbered 1 through 4, and the stackers are designated 0, 1, and 2. Card station 1 is the first read station. Card station 2 is the second read station; card images from this station are transferred to memory. Cards are transported by means of continuously moving rollers which advance cards from the input magazine through the two read stations, to card station 3, card station 4, and finally to one of the three output stackers.

OPERATION

The cards to be read are moved into the card path by the picker knife, which is program-controlled. At the first read station, the card is brush-sensed, and a hole count is stored for checking purposes. After the card is read at the second read station, hole counts from the two read stations are compared. If an error is detected, the program-testable data-error indicator is set and automatic program interrupt occurs.

INPUT MAGAZINE

Cards to be read are stored in the input magazine, which holds 2000 cards. Excellent card-feed reliability is achieved through the use of a vacuum which helps to position the card to be engaged by the picker knife. When the input magazine is empty or a misfeed occurs, the unit stops, the MISFEED light of the reader control panel lights, the program-testable operator-oversight indicator is set, and automatic program interrupt occurs.

OUTPUT STACKERS

Each of the three output stackers holds 1000 cards. When a stacker is full, the STACK FULL light on the reader control panel lights, the program-testable operator-oversight indicator is set, and automatic program interrupt occurs. All cards enter stacker 0 unless the program specifies stacker 1 or 2.

CARD IMAGE

Cards may be read either with or without automatic translation from the card code to the UNIVAC III character code. Figures 4-2 and 4-3 illustrate the data flow for 80-column cards, and figure 4-4 illustrates the data flow for 90-column cards.

When an 80-column Hollerith-code card is automatically translated, the card image occupies 20 UNIVAC III alphanumeric words in memory, as indicated in figure 4-2; an 80-column card image transferred without translation occupies 40 alphanumeric words. The first untranslated 24-bit word is represented by a card field in the upper left portion of the card, four columns wide by six rows deep, as shown in figure 4-3. Because cards are read without translation, non-Hollerith codes may be used. Bit configurations in memory for non-Hollerith codes, when reading with translation is specified, are given in Appendix E. For both Hollerith and non-Hollerith codes, a 0-bit is placed in each sign-bit position in memory when a card is read.

When a 90-column card image is transferred to memory, either with or without translation, it occupies 24 alphanumeric words, as shown in figure 4-4. Binary 0's are inserted in the three least significant character positions of the 12th word and of the 24th word of the card image in memory. A 0-bit is placed in each sign-bit position in memory when a card is read.

Hollerith and Remington Rand 90-column card codes are given in tables 4-1 and 4-2, respectively.

CARD CHECKING

After the synchronizer has completed transferring the data from a card, the number of memory accesses is checked to verify that an entire card image has been transferred to memory. Only the image from the second read station is transmitted to memory. The hole counts from the two read stations are compared. A modulo-3 check is made on each word that is transferred. If any of these checks detects an error, a program-testable indicator is set, and an automatic program interrupt occurs.

EXECUTION CYCLE

The timing of the card cycle, including the times at which program-testable indicators are set, is illustrated in figure 4-5. An initiate-input-output-function (IOF) instruction transfers the FS to the standby location for the High-Speed Reader and sets the standby-location interlock indicator for the High-Speed Reader. The Central Processor continues with the next instruction. When the synchronizer becomes available after completing a card cycle at point *A*, it tests the standby indicator. If the indicator is found to be set, the FS in the standby location is automatically transferred to the synchronizer for execution, and the standby indicator is reset. When the indicator is reset and input-output interrupt is specified (bit 16 of the FS is 1), the initiation indicator is set and

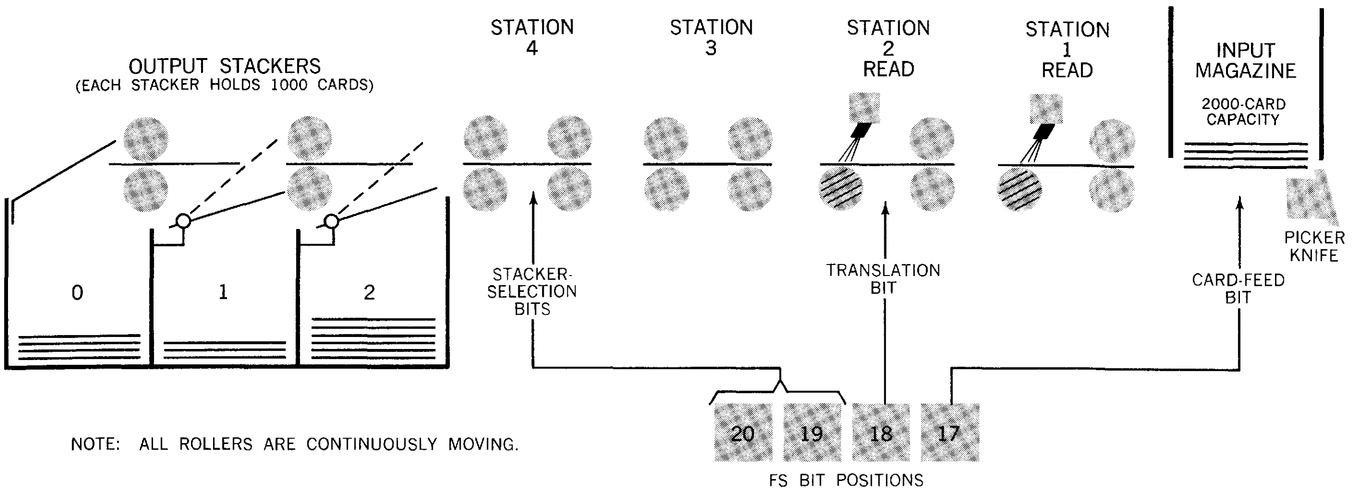


Figure 4-1. Card-Feed Path, High-Speed Reader

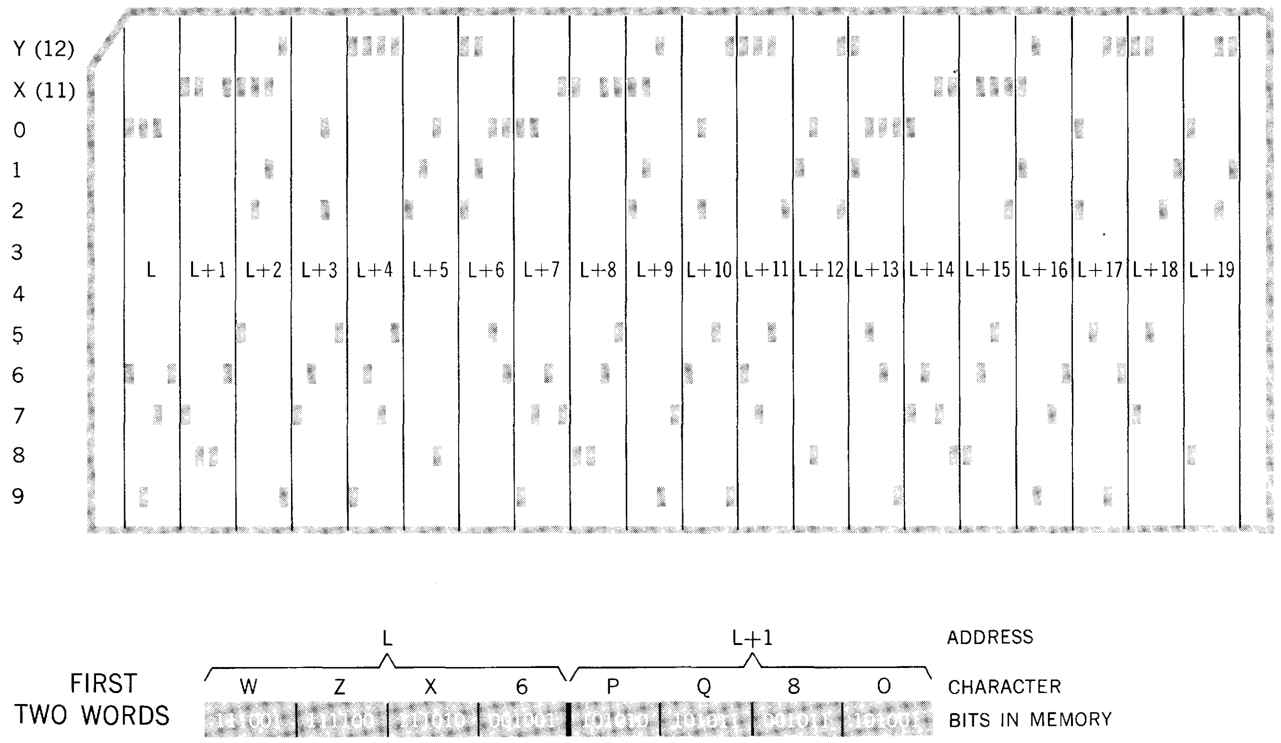


Figure 4-2. Data Transfer from Reader to Memory, With Translation, 80-Column Card

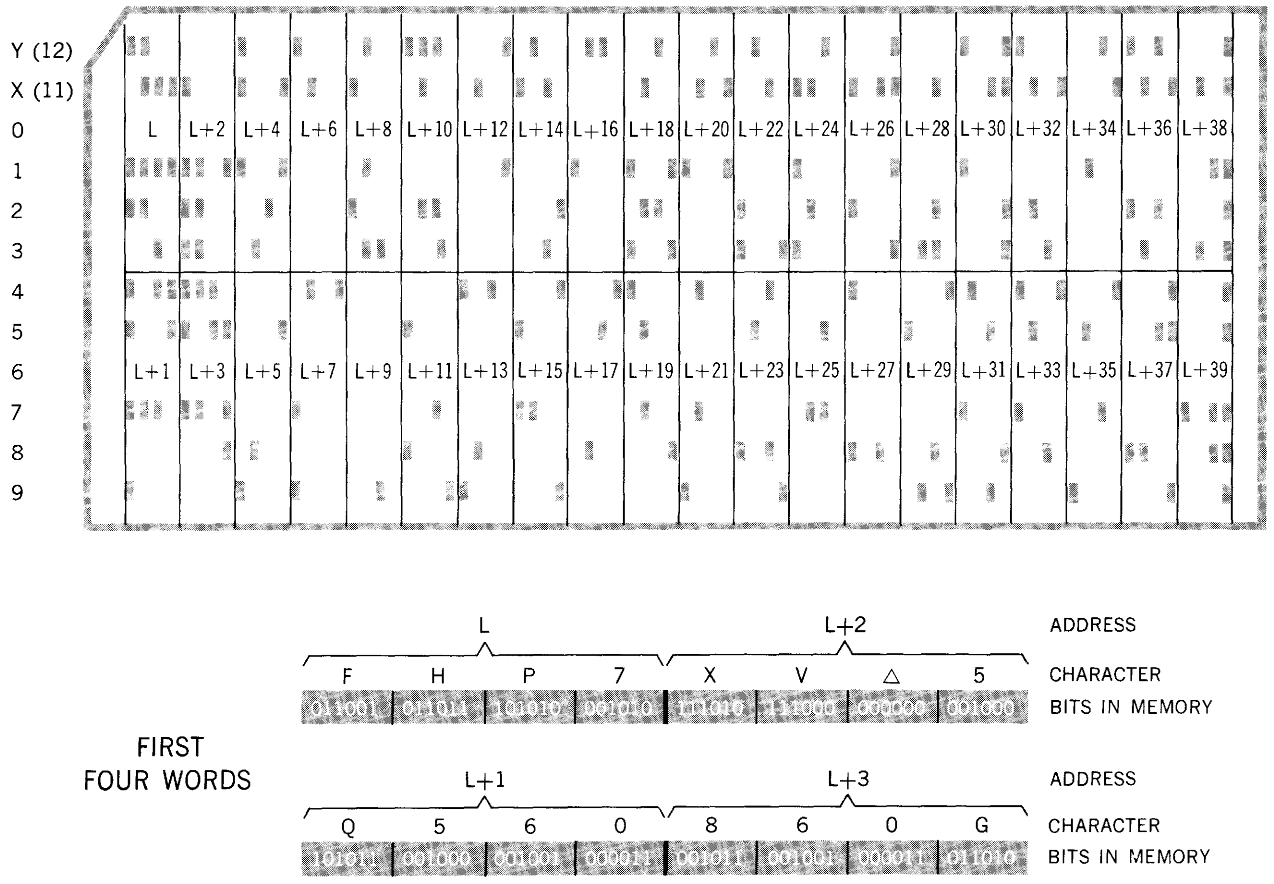


Figure 4-3. Data Transfer from Reader to Memory, Without Translation, 80-Column Card

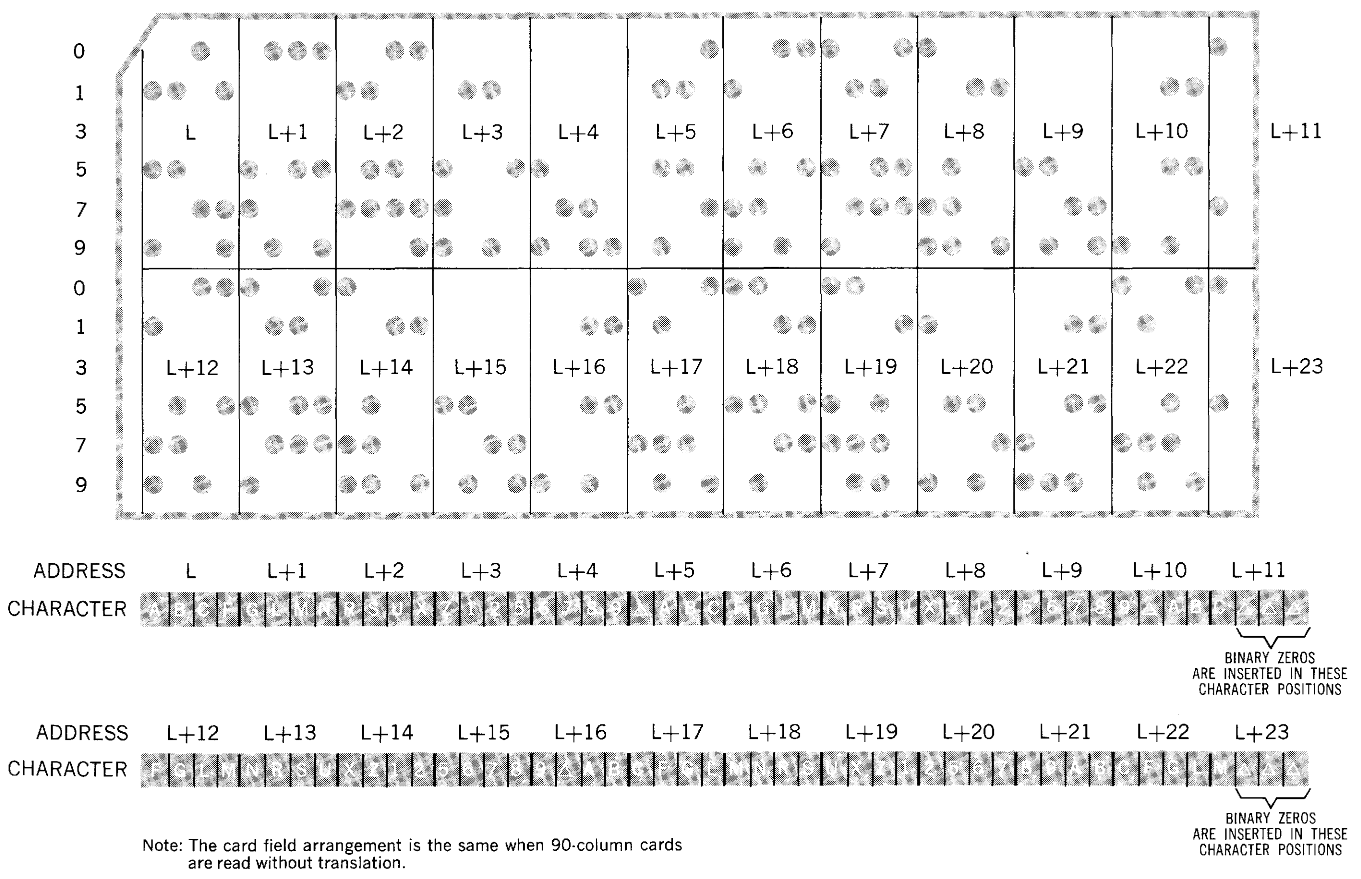

0
1
3
5
7
9
0
1
3
5
7
9
L
L+1
L+2
L+3
L+4
L+5
L+6
L+7
L+8
L+9
L+10
L+11
L+12
L+13
L+14
L+15
L+16
L+17
L+18
L+19
L+20
L+21
L+22
L+23
ADDRESS
CHARACTER
BINARY ZEROS ARE INSERTED IN THESE CHARACTER POSITIONS
Note: The card field arrangement is the same when 90-column cards are read without translation.

Figure 4-4. Data Transfer from Reader to Memory, With Translation, 90-Column Card

Table 4-1. Hollerith Code, High-Speed Reader

The upper entry represents the card punching positions.
The lower entry represents the corresponding High-Speed Printer character.
NP indicates a code which is not printed by the High-Speed Printer.
NS indicates nonstandard codes.

Numeric Bits	Zone Bits			
	00	01	10	11
0000	Blank Space	12 +	NP, NS	NP, NS
0001	1 4 8 ;	12 4 8)	11 4 8 *	0 4 8 (
0010	11 –	12 3 8 .	11 3 8 $	0 3 8 , (Comma)
0011	0 0	12 0 NP	11 0 NP	4 8 ' (Apostrophe)
0100	1 1	12 1 A	11 1 J	0 1 /
0101	2 2	12 2 B	11 2 K	0 2 S
0110	3 3	12 3 C	11 3 L	0 3 T
0111	4 4	12 4 D	11 4 M	0 4 U
1000	5 5	12 5 E	11 5 N	0 5 V
1001	6 6	12 6 F	11 6 O	0 6 W
1010	7 7	12 7 G	11 7 P	0 7 X
1011	8 8	12 8 H	11 8 Q	0 8 Y
1100	9 9	12 9 I	11 9 R	0 9 Z
1101	4 6 8 :	3 8 =	NP, NS	NP, NS
1110	4 5 8 <	NP, NS	11 5 8 NP	0 5 8 NP
1111	3 5 8 >	NP, NS	11 3 5 8 NP, NS	0 3 5 8 NP, NS

Table 4-2. 90-Column Remington Rand Card Code, High-Speed Reader

The upper entry represents the card punching positions.
The lower entry represents the corresponding High-Speed Printer character.
NP indicates a code which is not printed by the High-Speed Printer.

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

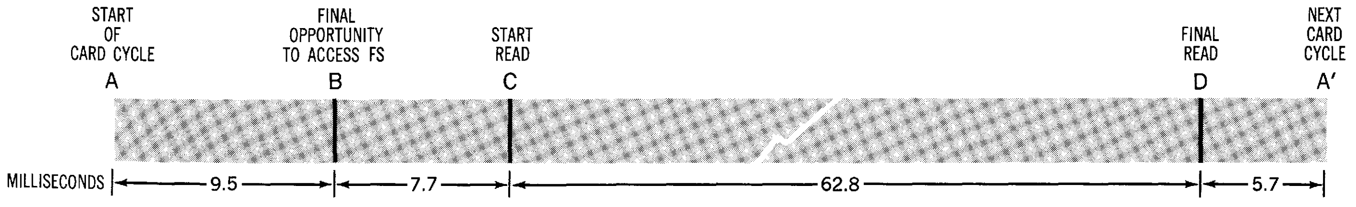


Figure 4-5. Card-Cycle Timing, High-Speed Reader

program interrupt occurs. A program interrupt also occurs if there is an error associated with the accessing of an FS; such an error would set the fault indicator. If input-output interrupt is specified and an error occurs when the FS is read from memory (FS-call error), both the initiation and the fault indicators are set.

The synchronizer tests the standby indicator at point *A*; if the indicator is not set, the synchronizer continues testing until point *B*, which is the last point at which an FS can be accessed until the next card cycle.

At point *C* the leading row of each card at the read stations is sensed; sensing of all other rows follows in succession until the final row is sensed at point *D*. Between points *D* and *A′* the synchronizer determines whether or not the correct number of memory accesses has been made for the card at the second read station, and whether or not the hole counts are equal. If no error is found, the synchronizer is alerted for the next FS; otherwise, interrupt takes place at point *A′* and the data-error indicator is set, unless an operator oversight or fault occurred at an earlier point in the cycle. More time than an entire card cycle requires (about 85 milliseconds) is available during which the program can direct a card with an error to an output stacker.

If bit 16 of the FS is a 0, input-output interrupt caused by initiation of the FS cannot occur. If desired, all input-output interrupts may be prevented by using the PIO instruction.

The following list is a summary of specifications for the High-Speed Reader:

PROGRAM-CONTROLLED FUNCTIONS:

Card feeding, stacker selection, program interrupt, memory-address selection, and automatic translation from card code to UNIVAC III character code.

CARD-FEED RATE:

700 cards per minute.

After the picker knife places a card in the card-feed path, the card moves through the reader continuously until it is deposited in one of the output stackers.

INPUT-MAGAZINE CAPACITY:

2000 cards

OUTPUT-STACKER CAPACITY:

1000 cards in each of three program-selectable output stackers

SENSING STATIONS:

Two—one for checking, and one for data transfers to memory

CHECKING:

The following checks are made on each card that is read:
- Hole count;
- Count of number of memory accesses; and
- Modulo-3 check on data transfers.

FUNCTION SPECIFICATIONS

To perform a programmed card operation, the reader requires an initiate-input-output-function (IOF) instruction and a function specification (FS).

The channel-designation bits of the IOF must specify the general-purpose channel assigned to the High-Speed Reader. Details of the reader operation are specified in the FS, which has the following format:

Binary 0's	Function Code	I	L-Address
25 21	20 17	16	15 1

Bits 21 through 25 . . . Must be binary 0's

Function Code:

Bits 20, 19 . . . Control stacker selection of card at station 4—as shown in figure 4-1, select stacker 0, 1, or 2 as follows:

Bit: *20*	*19*	*Stacker Selected*
0	1	1
1	0	2
0	0	0
1	1	2

Bit 18 . . . . . . Controls translation—if it is a 1-bit, the card image at read station 2 is translated from 80- or 90-column card code to UNIVAC III character code. If it is a 0-bit, the card is read as an untranslated image.

Bit 17 . . . . . . Controls card feeding—if it is a 1-bit, a card is fed; if it is a 0-bit, no card is fed.

I . . . . Controls interrupt—if it is a 1-bit, a program interrupt occurs when the FS is transferred to the synchronizer; otherwise, this interrupt will not occur.

L-Address . . . . The initial memory location of the first data word to be transferred. Bits 1 through 6 all must be 0's, so that the first word is transferred to a memory location which is a multiple of 64.

If instructions are not given for cards already committed to the reader, their images are transferred to the area of memory, and in the format (translated or untranslated) which were specified by the previous FS executed. If no FS is given to the reader synchronizer when there is a card at station 4, the card is deposited in stacker 0.

The individual function codes with their mnemonics are described under the headings which follow. In each operation, the memory address for 80-column cards is shown; for 90-column cards, all data will occupy memory addresses L through L + 23.

FEED, TRANSLATE, STACKER 0—FCT

Operation: Feed 1 card, image with translation → L . . . L + 19, stacker 0

Function Code: 0011

Description: Feed one card from the input magazine; translate the card image at read station 2 into the UNIVAC III character code and transfer the translated image to memory, and place the card at station 4 in stacker 0.

FEED, TRANSLATE, STACKER 1—FCTS1

Operation: Feed 1 card, image with translation → L . . . L + 19, select stacker 1

Function Code: 0111

Description: Feed one card from the input magazine; translate the card image at read station 2 into the UNIVAC III character code and transfer the translated image to memory; and place the card at station 4 in stacker 1.

FEED, TRANSLATE, STACKER 2—FCTS2

Operation: Feed 1 card, image with translation → L . . . L + 19, select stacker 2

Function Code: 1011

Description: Feed one card from the input magazine; translate the card image at read station 2 into the UNIVAC III character code and transfer the translated image to memory; and place the card at station 4 in stacker 2.

FEED WITHOUT TRANSLATION, STACKER 0—FC

Operation: Feed 1 card, image without translation → L . . . L + 39, stacker 0

Function Code: 0001

Description: Feed one card from the input magazine; transfer the card image at read station 2 to memory without translation; and place the card at station 4 in stacker 0.

FEED WITHOUT TRANSLATION, STACKER 1—FCS1

Operation: Feed 1 card, image without translation → L . . . L + 39, select stacker 1

Function Code: 0101

Description: Feed one card from the input magazine; transfer the card image at read station 2 to memory without translation; and place the card at station 4 in stacker 1.

FEED WITHOUT TRANSLATION, STACKER 2—FCS2

Operation: Feed 1 card, image without translation → L . . . L + 39, select stacker 2

Function Code: 1001

Description: Feed one card from the input magazine; transfer the card image at read station 2 to memory without translation; and place the card at station 4 in stacker 2.

NO FEED, TRANSLATION, STACKER 0—CT

Operation: Image with translation → L . . . L + 19, stacker 0

Function Code: 0010

Description: Translate the card image at read station 2 into the UNIVAC III character code and transfer it to memory; and place the card at station 4 in stacker 0. A card is not introduced into the feed path from the input magazine.

NO FEED, TRANSLATION, STACKER 1—CTS1

Operation: Image with translation → L . . . L + 19, select stacker 1

Function Code: 0110

Description: Translate the card image at read station 2 into the UNIVAC III character code and transfer it to memory; and place the card at station 4 in stacker 1. A card is not introduced into the feed path from the input magazine.

NO FEED, TRANSLATION, STACKER 2—CTS2

Operation: Image with translation → L . . . L + 19, select stacker 2

Function Code: 1010

Description: Translate the card image at read station 2 into the UNIVAC III character code and transfer it to memory; and place the card at station 4 in stacker 2. A card is not introduced into the feed path from the input magazine.

NO FEED, WITHOUT TRANSLATION, STACKER 0—CAD

Operation: Image without translation → L . . . L + 39, stacker 0

Function Code: 0000

Description: Transfer the card image at read station 2 to memory without translation; and place the card at station 4 in stacker 0. A card is not introduced into the feed path from the input magazine.

NO FEED, WITHOUT TRANSLATION, STACKER 1—CS1

Operation: Image without translation → L . . . L + 39, select stacker 1

Function Code: 0100

Description: Transfer the card image at read station 2 to memory without translation; and place the card at station 4 in stacker 1. A card is not introduced into the feed path from the input magazine.

NO FEED, WITHOUT TRANSLATION, STACKER 2—CS2

Operation: Image without translation → L . . . L + 39, select stacker 2

Function Code: 1000

Description: Transfer the card image at read station 2 to memory without translation; and place the card at station 4 in stacker 2. A card is not introduced into the feed path from the input magazine.

Function codes for reader operations, together with their mnemonic codes, are summarized in table 4-3.

Table 4-3. Summary of Function Codes for High-Speed Reader

Function	Function Code: Bit Positions				Mnemonic Code
	20	19	18	17	
Feed card and translate	0	0	1	1	FCT
Feed card, translate and select stacker 1	0	1	1	1	FCTS1
Feed card, translate and select stacker 2	1	0	1	1	FCTS2
Feed card	0	0	0	1	FC
Feed card and select stacker 1	0	1	0	1	FCS1
Feed card and select stacker 2	1	0	0	1	FCS2
Translate	0	0	1	0	CT
Translate and select stacker 1	0	1	1	0	CTS1
Translate and select stacker 2	1	0	1	0	CTS2
Card image transferred to memory without translation	0	0	0	0	CAD
Select stacker 1	0	1	0	0	CS1
Select stacker 2	1	0	0	0	CS2

■ INTERRUPT INDICATORS

The program-testable interrupt indicators for the High-Speed Reader and their mnemonic-code designations are as follows:

Indicator	*Mnemonic Code*
Initiation	2
Data error	5
Fault	7
Operator oversight	6

The specific conditions which cause these indicators to be set are summarized in table 4-4.

INITIATION

The initiation indicator is set if input-output interrupt is specified when the High-Speed Reader synchronizer receives the FS from the reader standby location.

DATA ERROR

When the data-error indicator is set, an error has been detected associated with the card that has just been read at the second read station. Interrupt occurs at the beginning of the next card cycle. Cards with errors are delivered to the stacker specified by the reader routine.

When an interrupt caused by a data error occurs, the FS in the standby location is transferred to the synchronizer and executed. The synchronizer is then inhibited from accessing memory until the data-error indicator is reset.

FAULT

When the unit is on-line, the fault indicator is set by conditions which require corrective action by the operator or maintenance personnel. It is set only if the standby-location interlock indicator is set or if cards are present in the card-feed path. A fault condition may occur at any time during the card cycle. When a fault is detected, the synchronizer does not access memory until the fault is corrected, the ABNORMAL CLEAR button-light on the reader control panel (table 4-5) is pressed, and the fault indicator is reset. When an interrupt occurs, cards in the stacker section of the card-feed path are directed into the stacker selected by the program. The cards at stations 1 through 4 at the end of the execution of the FS are automatically directed to stacker 0. If a data error or operator oversight occurs during the same card cycle as a fault, only the fault indicator is set.

OPERATOR OVERSIGHT

When the unit is on-line, the operator-oversight indicator is set as a result of conditions which require minor operator intervention. It is set only if the standby-location interlock indicator is set or if cards

Table 4-4. Abnormal-Interrupt Conditions, High-Speed Reader

Condition	Description
The following conditions set the data-error indicator.	
Check-read error	The number of holes counted for a given card as it passed read station 1 did not agree with the hole count for read station 2. The READ ERROR button-light on the reader control panel will be lit.
Memory-address error	An addressing error has occurred while data was being read from memory.
Number-of-memory-accesses error	The correct number of memory accesses needed to read a card image into memory has not been granted to the synchronizer (correct number for the 80-column reader is 480; 288 for the 90-column reader). The READ ERROR button-light on the reader control panel will be lit.
Synchronizer modulo-3 error	A modulo-3 error was detected while data was being transferred from the Central Processor to the synchronizer.
Memory modulo-3 error	A modulo-3 error was detected when memory was accessed to transfer a data word.
The following conditions set the fault indicator.	
Feed jam	The card at read station 2 has jammed or is out of phase with the reader card cycle. The drive motor is turned off, and the FEED JAM light on the reader control panel will be lit.
Stacker jam	A card has jammed in the stacker section of the feed path. The drive motor is turned off, and the STACK JAM light on the reader control panel will be lit.
FS memory-address error	A memory-address error occurred while attempting to access an FS from memory.
FS modulo-3 check	An error was detected during the modulo-3 check while an FS was being read from memory.
Logic check	A malfunction was detected as a result of one of the logic checks which the reader makes on its own operation.
The following conditions set the operator-oversight indicator.	
Door interlock	A door or cover in the reader is not properly positioned. The INTERLOCK and the ABNORMAL CLEAR lights on the reader control panel will be lit.
Full stacker	An output stacker is full. STACK FULL light on reader control panel will be lit.
Empty input magazine	No card is at read station 1 at the completion of an FS calling for card feed. The MISFEED light on the reader control panel will be lit.
Motor off	The MOTOR ON button on the reader control panel has been pressed while the reader was on-line and operating (either standby-location interlock indicator is set or cards are in card-feed path). The ABNORMAL CLEAR button-light on the reader control panel will be lit.
Off-line	The standby-location interlock indicator has been set by an IOF instruction when the reader was off-line.

are present in the card-feed path. The reader synchronizer cannot access memory until the condition is corrected, the ABNORMAL CLEAR buttonlight on the reader control panel is pressed, and the operator-oversight indicator is reset. When the interrupt occurs, cards in the stacker section are delivered to the stacker selected by the program. The cards at stations 1 through 4 at the end of the execution of the FS are automatically directed to stacker 0.

If a data error occurs during the same card cycle as an operator oversight, only the operator-oversight indicator is set.

If data error, fault, and operator oversight occur during the same card cycle, only the fault indicator is set.

■ CONTROL FEATURES

Each High-Speed Reader in the UNIVAC III System has a control panel which contains two rows of controls as shown in figure 4-6. The upper row contains lights which indicate abnormal conditions in the reader; the lower row contains buttons which are used to control reader operations. The function of each button and light is described in table 4-5.

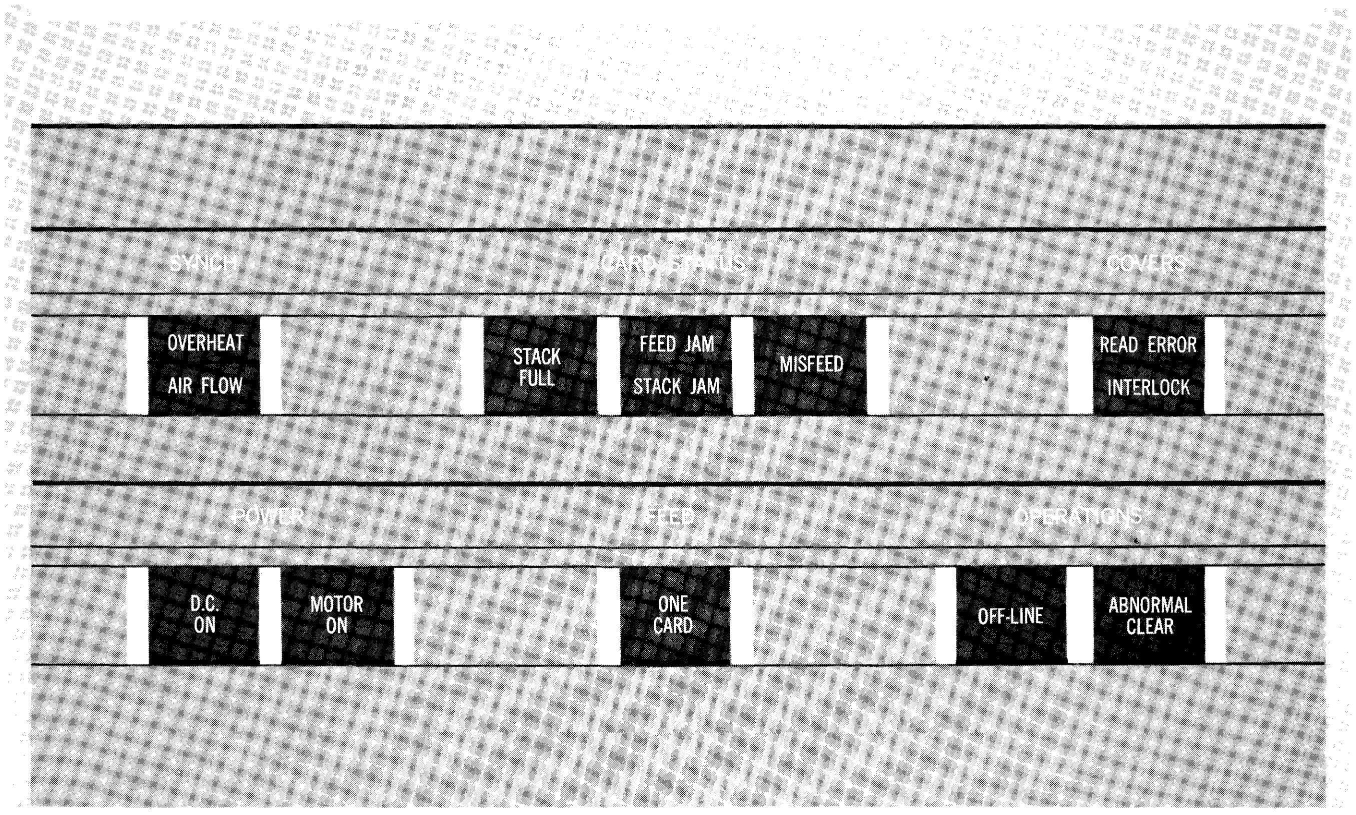


Figure 4-6. Control Panel, High-Speed Reader

Table 4-5. Function of Lights and Button-Lights on High-Speed Reader Control Panel

LIGHTS

Panel/Light Marking	Conditions Under Which Lighted
SYNCH	
OVERHEAT	Upper half: Air temperature in synchronizer is above normal level; d-c power is removed from reader.
AIR FLOW	Lower half: Insufficient cooling air flowing through reader; d-c power is removed from reader.
CARD STATUS	When any CARD STATUS or INTERLOCK light goes on, ABNORMAL CLEAR light on reader control panel, and upper half of High-Speed Reader light on monitor panel of operator's console also go on. Before reader is accessible by the program and its operation can be resumed, abnormal condition must be corrected, ABNORMAL CLEAR button pressed, and appropriate program-testable indicator reset.
STACK FULL	An output stacker is full. Operator-oversight indicator is set.
FEED JAM	Upper half: A card has jammed in the feed mechanism. Drive motor is turned off and fault indicator is set.
STACK JAM	Lower half: A card has jammed in stacker mechanism. Drive motor is turned off and fault indicator is set.
MISFEED	Input magazine is empty or a card has been misfed. Operator-oversight indicator is set.
COVERS	
READ ERROR*	Upper half: Hole-count or number-of-memory-accesses error has occurred. Data-error indicator is set. Light turns off when pressed.
INTERLOCK	Lower half: A door or cover in reader is improperly positioned. Operator-oversight indicator is set.

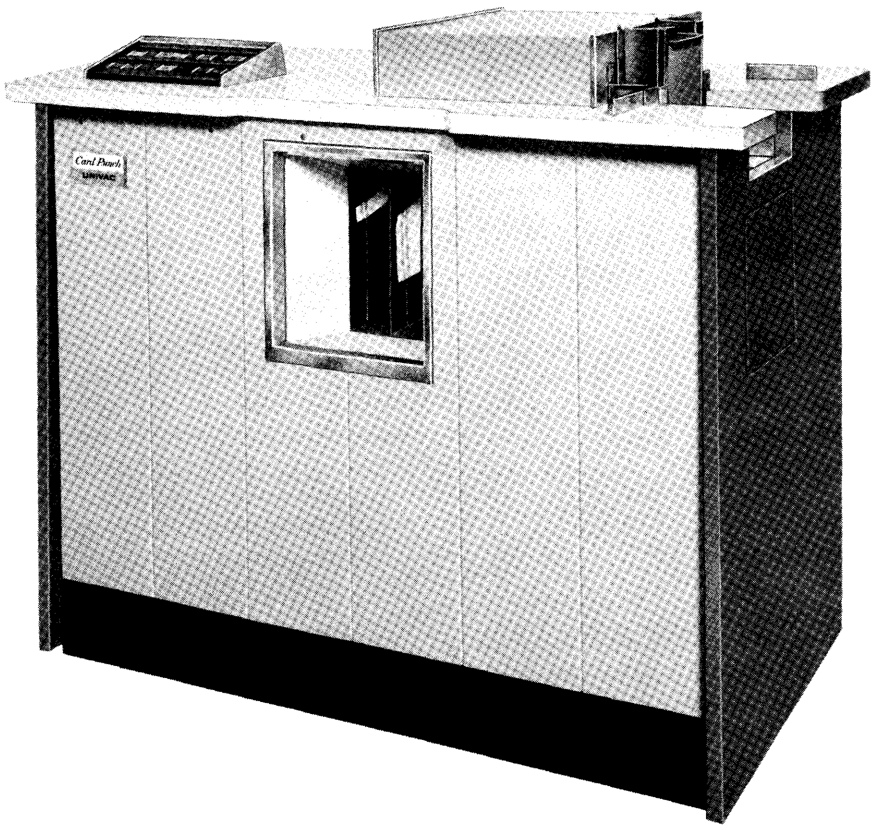
BUTTON-LIGHTS

Panel/Button Marking	Function
POWER	
DC ON	Applies d-c power to reader when power is off; turns light on. Removes d-c power from reader when power is on; turns off light of button.
MOTOR ON	Transfers control of reader drive motor to or from synchronizer when punch is on-line and operating; light goes on when drive motor is under synchronizer control.
FEED	
ONE CARD†	Feeds one card through reader, transfers a card image without translation to memory, and deposits the card in stacker 0. Pressing button clears last six bits of memory-address counter (MAC) to zero. Receiving area of memory is determined by contents of MAC after it is cleared. (Pressing CLEAR button on operator's console clears all MAC's to zero.)
OPERATIONS	
OFF-LINE	Places unit off-line and is lit if it is on-line. In this condition, FS's will not be accessed from their standby location, but synchronizer will complete any FS already in progress. Causes interrupt with operator-oversight indicator set if standby-location interlock indicator is set. The cards from stations 1, 2, 3 and 4 after completing the FS will be delivered to stacker 0. Places unit on-line if it is off-line. FS in standby location will be accessed and executed, provided the standby-location interlock indicator is set. Used primarily for maintenance.
ABNORMAL CLEAR	Indicates an abnormal condition in the reader which usually can be identified by examining lights on reader control panel. Upper half of High-Speed Reader light on monitor panel of operator's console also lights. When abnormal condition is corrected, button must be pressed before operation of reader can be resumed. When pressed, operation of reader can be resumed; turns off button-light, and light on monitor panel of operator's console.

*Also has button function.
†Button only.

SECTION V

CARD PUNCH



The Card-Punch Unit is available as either an 80-column model or a 90-column model. Both may be included in one UNIVAC III System. The synchronizer and power supply for the punch are housed within the punch cabinet. Cards are punched at the rate of 300 cards per minute with automatic checking of all punching. The 80-column punch translates the UNIVAC III character code into the standard Hollerith card code before punching; it can also punch any other card code. The 90-column punch translates the UNIVAC III character code into Remington Rand 90-column card code before punching; it can also punch any other card code.

The punch has two program-selectable output stackers, each with a capacity of 1000 cards.

As is the case with the other peripheral units, the automatic program-interrupt feature may be used with the punch. When this feature is employed, the punch synchronizer sends a signal to the Central Processor when a function specification (FS) has been initiated. This signal is used to alert the central-processor program that the standby location for the punch is ready to receive another FS. Thus, the operation of the punch is not delayed and it can function at its full capacity of 300 cards per minute.

The interrupt feature also is used to alert the Central Processor if an abnormal condition exists in the punch unit.

The Card-Punch Unit receives data and FS's from the Central Processor through one of the eight general-purpose channels.

■ OPERATING CHARACTERISTICS

The card-feed path of the punch is shown in figure 5-1. It includes a card-input magazine, two wait stations, punching dies, a post-punch station, check-read brushes, a stacker-selection station, and two output stackers. Card-feed rollers ahead of the stacker-selection station are clutched; those beyond this station rotate continuously. Thus, once a card reaches the stacker-selection station, it is transported continuously to one of the two output stackers.

OPERATION

The cards to be punched are moved into the card path by the picker knife, which is program-controlled. Cards are delayed at each wait station before being passed to the punching dies, where the cards are punched one row at a time. The card then is deposited in the post-punch station. The cards are passed from the post-punch station past the check-read brushes. At this point, the synchronizer checks the number of holes punched against the number of holes sensed by the check-read brushes. The card is then delivered to the continuously moving rollers of the stacker-selection station, where it is committed to one of the output stackers as dictated by the program.

INPUT MAGAZINE

The input magazine, which has a capacity of 1000 cards, holds the stack of blank cards to be punched. The picker knife, when activated by a feed order, moves one card from the input magazine to wait station 1. An empty input magazine causes the unit to stop, lights the EMPTY INPUT light on the punch control panel, and sets the program-testable operator-oversight indicator.

Eighty-column blank cards are placed in the input magazine face down, with the 9-edge leading; 90-column cards are placed face up with the lower field leading.

OUTPUT STACKERS

Each output stacker holds 1000 cards. When a stacker is full, the STACK FULL light on the punch control panel lights, the program-testable operator-oversight indicator is set, and an automatic program interrupt occurs. All cards enter stacker 0 unless the program specifies stacker 1.

CARD IMAGE

Cards may be punched either with or without automatic translation from UNIVAC III character code to the card code. Figures 5-2 and 5-3 show the data flow for 80-column cards, and figure 5-4 shows the data flow for 90-column cards. Cards are punched at the same rate whether translation is specified or not. The total time needed to access memory is 0.96 milliseconds for 80-column cards and 0.58 milliseconds for 90-column cards.

Twenty UNIVAC III alphanumeric words in memory can be automatically translated so that they are represented by an 80-column Hollerith-code card, as indicated in figure 5-2. When the unit is punching without translation, 40 alphanumeric words occupy one 80-column card. The first untranslated 24-bit word is punched into a card field in the upper left portion of the card, four columns wide by six rows deep, as shown in figure 5-3. Because cards are read without translation, non-Hollerith codes may be used. When the unit is punching without translation, a 1-bit in memory corresponds to a punch in the card. Sign bits are not punched.

Twenty-four UNIVAC III alphanumeric words can be automatically translated so that they are represented by one 90-column-code card, as shown in figure 5-4. The three least significant characters of the 12th and 24th words in memory are not punched. The 90-column-card fields remain the same whether data is punched with or without translation. Sign bits are not punched.

Hollerith and Remington Rand 90-column card codes are given in tables 5-1 and 5-2, respectively.

CARD CHECKING

After the synchronizer has received all of the data from a card, the number of memory accesses is checked to verify that an entire card image has been transferred.

The hole count determined when the card is check-read is compared with the hole count which should have been punched. A modulo-3 check also is made on each word as it is transferred from memory. If any of these checks detects an error, a program-testable indicator is set, and an input-output interrupt occurs.

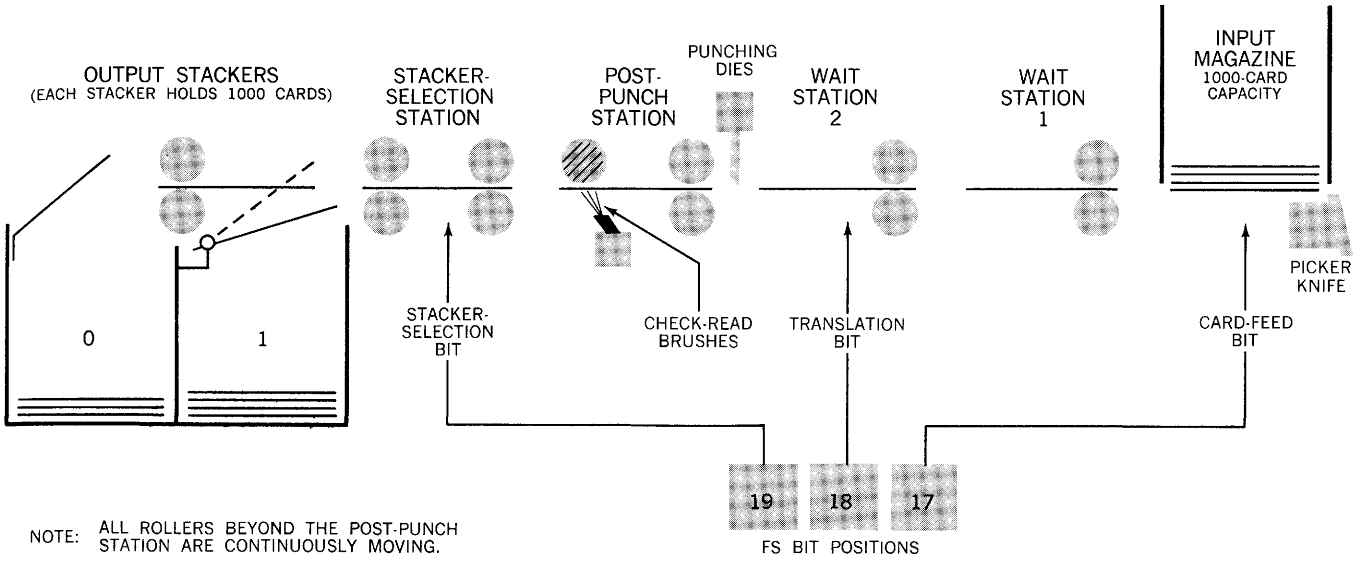


Figure 5-1. Card-Feed Path, Card-Punch Unit

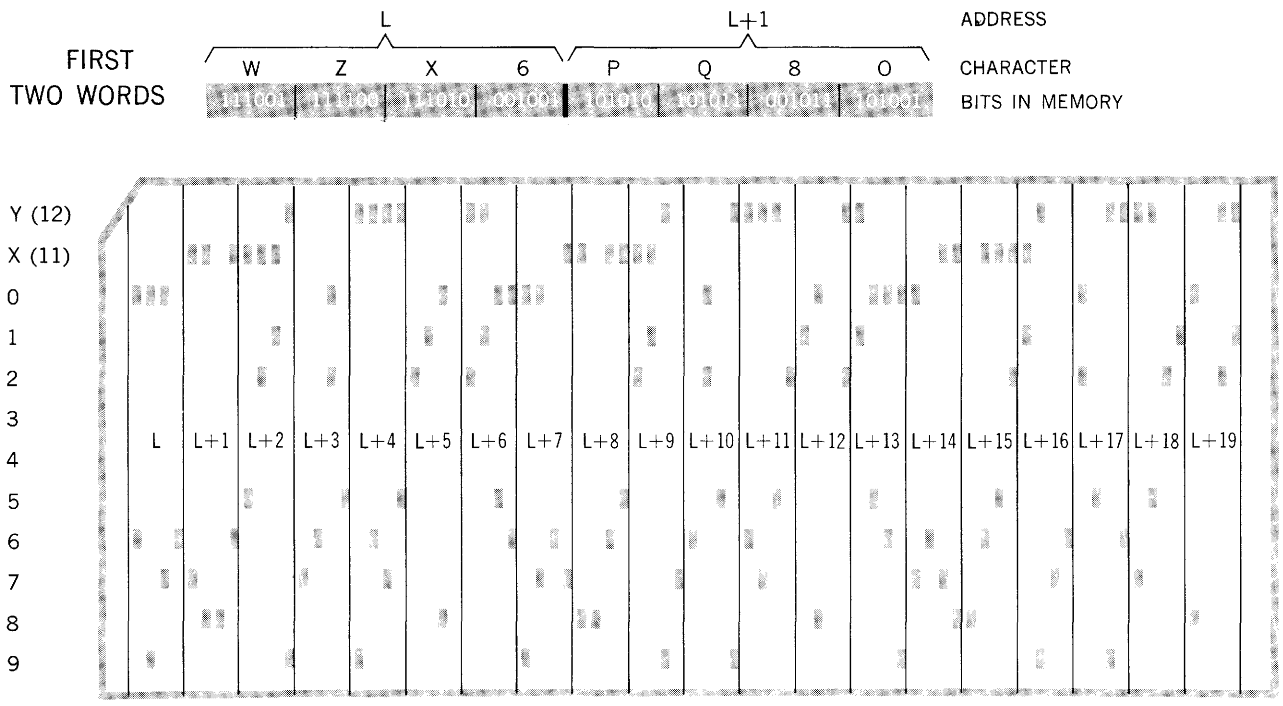


Figure 5-2. Data Transfer from Memory to Punch, With Translation, 80-Column Card

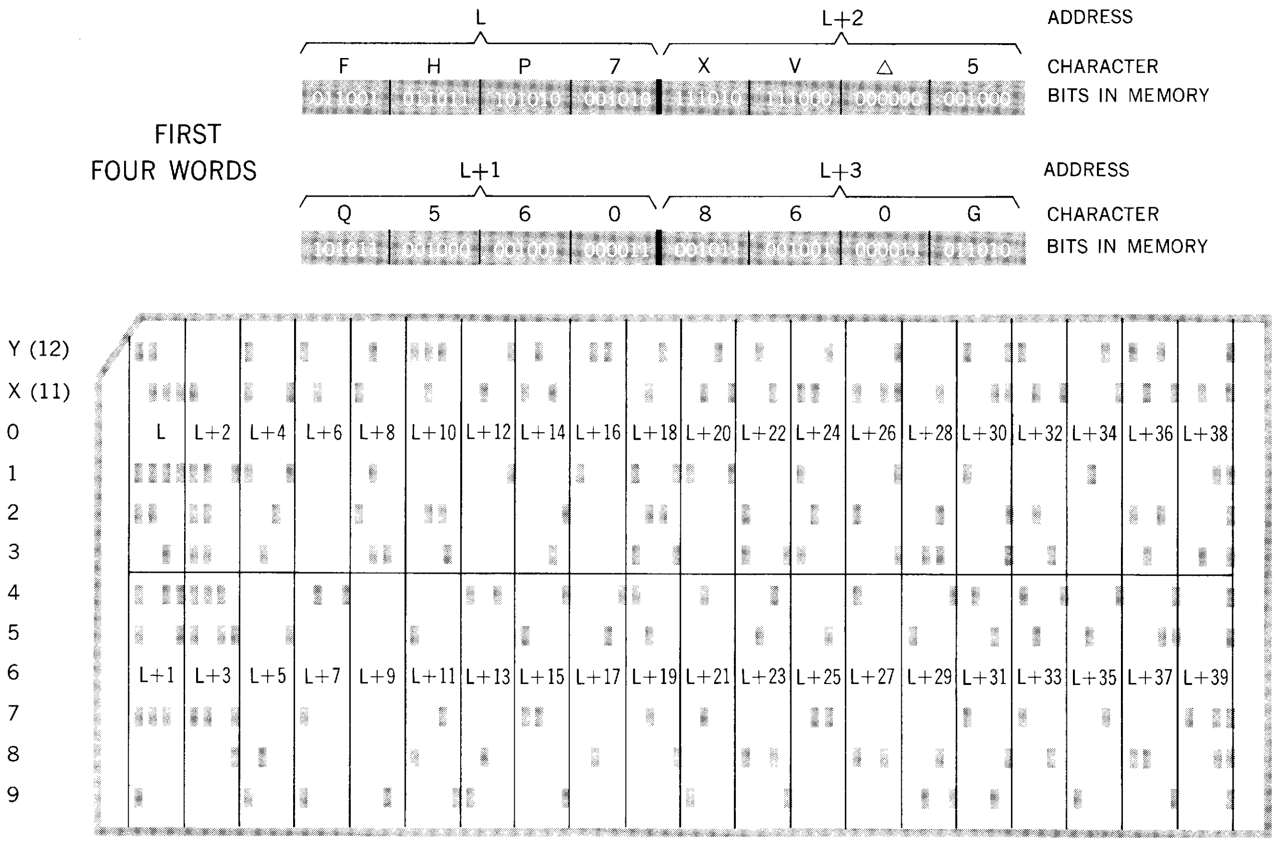


Figure 5-3. Data Transfer from Memory to Punch, Without Translation, 80-Column Card

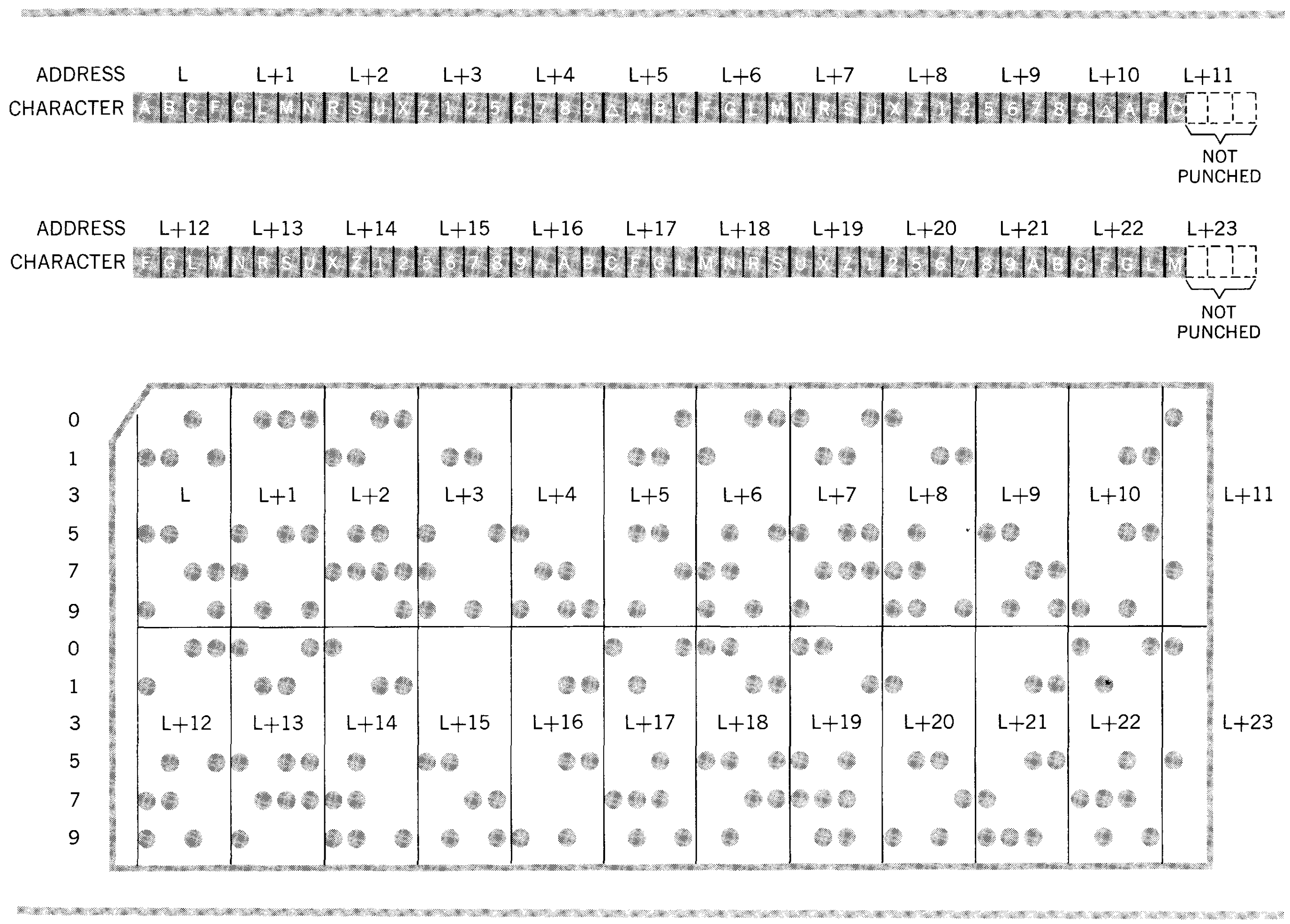


Figure 5-4. Data Transfer from Memory to Punch, 90-Column Card

Table 5-1. Hollerith Code, Card-Punch Unit

The upper entry represents the card punching positions.
The lower entry represents the corresponding High-Speed Printer character.
NP indicates a code which is not printed by the High-Speed Printer.
NS indicates a nonstandard punch code.

Numeric Bits	Zone Bits			
	00	01	10	11
	Blank	12	11	0
0000	Space	+	NP, NS	NP, NS
	1 4 8	12 4 8	11 4 8	0 4 8
0001	;	)	*	(
	11	12 3 8	11 3 8	0 3 8
0010	— (Minus)	.	$	, (Comma)
	0	12 0	11 0	4 8
0011	0	NP	NP	' (Apostrophe)
	1	12 1	11 1	0 1
0100	1	A	J	/
	2	12 2	11 2	0 2
0101	2	B	K	S
	3	12 3	11 3	0 3
0110	3	C	L	T
	4	12 4	11 4	0 4
0111	4	D	M	U
	5	12 5	11 5	0 5
1000	5	E	N	V
	6	12 6	11 6	0 6
1001	6	F	O	W
	7	12 7	11 7	0 7
1010	7	G	P	X
	8	12 8	11 8	0 8
1011	8	H	Q	Y
	9	12 9	11 9	0 9
1100	9	I	R	Z
	4 6 8	3 8	3 4 6 8	3 8
1101	:	=	NP, NS	NP, NS
	4 5 8	5 8	11 5 8	0 5 8
1110	<	NP, NS	NP	NP
	3 5 8	3 5 8	11 3 5 8	0 3 5 8
1111	>	NP, NS	NP, NS	NP, NS

Table 5-2. 90-Column Remington Rand Card Code, Card-Punch Unit

The upper entry represents the card punching positions.
The lower entry represents the corresponding High-Speed Printer character.
NP indicates a code which is not printed by the High-Speed Printer.

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

EXECUTION CYCLE

Punch-unit operations, which occur during a card cycle, are performed under program control. A typical card cycle, which includes the setting of program-testable indicators, is illustrated in figure 5-5. Each cycle consists of the following concurrent events:

The picker knife moves the bottom card in the input magazine to wait station 1;

The card placed at wait station 1 during the previous card cycle is transported to wait station 2;

The card placed at wait station 2 during the previous card cycle is transported through the punching mechanism where it is punched one row at a time and moved to the post-punch station;

The card placed at the post-punch station during the previous card cycle is sensed by the check-read brushes; and

The card passing through the continuously moving rollers of the stacker-selection station is committed to one of the output stackers according to the program.

The card cycle is initiated by an IOF instruction, which transfers an FS to the standby location for the Card-Punch Unit, and sets the standby-location interlock indicator for the punch unit. The Central Processor continues with its next instruction. When the punch synchronizer becomes available after completing a card cycle at point *A*, it tests the standby indicator. If the indicator is found to be set, the FS in the standby location is automatically transferred to the synchronizer for execution, and the standby indicator is reset. When this occurs and input-output interrupt is specified, the initiation indicator is set and program interrupt occurs. A program interrupt also occurs if there is an error associated with the accessing of an FS; this error sets the fault indicator. If input-output interrupt is specified and an error occurs when the FS is read from memory (FS-call error), both the initiation and the fault indicators are set.

The leading row of the card from wait station 2 is punched at point *C* (figure 5-5). The leading row of the card that was punched during the previous cycle is brush-sensed at point *B*. If a data error is detected, interrupt will occur at point *D*. Therefore, the program time available to segregate the card with the error is about 16.1 milliseconds. A fault or an operator oversight can occur at any time during the card cycle.

If bit 16 of the FS is a 0, input-putput interrupt caused by initiation of the FS cannot occur. If desired, all input-output interrupts may be prevented by using the PIO instruction.

The following list is a summary of Card-Punch-Unit specifications:

PROGRAM-CONTROLLED FUNCTIONS:
Card feeding, stacker selection, program interrupt, memory-address selection, and automatic translation from UNIVAC III character code to card code

CARD-FEED RATE:
300 cards per minute

CARD MOTION:
After a card is placed in the card-feed path by the picker knife, the card is moved in fixed steps until the çontinuously moving output rollers deposit it in one of the output stackers.

INPUT-MAGAZINE CAPACITY:
1000 cards

OUTPUT-STACKER CAPACITY:
1000 cards in each of two output stackers

PUNCHING:
Row-by-row for both 80- and 90-column cards

CHECKING:
The following checks are made on each card punched:
Hole count;
Count of number of memory accesses; and
Modulo-3 check on data transfers.

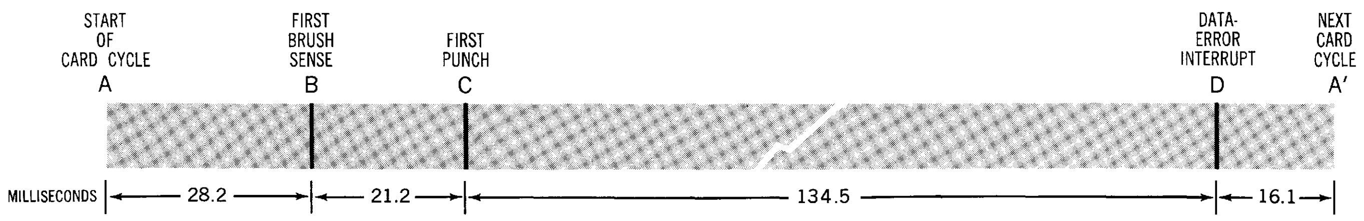


Figure 5-5. Card-Cycle Timing, Card-Punch Unit

■ FUNCTION SPECIFICATIONS

To perform a programmed card operation, the punch requires an initiate-input-output-function (IOF) instruction and a function specification (FS).

The channel-designation bits of the IOF must refer to the particular general-purpose channel assigned to the Card-Punch Unit. Details of the punch operation are specified in the FS, which has the following format:

Binary 0's	Function Code	I	L-Address
25 20	19 17	16	15 1

Bits 20 through 25 . . . Must be binary 0's

Function Code:

Bit 19	Controls stacker selection of card at stacker-selection station—selects stacker 1 if it is a 1-bit, stacker 0 if a 0-bit (see figure 5-1).
Bit 18	Controls translation—if it is a 1-bit, the card image from memory is translated and punched; if a 0-bit, the image will be punched without translation (see figure 5-1).
Bit 17	Controls card feeding—if it is a 1-bit, a card is fed, the cards already in the card-feed path advance one station, and the card moved through the punching mechanism is punched; if a 0-bit, no card is fed (see figure 5-1).
I	Controls interrupt—if it is a 1-bit, a program interrupt occurs at the start of the card cycle; otherwise, this interrupt will not occur.
L-Address	The initial memory location of the first data word to be transferred. Bits 1 through 6 all must be 0's, so that the first word transferred to the punch is taken from a memory location which is a multiple of 64.

The individual function codes with their mnemonics are described under the headings which follow. In each operation the memory address for 80-column cards is shown; for 90-column cards, all data will occupy memory addresses L through L + 23.

PUNCH CARD—PC

Operation: Feed 1 card, image without translation L. . .L + 39 → punch, stacker 0

Function Code: 001

Description: Feed one card from the input magazine and advance all cards in the card-feed path one station; punch one card without translation and move it to the post-punch station; check-read the card that was at the post-punch station; and deposit the card at the check-read station in stacker 0.

PUNCH CARD, TRANSLATE—PCT

Operation: Feed 1 card, image with translation L. . .L + 19 → punch, stacker 0

Function Code: 011

Description: Feed one card from the input magazine and advance all cards in the card-feed path one station; punch one card with translation and move it to the post-punch station; check-read the card that was at the post-punch station; and deposit the card at the stacker-selection station in stacker 0.

PUNCH CARD, SELECT—PCS

Operation: Feed 1 card, image without translation L. . .L + 39 → punch, select stacker 1

Function Code: 101

Description: Feed one card from the input magazine and advance all cards in the card-feed path one station; punch one card without translation and move it to the post-punch station; check-read the card that was at the post-punch station; and deposit the card at the stacker-selection station in stacker 1.

PUNCH CARD, TRANSLATE, SELECT—PCTS

Operation: Feed 1 card, image with translation L. . .L + 19 → punch, select stacker 1

Function Code: 111

Description: Feed one card from the input magazine and advance all cards in the card-feed path one station; punch one card with translation and move it to the post-punch station; check-read the card that was at the post-punch station; and deposit the card at the stacker-selection station in stacker 1.

CHECKED CARD, SELECT—CCS

Operation: Select stacker 1

Function Code: 100

Description: Deposit the card at the check-read station in stacker 1. Do not move any other cards. (At the end of the card cycle the stacker deflector is returned to its normal position, stacker 0.)

Function codes for punch operations together with their mnemonic codes, are summarized in table 5-3.

Table 5-3. Summary of Function Codes for Card-Punch Unit

Function	Function Code: Bit Positions			Mnemonic Code
	19	18	17	
Punch card	0	0	1	PC
Punch card, translate	0	1	1	PCT
Punch card, select	1	0	1	PCS
Punch card, translate, select	1	1	1	PCTS
Check card, select	1	0	0	CCS

INTERRUPT INDICATORS

The program-testable interrupt indicators and their mnemonic-code designations for the Card-Punch Unit are as follows:

Indicator	*Mnemonic Code*
Initiation	2
Data error	5
Fault	7
Operator oversight	6

The specific conditions which set these indicators are summarized in table 5-4.

INITIATION

The initiation indicator is set if input-output interrupt is specified when the punch synchronizer receives the FS from the punch standby location.

DATA ERROR

When the data-error indicator is set, an error has been detected on the card leaving the punching dies or on the card leaving the check-read brushes. Interrupt occurs at point *D* (figure 5-5) of the card cycle. Cards with errors are segregated automatically by the punch routine.

The synchronizer is inhibited from accessing memory while the data-error indicator is set.

FAULT

When the unit is on-line, the fault indicator is set for circumstances which require corrective action by the operator or maintenance personnel. It is set only if the standby-location interlock indicator is set or if cards are present in the card path.

The synchronizer does not access memory until the fault is corrected, the ABNORMAL CLEAR button-light on the punch control panel (table 5-5) is pressed, and the fault indicator is reset. If a data error occurs during the same card cycle as a fault, only the fault indicator is set. If an operator oversight occurs during the same cycle as a fault, the indicator associated with the condition which first occurs will be set.

OPERATOR OVERSIGHT

When the unit is on-line, the operator-oversight indicator is set as a result of conditions which require minor operator intervention. It is set only if the standby-location interlock indicator is set or if cards are present in the card path. The punch synchronizer cannot access memory until the condition is corrected, the ABNORMAL CLEAR button-light on the punch control panel is pressed, and the operator-oversight indicator is reset.

If a data error occurs during the same cycle as an operator-oversight condition, only the operator-oversight indicator is set.

If all three conditions (data error, fault, and operator oversight) occur during the same card cycle, the indicator associated with the condition which first occurs will be set.

CONTROL FEATURES

Each Card-Punch Unit in the UNIVAC III System has a control panel which contains two rows of controls as shown in figure 5-6. The upper row contains lights which indicate abnormal conditions in the punch unit; the lower row contains buttons which are used to control punch operations. The function of the buttons and lights is described in table 5-5.

Table 5-4. Abnormal-Interrupt Conditions, Card-Punch Unit

Condition	Description
The following conditions set the data-error indicator.	
Check-read error	A hole-count error was detected at the check-read brushes.
Memory-address error	An addressing error occurred while data was being read from memory.
Number-of-memory-accesses error	The correct number of memory accesses needed to transfer data from memory to the punch has not been granted. (The correct number for the 80-column punch is 240; for the 90-column punch, 144.)
Synchronizer modulo-3 error	A modulo-3 error was detected while data was being transferred from the Central Processor to the synchronizer.
Memory modulo-3 error	A modulo-3 error was detected while data was being read from memory.
Incomplete processing of memory request or memory access for a data word	An error occurred while attempting to transfer data from memory to the synchronizer.
The following conditions set the fault indicator.	
Feed jam	Any of the following conditions has occurred: A card jammed in the feed path; Card movement is not properly synchronized with the operations of the punch; or No card is at wait station 1 after an FS calling for punching has been executed. The drive motor is turned off and the FEED JAM light on the punch control panel will be lit.
Stacker jam	A card jam occurred in the stacker section of the card-feed path. The drive motor is turned off and the STACK JAM light on the punch control panel will be lit.
FS memory-address error	A memory-address error occurred while attempting to access an FS from memory.
FS modulo-3 check	An error was detected during the modulo-3 check while an FS was being read from memory.
Logic check	A malfunction was detected as a result of one of the logic checks which the punch makes on its own operation.
The following conditions set the operator-oversight indicator.	
Door interlock	A door or cover in the punch is not properly in place. The INTERLOCK light and the ABNORMAL CLEAR button-light on the punch control panel will be lit.
Full stacker	One of the output stackers is full. The STACK FULL light on the punch control panel will be lit.
Empty input magazine	The input magazine is empty. The EMPTY INPUT light on the punch control panel will be lit.
Motor off	The MOTOR ON button on the punch control panel has been pressed while the punch was on-line and operating (either standby-location interlock indicator is set or cards are in card-feed path). The ABNORMAL CLEAR button-light on the punch control panel will be lit.
Chip box	The chip box is full or improperly positioned. The CHIP BOX light on the punch control panel will be lit.
Off line	The standby-location interlock indicator has been set by an IOF instruction when the punch was off-line.

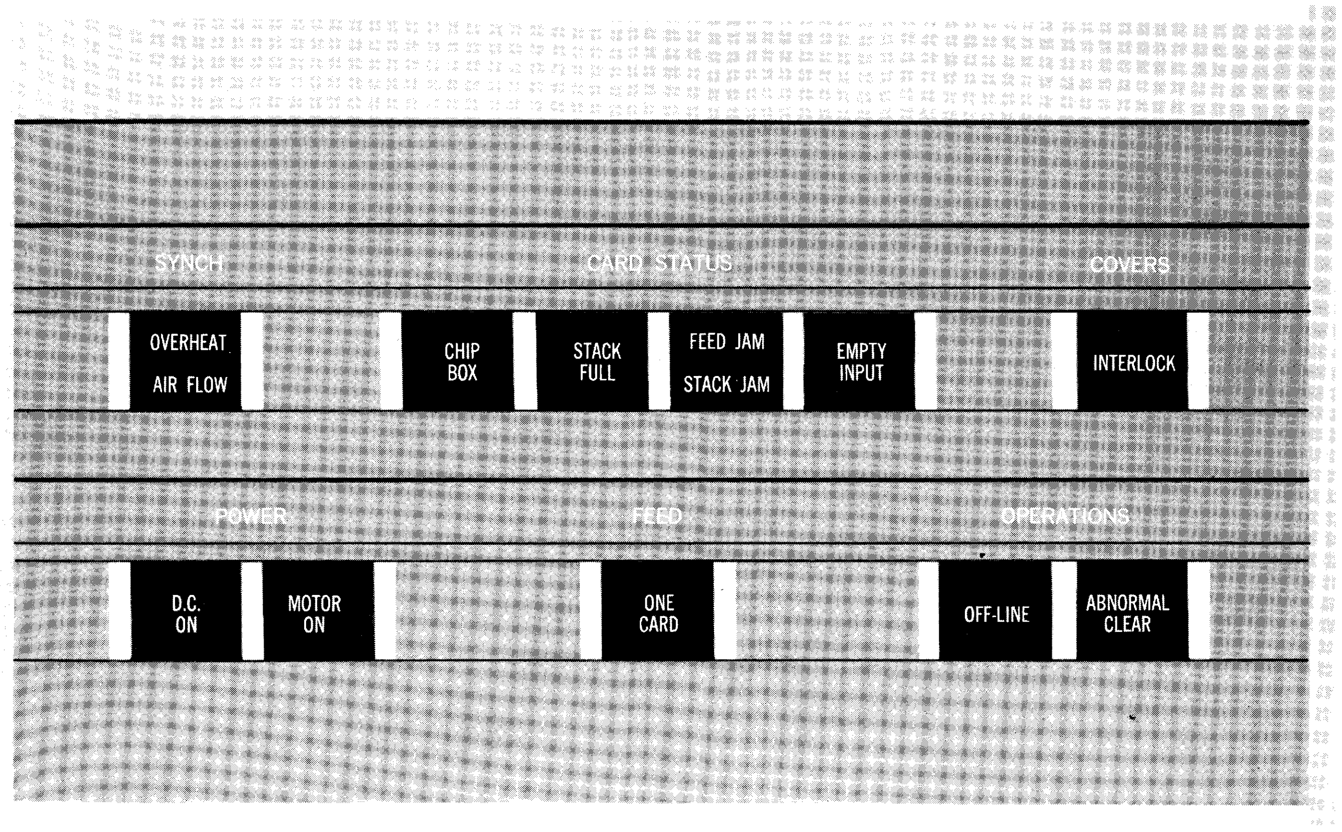


Figure 5-6. Control Panel, Card-Punch Unit

Table 5-5. Function of Lights and Button-Lights on Card-Punch Unit Control Panel

LIGHTS

Panel/Light Marking	Conditions Under Which Lighted
SYNCH	
OVERHEAT	Upper half: Air temperature in synchronizer is above normal level; d-c power is removed from punch.
AIR FLOW	Lower half: Insufficient cooling air flowing through punch; d-c power is removed from punch unit.
CARD STATUS	When any CARD STATUS or INTERLOCK light goes on, ABNORMAL CLEAR light on punch control panel, and upper half of Card Punch Unit light on monitor panel of operator's console also go on. Before punch unit is accessible by the program and punch operation can be resumed, abnormal condition must be corrected, ABNORMAL CLEAR button pressed, and appropriate program-testable indicator reset.
CHIP BOX	Chip box full or incorrectly positioned. Operator-oversight indicator is set.
STACK FULL	An output stacker is full. Operator-oversight indicator is set.
FEED JAM	Upper half: A card has jammed in the card-feed path, or a card has been misfed from input magazine. Drive motor is turned off and fault indicator is set.
STACK JAM	Lower half: A card has jammed in stacker section of card-feed path. Drive motor is turned off and fault indicator is set.
EMPTY INPUT	Input magazine is empty after execution of FS calling for card feed. Operator-oversight indicator is set.
COVERS INTERLOCK	A door or cover in punch is improperly positioned. Operator-oversight indicator is set.

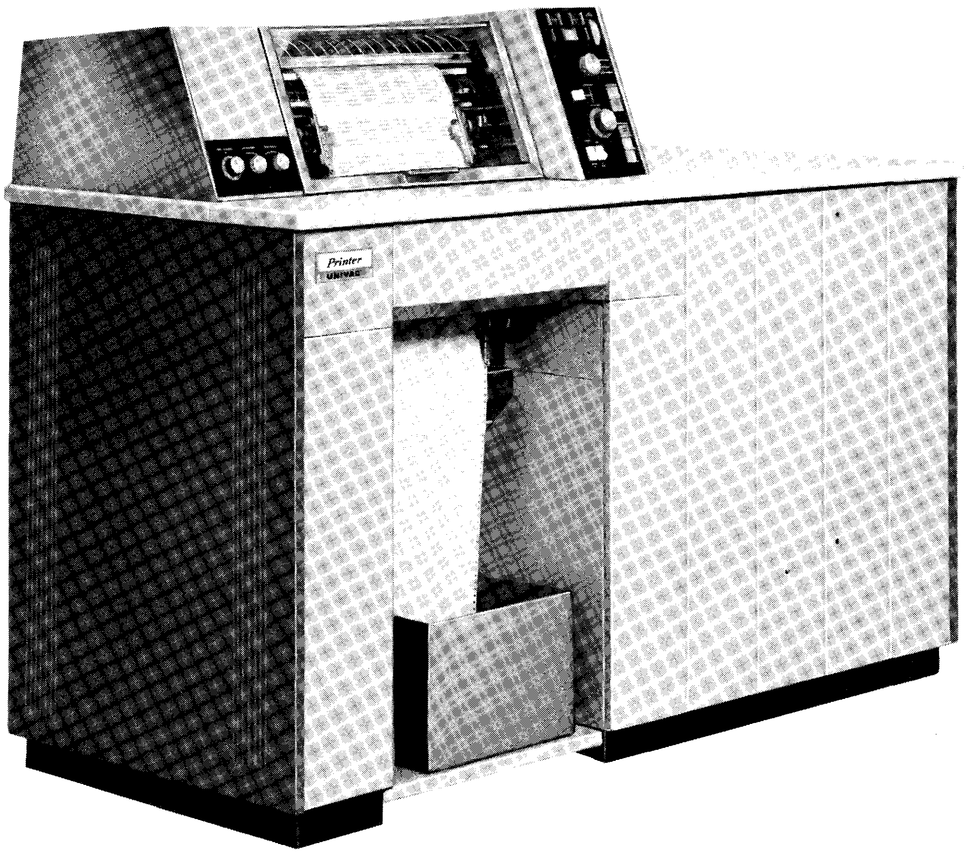
BUTTON-LIGHTS

Panel/Button Marking	Function
POWER	
DC ON	Applies d-c power to punch when power is off; turns light on. Removes d-c power from punch when power is on; turns off light of button.
MOTOR ON	Transfers control of punch drive motor to or from synchronizer when punch is on-line and operating; light goes on when drive motor is under synchronizer control.
FEED	
ONE CARD*	Feeds one card from input magazine, and advances all cards in card-feed path one station when punch is off-line. No punching occurs. Must be pressed when punch is started to load the wait and post-punch stations with cards. Has no effect if punch is on-line.
OPERATIONS	
OFF-LINE	Places unit off-line and is lit if it is on-line. In this condition, FS's will not be accessed from their standby location, but synchronizer will complete any FS already in progress. If standby-location interlock indicator is set, sets operator-oversight indicator and causes interrupt. Card in stacker-selection station after FS has been executed is delivered to stacker 0. Places unit on-line if it is off-line. FS in standby location will be accessed and executed, provided the standby-location-interlock indicator is set. Used primarily for maintenance.
ABNORMAL CLEAR	Indicates an abnormal condition in the punch which usually can be identified by examining lights on punch control panel. Upper half of Card Punch Unit light on monitor panel of operator's console also lights. When abnormal condition is corrected, button must be pressed before operation of punch can be resumed. When pressed, operation of punch can be resumed; turns off button light, and light on monitor panel of operator's console.

*Button only.

SECTION VI

HIGH-SPEED PRINTER



Under control of the central-processor program, the High-Speed Printer produces documents at the rate of 700 lines per minute for alphanumeric data and 922 lines per minute for numeric data. Lines are composed of 128 characters taken from the UNIVAC III COBOL-FORTRAN set. In addition to the original, up to five carbon copies may be produced.

Data and function specifications flow from the Central Processor to the printer synchronizer through one of the eight general-purpose channels. The synchronizer controls printing of the data according to the function specifications. To assure that the printer operates at full capacity without delaying the operation of the Central Processor, the automatic program-interrupt feature is used.

■ PHYSICAL CHARACTERISTICS

The High-Speed Printer consists of a printer cabinet and an adjoining synchronizer. The printer cabinet contains a continuously rotating type drum with 128 printing positions, 128 print hammers which correspond to the printing positions, a self-reversing ribbon-feed mechanism, and a paper-drive mechanism. The synchronizer contains the circuitry that controls data transfers, paper advance, and printing.

TYPE DRUM

Along the length of the type drum (figure 6-1) are 128 bands of printing characters. Each band contains the 51-character print set around the circumference of the drum, in the following order:

; — 0 1 2 3 4 5 6 7 8 9 : < > +) . A B C D E F G
H I = * $ J K L M N O P Q R (, ' / S T U V W X Y Z

Characters are arranged on the drum in a checkerboard pattern so that they are separated from characters on adjacent bands by approximately ⅛ inch; this space reduces the possibility of smudging by characters on a band adjacent to the one being printed.

PAPER-ADVANCE MECHANISM

Two sets of sprocketed tractors—an upper set and a lower set—advance the continuous paper through the printer under program control. While the paper is being printed, the two sets of tractors maintain paper tension.

Each of the four tractors is equipped with a tractor-locking lever. These levers are pushed in before the tractors are adjusted; after tractor adjustment, the locking levers are pulled out, thus locking the tractors to prevent any further lateral motion.

Blank or preprinted paper from 4 to 22 inches wide and up to 22 inches long between folds can be used with the printer. The original document and up to five carbon copies may be printed, using paper of between approximately 11 and 13.5 pounds in weight, up to a pack thickness of approximately 15.5 mils; this includes card stock. Vertical spacing, which may be set at the control panel by the operator, is either 6 or 8 lines per inch; horizontal spacing is 10 characters per inch. When only 2½ inches of paper remains below the print hammers, a signal from the High-Speed Printer alerts the Central Processor and automatic program interrupt occurs. The Central Processor also is alerted and paper movement stops if paper has advanced continuously for more than 1.5 seconds.

■ OPERATING CHARACTERISTICS

DATA TRANSFER

Thirty-two words from consecutive memory locations are transferred to the printer synchronizer and modulo-3 checked. They remain in the synchronizer until they are printed according to the printable character code (table 6-1); sign bits are ignored. The FS being executed controls paper advance and printing only. If editing of the final printed page is desired, it is accomplished within the 32 consecutive memory locations by the internal program before the FS for printing is given.

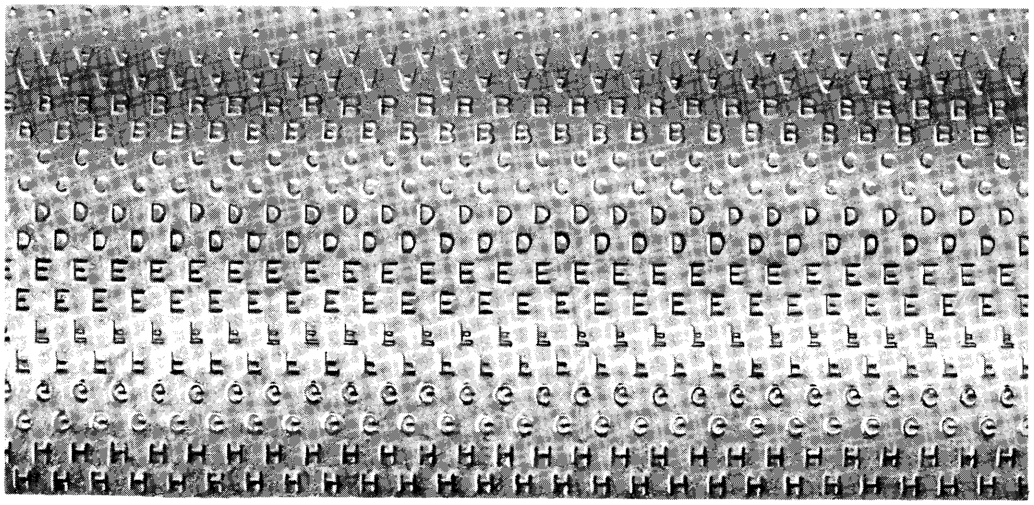


Figure 6-1. Type Drum, High-Speed Printer, Front View

Table 6-1. Printable Character Code, High-Speed Printer

NP indicates a code which is not printed by the High-Speed Printer.

Numeric Bits	Zone Bits			
	00	01	10	11
0000	Space	+	NP	NP
0001	;	)	*	(
0010	–	.	$	Comma ,
0011	0	NP	NP	' Apostrophe
0100	1	A	J	/
0101	2	B	K	S
0110	3	C	L	T
0111	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
1101	:	=	NP	NP
1110	<	NP	NP	NP
1111	>	NP	NP	NP

After the 32 words are transferred to the synchronizer, each character in the 32 words is printed in a sequence governed by the order of the characters on the type drum. The determination of which characters are to be printed next is made in the following way:

1. As the type drum rotates, the printer synchronizer keeps track of which row of characters is in printing position.
2. The character in printing position is compared with the characters stored in the synchronizer.
3. When the character on the drum in printing position matches the characters in the synchronizer, the appropriate print hammers are actuated to drive the paper and ribbon against the type drum, thereby printing the desired characters onto the paper.

TIMING

Single-spaced numeric information can be printed at the rate of 922 lines per minute; single-spaced alphanumeric information can be printed at the rate of 700 lines per minute. Timing of the paper advance depends on the number of lines advanced and on whether the line spacing is 6 or 8 lines per inch. For spacing of either 6 or 8 lines per inch, 10 milliseconds is required to advance the first line of paper; each additional line requires 8.3 milliseconds if the spacing is 6 lines per inch, or 6.25 milliseconds if the spacing is 8 lines per inch. After the paper is advanced, 10 milliseconds is required to stabilize the paper before actual printing begins. When all the characters stored in the synchronizer have been printed, the line is complete and interrupt occurs if it was specified. The printed-line-per-minute rate depends on the required paper advance and on the group of characters to be printed. The method for calculating approximate print rates for continuous printing of the same set of characters is given in Appendix F.

EXECUTION CYCLE

The typical sequence of events when a printer FS is executed, including the times when the program-testable indicators are set, is described below.

An initiate-input-output-function (IOF) instruction transfers the FS to the standby location for the High-Speed Printer and sets the printer standby-location interlock indicator. The Central Processor then continues executing the next instruction. When the printer synchronizer becomes available after completing a line of print, the FS in the standby location is transferred to the synchronizer for execution and the standby indicator is reset, unless an FS error has occurred. If an error associated with the FS has occurred, the data-error and standby indicators are set.

Concurrently with the paper advance, the synchronizer is loaded with 32 words from memory. When paper movement ends, printing can begin. The printing cycle is completed as soon as all of the characters have been printed. If input-output interrupt is specified, the successful-completion indicator then is set and interrupt occurs. If a fault is detected during printing, the fault indicator is set and interrupt occurs. If a second FS is completed before the program resets the successful-completion indicator, the synchronizer will not set the indicator again, nor will it transfer an FS from memory until the program has reset it.

The following list is a summary of specifications for the High-Speed Printer:

PROGRAM-CONTROLLED FUNCTIONS:
Paper advance and printing

SPEED:
Alphanumeric Data—700 lines per minute with single spacing
Numeric Data—922 lines per minute with single spacing

CHARACTER SET:
51 printing characters—A through Z, 0 through 9, and 15 special symbols and punctuation marks:
; — : < > +) . = * $ (, ' /

CHARACTERS PER LINE:
128

SPACING:
Horizontal—10 characters per inch
Vertical—6 or 8 lines per inch (operator-controlled)

PAPER:
Continuously folded, sprocket-fed paper from 4 to 22 inches wide, with up to 22 inches between folds.

NUMBER OF COPIES:
An original and up to 5 carbons using paper between approximately 11 and 13.5 pounds in weight

RIBBON:
Nylon, self-reversing

PAPER-ADVANCE TIMING:
First Line—10 milliseconds
Succeeding Lines—8.3 milliseconds (6 lines per inch) or
6.25 milliseconds (8 lines per inch)
Paper Stabilization—10 milliseconds

CHECKING:
Paper runaway; low paper supply; modulo-3 check when either an FS or data is transferred to the synchronizer

■ FUNCTION SPECIFICATIONS

To perform a programmed printing operation, the printer requires an initiate-input-output-function (IOF) instruction and a function specification (FS).

The channel-designation bits of the IOF must refer to the general-purpose channel assigned to the High-Speed Printer. Details of the printing operation are specified in the FS, which has the following format:

0	Lines of Vertical Spacing	Function Code	I	L-Address
25	24 — 19	18 — 17	16	15 — 1

Bit 25. . . . . Must be binary 0

Lines of
Vertical Spacing. . . Represent, in binary, the number of lines of vertical form movement which will occur before printing

Function Code:
Bit 18. . . . . Controls printing—if it is a 1, printing takes place; if it is a 0, no printing will take place.
Bit 17. . . . . Must be binary 0

I. Controls program interrupt—if it is a 1, a program interrupt occurs when execution of the FS is completed; otherwise, this interrupt will not occur.

L-Address. . . . . Specifies, in binary, the starting memory location of the line to be printed.

The individual function codes for the High-Speed Printer are described under the headings which follow.

ADVANCE PAPER AND PRINT—PRT

Operation: Advance paper; print 1 line L . . . L + 31

Function Code: 10

Description: Advance paper the number of lines specified by bits 19 through 24; print the contents of 32 sequential memory locations, beginning with the memory address specified by bits 1 through 15.

PAPER ADVANCE—PAD

Operation: Advance paper

Function Code: 00

Description: Advance paper the number of lines specified by bits 19 through 24.

The function codes for the High-Speed Printer and the corresponding instruction mnemonic codes are summarized in table 6-2.

■ INTERRUPT INDICATORS

The program-testable interrupt indicators for the High-Speed Printer and their mnemonic-code designations are as follows:

Indicator	*Mnemonic Code*
Successful completion	2
Data error	5
Fault	7
Out-of-paper warning	6

The specific conditions which set these indicators are summarized in table 6-3.

Table 6-2. Summary of Function Codes, High-Speed Printer

Function	Function Code: Bit Positions		Mne-monic Code
	18	17	
Advance paper and print	1	0	PRT
Advance paper	0	0	PAD

Table 6-3. Abnormal-Interrupt Conditions, High-Speed Printer

Condition	Description
The following conditions set the data-error indicator.	
Data memory-address error	An address error occurred while accessing memory for a data word.
Data modulo-3 error	An error was detected during the modulo-3 check while a data word was in the synchronizer, was being transferred |to the synchronizer, or was being accessed from memory.
FS memory-address error	A memory-address error occurred while attempting to access an FS from memory.
FS modulo-3 error	An error was detected during the modulo-3 check while an FS was being accessed from memory or was being transferred to the synchronizer.
The following conditions set the fault indicator.	
Motor off	THE MOTOR ON button on the printer control panel has been pressed before or during the execution of an FS. The ABNORMAL CLEAR light on the printer control panel will be lit.
Carriage out	The carriage is not in the correct position for printing. The CARRIAGE OUT and ABNORMAL CLEAR lights on the printer control panel will be lit.
Ribbon out	The ribbon has been rewound by pressing the CHANGE RIBBON button on the printer control panel.
Logic check	A malfunction was detected as a result of one of the logic checks which the synchronizer makes on its own operation. The ABNORMAL CLEAR light on the printer control panel will be lit.
Paper runaway	Paper has been advancing for more than 1.5 seconds. The paper-advance mechanism will be stopped automatically. The ABNORMAL CLEAR light and the lower half of the FORMS light on the printer control panel will be lit.
D-c power fault	A d-c power failure has occurred in the printer. The ABNORMAL CLEAR light and the upper half of the DC light on the printer control panel will be lit.
Air temperature	The air temperature in the printer cabinet exceeds the normal level. The ABNORMAL CLEAR light and the upper half of the TEMP light on the printer control panel will be lit.
Door improperly closed	A door on the printer is not properly closed. The INTERLOCK and ABNORMAL CLEAR lights on the printer control panel will be lit.
The following condition sets the out-of-paper-warning indicator.	
Out-of-paper warning	Only 2½ inches of paper remains in the paper-advance mechanism below the printing position. The upper half of the FORMS light on the printer control panel will be lit.

SUCCESSFUL COMPLETION

When an FS is completed, the successful-completion indicator is set if input-output interrupt is specified and if no errors or other unusual circumstances are associated with the completion of the FS. The synchronizer never sets this indicator in combination with any other indicators, except the out-of-paper-warning indicator.

DATA ERROR

When the data-error indicator is set, an error associated with the data that is about to be printed has been detected. At the time this indicator is set, paper has advanced and part of a line may have been printed; the portion of the line that has been printed is correct. Interrupt occurs as soon as the error is detected, and the printer synchronizer cannot access memory again until the data-error indicator is reset. If an error associated with an FS occurs, both the data-error indicator and the standby-location interlock indicator are set. The synchronizer cannot access memory until the data-error indicator is reset. When an FS error is detected, the paper does not advance and no printing takes place.

FAULT

The fault indicator is set by conditions that require corrective action by the operator or maintenance personnel.

When the fault indicator is set, the ABNORMAL CLEAR button-light on the printer control panel lights; usually, another light on that panel also will light, identifying the particular fault. The synchronizer does not access memory until the condition is corrected, the ABNORMAL CLEAR button-light is pressed, and the fault indicator is reset. It is possible that when the interrupt arising from the fault condition occurs, part of an incorrect line already may have been printed. If both error and fault conditions are present, the indicator associated with the condition which occurs first will be set, and interrupt will occur.

OUT-OF-PAPER WARNING

The out-of-paper-warning indicator is set when execution of an FS has been completed and when only about 2½ inches of paper remains in the paper-advance mechanism below the line of print. If interrupt is specified, the successful-completion indicator also will be set, provided no error or abnormal condition has occurred. The synchronizer will not access memory until the out-of-paper-warning indicator is reset by the program.

CONTROL FEATURES

Each High-Speed Printer in the UNIVAC III System has a group of controls on the left side (figure 6-2) and a group on the right side (figure 6-3). The function of lever and rotary controls on both sides is described in table 6-4; the function of buttons and lights on the left side is described in table 6-5.

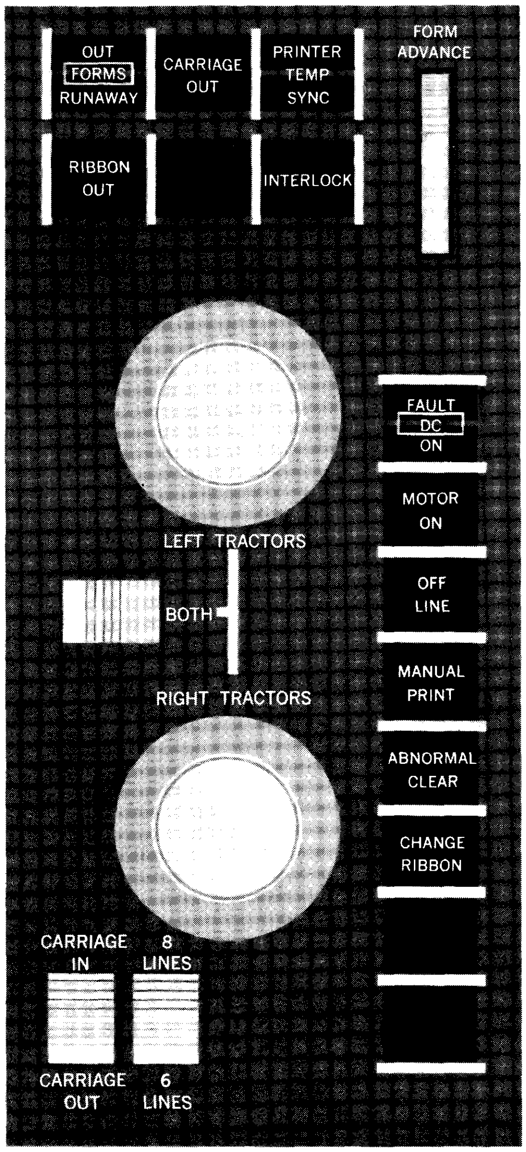


Figure 6-2. Left-Side Controls, High-Speed Printer

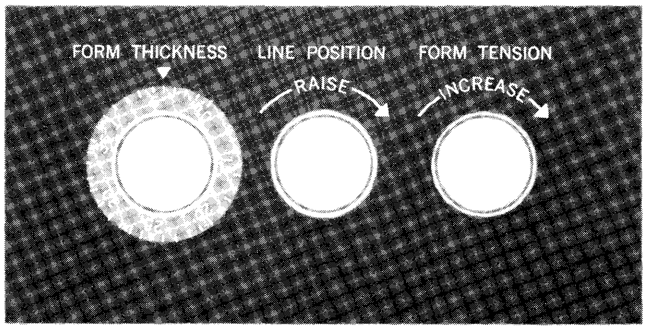


Figure 6-3. Right-Side Controls, High-Speed Printer

Table 6-4. Controls, High-Speed Printer

RIGHT SIDE

Panel Marking	Function When Operated
FORM THICKNESS	Adjusts printer mechanism for various form thicknesses.
LINE POSITION	Adjusts vertical position of form in relation to print hammers.
FORM TENSION	Adjusts vertical tension applied to form by upper and lower tractors.

LEFT SIDE

FORM ADVANCE	When printer is off-line, advances paper when control is rotated upward.
LEFT TRACTORS	Adjusts left pair of tractors so that horizontal position of paper can be changed; if BOTH switch is in BOTH position, also adjusts left pair of tractors.
BOTH	
RIGHT TRACTORS	Adjusts right pair of tractors so that horizontal position of paper can be changed; if BOTH switch is in BOTH position, also adjusts left pair of tractors.
CARRIAGE IN CARRIAGE OUT	In CARRIAGE IN position, causes carriage to move into printing position. In CARRIAGE OUT position, causes carriage to move out of printing position; when carriage is out, ABNORMAL CLEAR and CARRIAGE OUT lights on printer control panel will be lit and fault indicator will be set.
8 LINES 6 LINES	In 8 LINES position, selects vertical spacing of 8 lines per inch. In 6 LINES position, selects vertical spacing of 6 lines per inch.

Table 6-5. Button-Lights and Lights, High-Speed Printer

BUTTON-LIGHTS

Button Marking	Function
FAULT DC ON	If d-c power to the printer is on, pressing button will remove power and light will be extinguished. Upper half lights red when a power failure has occurred in the printer; fault indicator will be set. Lower half lights green when all d-c power is present.
MOTOR ON	Turns print drive motor on or off; when motor is on and running, light will be lit. If motor is turned off during execution of FS, fault indicator will be set; if no FS is being executed, fault indicator will be set the next time FS execution is attempted.
OFF LINE	Places printer on-line if it is off-line, and off-line if it is on-line. Light is lit when unit is off-line. If printer is placed off-line while it is executing an FS, this FS will be completed, but no other FS will be accessed or executed; interrupt does not occur and no error or fault indicators are set. When printer is returned on-line, printing will resume if printer standby location contains an FS.
MANUAL PRINT	When unit is off-line and button is pressed, light is lit and characters in printer synchronizer are repeatedly printed until button is pressed again. When button is pressed again, printing stops and light is extinguished.
ABNORMAL CLEAR	Lights to indicate an abnormal condition in the printer; condition usually can be identified by examining lights on printer control panel. Upper half of High-Speed Printer light on monitor panel of operator's console also lights. When abnormal condition is corrected, button must be pressed before printer operation can be resumed. When pressed, operation of printer can be resumed; button-light and light on monitor panel of operator's console are extinguished.
CHANGE RIBBON	Causes ribbon to begin rewinding; button is lit. When ribbon is rewound RIBBON OUT light is lit and fault indicator will be set; ribbon then may be changed. (The two button positions below the CHANGE RIBBON button-light are unused.)

LIGHTS

Light Marking	Condition Under Which Lighted
OUT FORMS RUNAWAY	Upper Half: Only 2½ inches of paper remains in the paper-advance mechanism below line of print, and out-of-paper-warning indicator will be set. Lower Half: Paper has advanced continuously for more than 1.5 seconds; fault indicator will be set.
CARRIAGE OUT	Carriage is not in printing position. ABNORMAL CLEAR button-light is lit and fault indicator will be set.
PRINTER TEMP SYNC	Upper Half: Air temperature in printer cabinet exceeds normal level; d-c power is removed from printer; fault indicator will be set. Lower Half: Air temperature in synchronizer cabinet exceeds normal level; d-c power is removed from printer.
RIBBON OUT	Ribbon has been rewound and new ribbon can be installed; fault indicator will be set.
INTERLOCK	A door on the printer is not closed; d-c power is removed from printer; fault indicator will be set. (The button position between the RIBBON OUT and INTERLOCK lights is unused.)

APPENDICES

APPENDIX A

MODULO-3 CHECKING

A UNIVAC III word consists of 25 information bits, including the sign bit, and 2 modulo-3 check bits. The check bits are used by the circuitry for two purposes: to check the transmission of information through the system, and to check the results of arithmetic operations.

The check bits contain the binary value necessary to make the entire word, considered as a binary number, an integral multiple of 3. Thus, if the remainder is 1 when the information bits are divided by 3, the check bits contain a value of 2. If the remainder is 2, the check bits contain a value of 1. If the remainder is 0, the check bits contain a value of 0.

For example, if the binary value 11001 (decimal 25) is divided by 3, a remainder of 1 is obtained. The value contained by the check bits would be 2 (or 10, in binary code) and the entire value would therefore be an integral multiple of 3.

An equivalent method of deriving the check bits is to assign a weight of 1 to odd bit positions containing a 1-bit, and a weight of 2 to even bit positions containing a 1-bit. The weights are added, and the sum is reverted to 0 whenever 3 is obtained. The final value produced by this modulo-3 addition is equal to the remainder that would be obtained by dividing the information bits by 3. This method of deriving the check bits is the basis of that used by the circuitry.

To check the operations of the adder, the modulo-3 check circuits generate check bits from the sum of the operands and compare them with a sum generated from the check bits of the operands. If the two values are congruent, that is, they have the same remainder, modulo-3, the addition was performed correctly. This procedure is illustrated by the following example:

Operands:	2762	*Check bits:*	1
	3430		2
Sum:	6192		
Check bits generated from sum:	0	*Sum generated from check bits:*	3 = 0, modulo-3

The check bits generated from the sum have the same remainder, modulo-3, as the sum generated from the check bits; therefore the addition was performed correctly.

EXECUTION TIME OF MULTIPLICATION

This appendix describes the method of computing the time required to perform a multiply (M) instruction, based on the value of the multiplier.

The six-digit multiplier is contained in arithmetic register 1 and remains unchanged at the end of the multiplication. Each digit in the multiplier is an excess-three binary-coded-decimal number from 0 through 9, designated n. Each digit, beginning with the least significant, has a position in the multiplier from 1 through 6, designated i subscript of the number n. For purposes of timing the multiplication, the value of a number varies according to the value of the number to its right. The final value of a number is designated n'.

The following formulas are used to compute n'; the table that follows the formulas gives the execution time, T_i, in 4-microsecond memory cycles required for each digit position, based on the value of n':

```
For $n_1$, $n'_1 = n_1$.
For $n_2$ through $n_6$,
      if      $n'_{i-1} < 5$,
    then        $n'_i = n_i$;
      if      $n'_{i-1} \geq 5$ and $n_i \neq 9$,
    then        $n'_i = n_i + 1$;
      if      $n'_{i-1} \geq 5$ and $n_i = 9$,
    then        $n'_i = 0$
     and [*]  $n'_{i+1} = n_{i+1} + 1$
```

[*] If $i = 6$, this step is ignored.

n'_i	T_i
0	2
1, 2	2
3, 4	3
5	4
6, 7	3
8, 9	2

The total execution time, T, for the multiplication is the sum of the times for the individual digit positions, plus 7:

$$T = \sum_{i=1}^{6} T_i + 7$$

Thus, for example, if the multiplier is 945270, the execution time is determined as follows:

i	n_i	n'_i	T_i
1	0	0	2
2	7	7	3
3	2	3	3
4	5	5	4
5	4	5	4
6	9	0	2
			18

$T = 18 + 7 = 25$ memory cycles

APPENDIX C

EXECUTION TIME OF DIVISION

This appendix describes the method of computing the time required to perform a divide (D) instruction, based on the value of the quotient.

The six-digit quotient appears in AR2 at the completion of the instruction. Each digit is an excess-three binary-coded-decimal number from 0 through 9, designated Q. Each digit, beginning with the least significant, has a position in the quotient from 1 through 6, designated *i* subscript of the number Q.

The time (T_i) required to produce a given digit of the quotient (Q_i) varies as the digit to its left (Q_{i+1}) is odd or even. This time, expressed in memory cycles, is found in table C-1; for the most significant digit of the quotient (Q_6), use the time given in the *Odd* column.

The total time (T) required to perform the divide instruction is the sum of the times for the individual digit positions, plus 5:

$$T = \sum_{i=1}^{6} T_i + 5$$

Thus the maximum time required to perform a divide instruction is 35 memory cycles, and the minimum time is 17 memory cycles.

For example, the time required to perform the divide instruction that results in a quotient of 806491 is determined as follows:

i	Q_i	T_i
6	8	5
5	0	5
4	6	3
3	4	4
2	9	2
1	1	2
		21

$T = 21 + 5 = 26$ memory cycles

Table C-1. Time to Produce Quotient Digit

Q_i Quotient Digit	T_i Cycles to Produce Q_i	
	Q_{i+1} Odd	Q_{i+1} Even
0	2	5
1	2	5
2	3	5
3	3	5
4	4	4
5	4	4
6	5	3
7	5	3
8	5	2
9	5	2

APPENDIX D

DECIMAL OPERATIONS ON NON-NUMERIC DATA

This appendix contains procedures and tables for determining the result of performing decimal addition, subtraction, multiplication, division, or shift operations on data that contains non-numeric digit codes. These digit codes are 0000, 0001, 0010, 1101, 1110, and 1111.

ADDITION

The procedures below, and table D-1, may be used to determine the result of a decimal addition (addition with like signs, or subtraction with unlike signs) involving non-numeric digits. Decimal subtraction involving non-numeric digits is described under the heading *Subtraction*, below.

ADDITION PROCEDURE

1. Convert binary 0's (0000) in the operand from the arithmetic registers (augend) to decimal 0's (0011). Convert binary 0's, 1's, and 2's (0000, 0001, and 0010) in the operand from memory (addend) to decimal 0's (0011).
2. Perform a binary addition on the addend and augend, adding binary 1 (0001) to the least significant digit of the sum. Note any carries from one digit position to the next.
3. To complete the addition, add an excess-three correction factor to each digit of the sum as follows:
   a. If there was a carry from the most significant bit of the digit in step 2, add 0011 to the digit, ignoring the resultant carries into the next digit position.
   b. If there was no carry from the most significant bit of the digit in step 2, add 1101 to the digit, ignoring the resultant carries into the next digit position.

ADDITION TABLE

Table D-1 tabulates the result of performing decimal addition on all possible combinations of one-digit augends and addends.

NOTES

1. The non-numeric digit codes, 0000, 0001, 0010, 1101, 1110, and 1111, are designated *a*, *b*, *c*, *d*, *e*, and *f*, respectively.
2. Results in gray produce a carry to the next digit position.
3. As described under the heading *Addition Procedure*, *a* in the augend, and *a*, *b*, and *c* in the addend, are changed to decimal 0. In the case of *b* and *c*, this conversion will result in a modulo-3 error unless the remainder, modulo-3, remains unchanged. For example, if 0001 0001 is changed to 0011 0011, the remainder of the value is changed from 2 to 0 and a modulo-3 error would result. If 0001 0010 is changed to 0011 0011, the remainder does not change, and no modulo-3 error would result.

SUBTRACTION

The procedure below, and table D-2, may be used to determine the result of a decimal subtraction (subtraction with like signs, or addition with unlike signs) involving non-numeric digits. Decimal addition involving non-numeric digits is described under the heading *Addition*, above.

SUBTRACTION PROCEDURE

1. Convert binary 0's (0000) in the operand from the arithmetic registers (minuend) to decimal 0's (0011). Convert binary 0's, 1's, and 2's (0000, 0001, 0010) in the operand from memory (subtrahend) to decimal 0's (0011).
2. Complement the subtrahend by changing 1-bits to 0-bits, and 0-bits to 1-bits.
3. Perform a binary addition on the subtrahend and minuend, adding binary 1 (0001) to the least significant digit position of the result. Note any carries from one digit position to the next.
4. Add an excess-three correction factor to each digit of the result as follows:
   a. If there was a carry from the most significant digit in step 3, add 0011 to the digit, ignoring resultant carries into the next digit position.
   b. If there was no carry from the most significant digit of the result of step 3, add 1101 to the digit, ignoring resultant carries into the next digit position.
5. If there was a carry from the most significant digit of the result in step 3, the result is in the correct

Table D-1. Decimal Addition

Addend (m')	Augend (ARi)															
	a 0000	b 0001	c 0010	0 0011	1 0100	2 0101	3 0110	4 0111	5 1000	6 1001	7 1010	8 1011	9 1100	d 1101	e 1110	f 1111
a 0000	0	b	c	0	1	2	3	4	5	6	7	8	9	0	1	2
b 0001	0	b	c	0	1	2	3	4	5	6	7	8	9	0	1	2
c 0010	0	b	c	0	1	2	3	4	5	6	7	8	9	0	1	2
0 0011	0	b	c	0	1	2	3	4	5	6	7	8	9	0	1	2
1 0100	1	c	0	1	2	3	4	5	6	7	8	9	0	1	2	3
2 0101	2	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4
3 0110	3	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5
4 0111	4	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6
5 1000	5	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7
6 1001	6	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8
7 1010	7	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
8 1011	8	6	7	8	9	0	1	2	3	4	5	6	7	8	9	d
9 1100	9	7	8	9	0	1	2	3	4	5	6	7	8	9	d	e
d 1101	0	8	9	0	1	2	3	4	5	6	7	8	9	d	e	f
e 1110	1	9	0	1	2	3	4	5	6	7	8	9	d	e	f	5
f 1111	2	0	1	2	3	4	5	6	7	8	9	d	e	f	5	6

form and the subtraction is complete; if there was no carry from the most significant digit of the result in step 3, the result is in complement form. To recomplement the result, repeat steps 2, 3, and 4, using the result as the subtrahend and a value of all decimal 0's as the minuend. The subtraction is now complete. (When performed by the arithmetic unit, this recomplementing procedure requires one memory cycle for each word of the result.)

SUBTRACTION TABLE

Table D-2 tabulates the result of performing decimal subtraction on all possible combinations of one-digit minuends and subtrahends.

NOTES

1. The non-numeric digit codes, 0000, 0001, 0010, 1101, 1110, and 1111, are designated *a*, *b*, *c*, *d*, *e*, and *f*, respectively.
2. It is assumed that there is no carry from the preceding digit position.
3. A minus sign indicates that recomplementing of the result was required.
4. A result in gray indicates that there was a carry to the next digit position upon recomplementing.
5. For special considerations involved in the conversion of *b* and *c*, refer to note 4 under the heading *Addition Table*.

Table D-2. Decimal Subtraction

Subtrahend (m′)	Minuend (ARi)															
	a 0000	b 0001	c 0010	0 0011	1 0100	2 0101	3 0110	4 0111	5 1000	6 1001	7 1010	8 1011	9 1100	d 1101	e 1110	f 1111
a 0000	0	−2	−1	0	1	2	3	4	5	6	7	8	9	d	e	f
b 0001	0	−2	−1	0	1	2	3	4	5	6	7	8	9	d	e	f
c 0010	0	−2	−1	0	1	2	3	4	5	6	7	8	9	d	e	f
0 0011	0	−2	−1	0	1	2	3	4	5	6	7	8	9	d	e	f
1 0100	−1	−3	−2	−1	0	1	2	3	4	5	6	7	8	9	d	e
2 0101	−2	−4	−3	−2	−1	0	1	2	3	4	5	6	7	8	9	d
3 0110	−3	−5	−4	−3	−2	−1	0	1	2	3	4	5	6	7	8	9
4 0111	−4	−6	−5	−4	−3	−2	−1	0	1	2	3	4	5	6	7	8
5 1000	−5	−7	−6	−5	−4	−3	−2	−1	0	1	2	3	4	5	6	7
6 1001	−6	−8	−7	−6	−5	−4	−3	−2	−1	0	1	2	3	4	5	6
7 1010	−7	−9	−8	−7	−6	−5	−4	−3	−2	−1	0	1	2	3	4	5
8 1011	−8	−0	−9	−8	−7	−6	−5	−4	−3	−2	−1	0	1	2	3	4
9 1100	−9	−1	−0	−9	−8	−7	−6	−5	−4	−3	−2	−1	0	1	2	3
d 1101	−0	−2	−1	−0	−9	−8	−7	−6	−5	−4	−3	−2	−1	0	1	2
e 1110	−1	−0	−2	−1	−0	−9	−8	−7	−6	−5	−4	−3	−2	−1	0	1
f 1111	−2	−6	−0	−2	−1	−0	−9	−8	−7	−6	−5	−4	−3	−2	−1	0

■ MULTIPLICATION

Table D-3 is used to determine the result of a multiplication involving non-numeric digits. The table shows the result of performing multiplication on all possible combinations of one-digit multiplicands and multipliers. Each operand is justified to the right within a computer word; the remaining five digits of the word are decimal 0's (0011).

NOTE

The non-numeric digits 0000, 0001, 0010, 1101, 1110, and 1111, are designated *a*, *b*, *c*, *d*, *e*, and *f*, respectively.

Table D-3. Decimal Multiplication

Digits Multiplied		Product		Digits Multiplied		Product	
(m′)	(AR1)	(AR2)	(AR3)	(m′)	(AR1)	(AR2)	(AR3)
a	a	00000a	aaaaa2	c	a	00000a	aaaaa4
	b	00000a	aaaaaa		b	00000a	aaaaaa
	c	000000	00000a		c	000000	00000c
	0	000000	000000		0	000000	000000
	1	000000	00000a		1	000000	00000c
	2	000000	000002		2	000000	00000b
	3	000000	00000c		3	000000	00000a
	4	000000	000004		4	000000	000004
	5	999999	999969		5	999999	999995
	6	999999	999966		6	999999	999994
	7	999999	999971		7	999999	999993
	8	999999	999968		8	999999	999992
	9	999999	999973		9	999999	999991
	d	000000	0000a0		d	000000	0000c0
	e	000000	0000aa		e	000000	0000cc
	f	000000	0000a2		f	000000	0000cb
b	a	00000a	aaaaa3	0	a	00000a	aaaaaa
	b	00000a	aaaaaa		b	00000a	aaaaaa
	c	000000	00000b		c	000000	000000
	0	000000	000000		0	000000	000000
	1	000000	00000b		1	000000	000000
	2	000000	000004		2	000000	000000
	3	000000	000002		3	000000	000000
	4	000000	000008		4	000000	000000
	5	999999	999974		5	000000	000000
	6	999999	999972		6	000000	000000
	7	999999	999978		7	000000	000000
	8	999999	999976		8	000000	000000
	9	999999	999982		9	000000	000000
	d	000000	0000b0		d	000000	000000
	e	000000	0000bb		e	000000	000000
	f	000000	0000b4		f	000000	000000

Table D-3. Decimal Multiplication (cont)

Digits Multiplied		Product		Digits Multiplied		Product	
(m′)	(AR1)	(AR2)	(AR3)	(m′)	(AR1)	(AR2)	(AR3)
1	a	00000a	aaaaab	4	a	00000a	aaaaa1
	b	00000a	aaaaaa		b	00000a	aaaaaa
	c	000000	000001		c	000000	000004
	0	000000	000000		0	000000	000000
	1	000000	000001		1	000000	000004
	2	000000	000002		2	000000	000008
	3	000000	000003		3	000000	000012
	4	000000	000004		4	000000	000016
	5	000000	000005		5	000000	000020
	6	000000	000006		6	000000	000024
	7	000000	000007		7	000000	000028
	8	000000	000008		8	000000	000032
	9	000000	000009		9	000000	000036
	d	000000	000010		d	000000	000040
	e	000000	000011		e	000000	000044
	f	000000	000012		f	000000	000048
2	a	00000a	aaaaac	5	a	00000a	aaaaa2
	b	00000a	aaaaaa		b	00000a	aaaaaa
	c	000000	000002		c	000000	000005
	0	000000	000000		0	000000	000000
	1	000000	000002		1	000000	000005
	2	000000	000004		2	000000	000010
	3	000000	000006		3	000000	000015
	4	000000	000008		4	000000	000020
	5	000000	000010		5	000000	000025
	6	000000	000012		6	000000	000030
	7	000000	000014		7	000000	000035
	8	000000	000016		8	000000	000040
	9	000000	000018		9	000000	000045
	d	000000	000020		d	000000	000050
	e	000000	000022		e	000000	000055
	f	000000	000024		f	000000	000060
3	a	00000a	aaaaa0	6	a	00000a	aaaaa3
	b	00000a	aaaaaa		b	00000a	aaaaaa
	c	000000	000003		c	000000	000006
	0	000000	000000		0	000000	000000
	1	000000	000003		1	000000	000006
	2	000000	000006		2	000000	000012
	3	000000	000009		3	000000	000018
	4	000000	000012		4	000000	000024
	5	000000	000015		5	000000	000030
	6	000000	000018		6	000000	000036
	7	000000	000021		7	000000	000042
	8	000000	000024		8	000000	000048
	9	000000	000027		9	000000	000054
	d	000000	000030		d	000000	000060
	e	000000	000033		e	000000	000066
	f	000000	000036		f	000000	000072

Table D-3. Decimal Multiplication (cont)

Digits Multiplied		Product		Digits Multiplied		Product	
(m')	(AR1)	(AR2)	(AR3)	(m')	(AR1)	(AR2)	(AR3)
7	a	00000a	aaaaa4	d	a	00000a	aaaaba
	b	00000a	aaaaaa		b	00000a	aaaaaa
	c	000000	000007		c	000000	000010
	0	000000	000000		0	000000	000000
	1	000000	000007		1	000000	000010
	2	000000	000014		2	000000	000020
	3	000000	000021		3	000000	000030
	4	000000	000028		4	000000	000040
	5	000000	000035		5	000000	000050
	6	000000	000042		6	000000	000060
	7	000000	000049		7	000000	000070
	8	000000	000056		8	000000	000080
	9	000000	000063		9	000000	000090
	d	000000	000070		d	000000	000100
	e	000000	000077		e	000000	000110
	f	000000	000084		f	000000	000120
8	a	00000a	aaaaa5	e	a	00000a	aaaabb
	b	00000a	aaaaaa		b	00000a	aaaaaa
	c	000000	000008		c	000000	000011
	0	000000	000000		0	000000	000000
	1	000000	000008		1	000000	000011
	2	000000	000016		2	000000	000022
	3	000000	000024		3	000000	000033
	4	000000	000032		4	000000	000044
	5	000000	000040		5	000000	000063
	6	000000	000048		6	000000	000074
	7	000000	000056		7	000000	00008a
	8	000000	000064		8	000000	00009b
	9	000000	000072		9	000000	00010c
	d	000000	000080		d	000000	000110
	e	000000	000088		e	000000	000121
	f	000000	000096		f	000000	000132
9	a	00000a	aaaaa6	f	a	00000a	aaaabc
	b	00000a	aaaaaa		b	00000a	aaaaaa
	c	000000	000009		c	000000	000012
	0	000000	000000		0	000000	000000
	1	000000	000009		1	000000	000012
	2	000000	000018		2	000000	000016
	3	000000	000027		3	000000	000028
	4	000000	000036		4	000000	000032
	5	000000	000045		5	000000	000076
	6	000000	000054		6	000000	000088
	7	000000	000063		7	000000	000092
	8	000000	000072		8	000000	000104
	9	000000	000081		9	000000	00011b
	d	000000	000090		d	000000	000120
	e	000000	000099		e	000000	000132
	f	000000	000108		f	000000	000136

■ DIVISION

Table D-4 may be used to determine the result of a division involving non-numeric digits. The table shows the result of performing division on all possible combinations of one-digit dividends and divisors. The dividend is justified to the right in AR1; AR2, and .the remaining five digits of AR1, are decimal 0's (0011). The divisor (contents of *m'*) is in the form *OOOOnO*, where *O* is a decimal 0 and *n* is the digit being used as the divisor.

The following cases shown in the table are improper divide operations in that the absolute value of the divisor is not greater than the absolute value of the dividend:

A divisor of a, b, or c with any dividend; or
A divisor of 0 with a dividend greater than c.

Under normal conditions, attempting to perform one of these operations will set the arithmetic-overflow indicator. A subsequent attempt to transfer the contents of AR1 to memory may set a modulo-3 error indicator.

NOTE

The non-numeric digits 0000, 0001, 0010, 1101, 1110, and 1111, are designated *a*, *b*, *c*, *d*, *e*, and *f*, respectively.

Table D-4. Decimal Division

Divisor	Dividend	Quotient	Remainder	Divisor	Dividend	Quotient	Remainder
(m')	(AR1, AR2)	(AR2)	(AR1)	(m')	(AR1, AR2)	(AR2)	(AR1)
a	a	000000	999950	c	a	000000	999990
	b	000000	999960		b	000000	9999d0
	c	000000	999970		c	000000	9999e0
	0	000000	999980		0	000000	000020
	1	000000	999990		1	000000	000030
	2	000000	000000		2	000000	000040
	3	000000	000010		3	000000	000050
	4	000000	000020		4	000000	000060
	5	000000	000030		5	000000	000070
	6	000000	000040		6	000000	000080
	7	000000	000050		7	000000	000090
	8	000000	000060		8	000000	0000d0
	9	000000	000070		9	000000	0000e0
	d	000000	000080		d	000000	0000f0
	e	000000	000090		e	000000	000050
	f	000000	0000d0		f	000000	000060
b	a	000000	999930	0	a	999999	000000
	b	000000	999940		b	999999	00C000
	c	000000	999950		c	999999	000000
	0	000000	999960		0	000000	000000
	1	000000	999970		1	000000	000010
	2	000000	999980		2	000000	000020
	3	000000	999990		3	000000	000030
	4	000000	000000		4	000000	000040
	5	000000	000010		5	000000	000050
	6	000000	000020		6	000000	000060
	7	000000	000030		7	000000	000070
	8	000000	000040		8	000000	000080
	9	000000	000050		9	000000	000090
	d	000000	000060		d	000000	0000d0
	e	000000	000070		e	000000	0000e0
	f	000000	000080		f	000000	0000f0

Table D-4. Decimal Division (cont)

Divisor (m')	Dividend (AR1, AR2)	Quotient (AR2)	Remainder (AR1)	Divisor (m')	Dividend (AR1, AR2)	Quotient (AR2)	Remainder (AR1)
1	a	999999	000010	4	a	999999	000040
	b	999999	000010		b	999999	000040
	c	999999	000010		c	999999	000040
	0	000000	000000		0	000000	000000
	1	100000	000000		1	025000	000000
	2	200000	000000		2	050000	000000
	3	300000	000000		3	075000	000000
	4	400000	000000		4	100000	000000
	5	500000	000000		5	125000	000000
	6	600000	000000		6	150000	000000
	7	700000	000000		7	175000	000000
	8	800000	000000		8	200000	000000
	9	900000	000000		9	225000	000000
	d	999999	000010		d	250000	000000
	e	999999	000010		e	275000	000000
	f	999999	000010		f	300000	000000
2	a	999999	000020	5	a	999999	000050
	b	999999	000020		b	999999	000050
	c	999999	000020		c	999999	000050
	0	000000	000000		0	000000	000000
	1	050000	000000		1	020000	000000
	2	100000	000000		2	040000	000000
	3	150000	000000		3	060000	000000
	4	200000	000000		4	080000	000000
	5	250000	000000		5	100000	000000
	6	300000	000000		6	120000	000000
	7	350000	000000		7	140000	000000
	8	400000	000000		8	160000	000000
	9	450000	000000		9	180000	000000
	d	500000	000000		d	199999	000050
	e	550000	000000		e	199999	000050
	f	600000	000000		f	199999	000050
3	a	999999	000030	6	a	999999	000060
	b	999999	000030		b	999999	000060
	c	999999	000030		c	999999	000060
	0	000000	000000		0	000000	000000
	1	033333	000010		1	016666	000040
	2	066666	000020		2	033333	000020
	3	100000	000000		3	050000	000000
	4	133333	000010		4	066666	000040
	5	166666	000020		5	083333	000020
	6	200000	000000		6	100000	000000
	7	233333	000010		7	116666	000040
	8	266666	000020		8	133333	000020
	9	300000	000000		9	150000	000000
	d	333333	000010		d	166666	000040
	e	366666	000020		e	183333	000020
	f	400000	000000		f	199999	000060

Table D-4. Decimal Division (cont)

Divisor	Dividend	Quotient	Remainder	Divisor	Dividend	Quotient	Remainder
(m′)	(AR1, AR2)	(AR2)	(AR1)	(m′)	(AR1, AR2)	(AR2)	(AR1)
7	a	999999	000070	d	a	999999	000100
	b	999999	000070		d	999999	000100
	c	999999	000070		c	999999	000100
	0	000000	000000		0	000000	000000
	1	014285	000050		1	010000	000000
	2	028571	000030		2	020000	000000
	3	042857	000010		3	030000	000000
	4	057142	000060		4	040000	000000
	5	071428	000040		5	050000	000000
	6	085714	000020		6	060000	000000
	7	100000	000000		7	070000	000000
	8	114285	000050		8	080000	000000
	9	128571	000030		9	090000	000000
	d	142857	000010		d	100000	000000
	e	157142	000060		e	110000	000000
	f	171428	000040		f	120000	000000
8	a	999999	000080	e	a	999999	8890c0
	b	999999	000080		b	999999	8890c0
	c	999999	000080		c	999999	8890c0
	0	000000	000000		0	000000	9999d0
	1	012500	000000		1	009090	0000d0
	2	025000	000000		2	019000	9899d0
	3	037500	000000		3	027353	000050
	4	050000	000000		4	037090	0000d0
	5	062500	000000		5	045535	000030
	6	075000	000000		6	055353	000050
	7	087500	000000		7	063709	000010
	8	100000	000000		8	073535	000030
	9	112500	000000		9	081900	9989d0
	d	125000	000000		d	090909	000010
	e	137500	000000		e	099999	0890c0
	f	150000	000000		f	109090	0000d0
9	a	999999	000090	f	a	999999	555640
	b	999999	000090		b	999999	555640
	c	999999	000090		c	999999	555640
	0	000000	000000		0	044444	000080
	1	011111	000010		1	060000	844480
	2	022222	000020		2	070444	000080
	3	033333	000030		3	082044	000080
	4	044444	000040		4	095555	000040
	5	055555	000050		5	099999	555640
	6	066666	000060		6	129555	000040
	7	077777	000070		7	139999	155640
	8	088888	000080		8	155555	000040
	9	100000	000000		9	167044	000080
	d	111111	000010		d	179555	000040
	e	122222	000020		e	182044	000080
	f	133333	000030		f	195555	000040

■ SHIFTING

The result of performing decimal shift operations on non-numeric digits is as follows:

Before Shifting		*After Shifting*	
Designation	*Bit Code*	*Bit Code*	*Designation*
a	0000	0000	0
b	0001	0011	0
c	0010	0011	0
d	1101	0011	0
e	1110	0100	1
f	1111	0101	2

The digits *d*, *e*, and *f* produce carries to the next higher digit position when shifted. Such carries are allowed to propagate through bit position 24 of the most significant arithmetic register designated. Carries beyond this point are inhibited so that overflow will not occur.

Except for *a*, shifting these digits will result in a modulo-3 error unless the remainder, modulo-3, of the operand shifted remains unchanged. For example, shifting 0010 1101 changes these digits to 0100 0011; the remainder of the value is changed from 2 to 1 and a modulo-3 error would result. Shifting 0001 0010 changes these digits to 0011 0011; the remainder of the value is unchanged and no modulo-3 error would result.

APPENDIX E

READER TRANSLATION OF NONSTANDARD 80-COLUMN CODES

This appendix contains a table which may be used to determine the result in memory of reading with translation 80-column cards punched in a nonstandard code. Each entry in the *Bit Configuration* column of the table corresponds to one of the 64 possible bit configurations of a character in memory; each entry in the *Punch Combination(s)* column gives the punch combinations that produce the given bit configuration when read with translation.

To facilitate use of the table, the following rules concerning zone punches should be considered:

If rows 12*, 11*, and 0 of a column contain no punches, the zone of the bit configuration in memory will be *00*.

If the zone of a column contains only a 12-punch, the zone in memory will be *01*.

If the zone of a column contains only an 11-punch, the zone in memory will be *10*.

If the zone of a column contains any other combination of punches, the zone in memory will be *11*.

In the table, each punch combination is described using the following notation:

n.	Row *n* contains a punch.
$\bar{\text{n}}$.	Row *n* does not contain a punch.
n,n′.	Rows *n* and *n′* are punched as indicated.
n—n′. . .	Rows *n* through *n′* may contain any combination of punches, but row *n* is punched as indicated. For example, *0—12* designates the following combinations of punches: 0 0,11 0,12 0,11,12; $\bar{0}$—12 designates 11 12 11,12.

*In this appendix, a Y-punch is designated *12* and an X-punch is designated *11*.

Table E-1. Punch Combination(s) of Nonstandard 80-Column Card Codes

Bit Configuration		Punch Combination(s)	Bit Configuration		Punch Combination(s)
Zone	Numeric		Zone	Numeric	
00	0000	Blank	01	0000	12
	0001	8,4—2,1 8,4		0001	12,8,4—2,1 12,8,4
	0010	11 8,3,2,1 8,3,1 8,3		0010	12,11 12,8,3,2,1 12,8,3,1 12,8,3
	0011	0		0011	12,0
	0100	1		0100	12,1
	0101	2—1		0101	12,2—1
	0110	3—1		0110	12,3—1
	0111	4—1		0111	12,4—1
	1000	5—4,$\bar{3}$—1		1000	12,5—4,$\bar{3}$—1
	1001	6—1 5—4,3—1		1001	12,6—1 12,5—4,3—1
	1010	7—4,$\bar{3}$—1		1010	12,7—4,$\bar{3}$—1
	1011	8 8,7—1 8,2—1 8,1 7—4,3—1		1011	12,8 12,8,7—1 12,8,2—1 12,8,1 12,7—4,3—1
	1100	9—4,$\bar{3}$—1		1100	12,9—4,$\bar{3}$—1
	1101	9—4,3—1 8,6—1		1101	12,9—4,3—1 12,8,6—1 8,3,2 12,8,3,2
	1110	8,5—4,$\bar{3}$—1		1110	12,8,5—4,$\bar{3}$—1
	1111	8,5—4,3—1		1111	12,8,5—4,3—1

Table E-1. Punch Combination(s) of Nonstandard 80-Column Card Codes (cont)

Bit Configuration		Punch Combination(s)	Bit Configuration		Punch Combination(s)
Zone	Numeric		Zone	Numeric	
10	0000	[*]	11	0000	[*]
	0001	11,8,4—2,1 11,8,4		0001	0—12,8,4—2,1 0—12,8,4 12,11,8,4—2,1 12,11,8,4
	0010	11,8,3,2,1 11,8,3,1 11,8,3		0010	0—12,8,3,2,1 0—12,8,3,1 0—12,8,3 12,11,8,3,2,1 12,11,8,3,1 12,11,8,3
	0011	11,0		0011	0,11,12 0—12,8,4—2 12,11,8,4—2 8,4—2
	0100	11,1		0100	0—12,1 12,11,1
	0101	11,2—1		0101	0—12,2—1 12,11,2—1
	0110	11,3—1		0110	0—12,3—1. 12,11,3—1
	0111	11,4—1		0111	0—12,4—1 12,11,4—1
	1000	11,5—4,3̄—1		1000	0—12,5—4,3̄—1 12,11,5—4,3̄—1
	1001	11,6—1 11,5—4,3—1		1001	0—12,6—1 0—12,5—4,3—1 12,11,6—1 12,11,5—4,3—1
	1010	11,7—4,3̄—1		1010	0—12,7—4,3̄—1 12,11,7—4,3̄—1
	1011	11,8 11,8,7—1 11,8,2—1 11,8,1 11,7—4,3—1		1011	0—12,8 0—12,8, 7—1 0—12,8,2—1 0—12,8,1 0—12,7—4,3—1 12,11,8 12,11,8,7—1 12,11,8,2—1 12,11,8,1 12,11,7—4,3—1
	1100	11,9—4,3̄—1		1100	0—12,9—4,3̄—1 12,11,9—4,3̄—1
	1101	11,9—4,3—1 11,8,6—1		1101	0—12,9—4,3—1 0—12,8,6—1 0—12,8,3,2 12,11,9—4,3—1 12,11,8,6—1 11—12,8,3,2
	1110	11,8,5—4,3̄—1		1110	0—12,8,5—4,3̄—1 12,11,8,5—4,3̄—1
	1111	11,8,5—4,3—1		1111	0—12,8,5—4,3—1 12,11,8,5—4,3—1

[*] No punch combination will give this bit configuration.

HIGH-SPEED PRINTER TIMING

This appendix contains the method for determining approximate speeds, in lines per minute, for continuously printing characters taken from the same group of contiguous characters on the type drum. Rates for several typical printing situations are given in table F-1.

Throughout this appendix it is assumed that characters from the same group of characters on the drum are to be printed and that paper advance and vertical line spacing remain unchanged from one line to the next. The time required to print a single line depends on paper-advance time and printing time.

Paper-advance time, t, in milliseconds is given by

$$t = 20 + K(L - 1)$$

where L = number of lines of paper advance,
and K = 8.3 for 6 lines per inch spacing or 6.25 for 8 lines per inch spacing.

Printing time depends on the group of characters to be printed. If they comprise the entire 51-character set, then an entire type-drum revolution, 65 milliseconds (0.00108 minute), will be required for printing. If the characters to be printed comprise only a part of the entire character set, for example when only numerics are printed, the printing time is computed as shown in figure F-1.

Table F-1. Typical Printing Rates

Characters to be Printed	Line Spacing:	Average Lines per Minute
All alphanumeric	Single	705
	Double	642
	Triple	590
	Quadruple	545
0 1 2 3 4 5 6 7 8 9	Single through quadruple	922
− 0 1 2 3 4 5 6 7 8 9 : < > +) .	Single through triple	922
	Quadruple	615
0 1 2 3 4 5 6 7 8 9 : < > . A B C D E F G H I = * $	Single and double	922
	Triple	615
	Quadruple	615

Figure F-1 represents the timing involved when printing the following characters:

— 0 1 2 3 4 5 6 7 8 9 : < > +) .

It is assumed that four lines of paper are to be advanced, with a vertical spacing of six lines per inch. Type-drum character positions are diagrammed together with the concurrent operations performed on the paper. Line *n* on the diagram begins at *A*, when line *n — 1* is complete. The four-line paper-advance time is calculated from the preceding equation as follows:

$$t = 20 + 8.3(3)$$
$$= 44.9 \text{ milliseconds}$$

Paper advance will stop at *B*, when the character 0 is aligned with the print hammers. The characters from 0 to . (period) are printed from *B* to *C*. From *C* to *D*, no hammers are actuated; from *D* to *E*, the last character, — (minus), is printed. Paper advance for line *n + 1* then follows at *E*. At *F*, paper advance stops but none of the characters to be printed are in line with the print hammers until *G* is reached. Between *G* and *H*, all of line *n + 1* is printed. This two-line cycle of events repeats continuously. From the diagram it may be noted that these two lines were printed in three revolutions of the type drum; thus, the line-per-minute rate is

$$\frac{2 \text{ lines}}{3 \text{ revolutions} \times 0.00108 \text{ minutes/revolution}}$$
$$= 615 \text{ lines per minute}$$

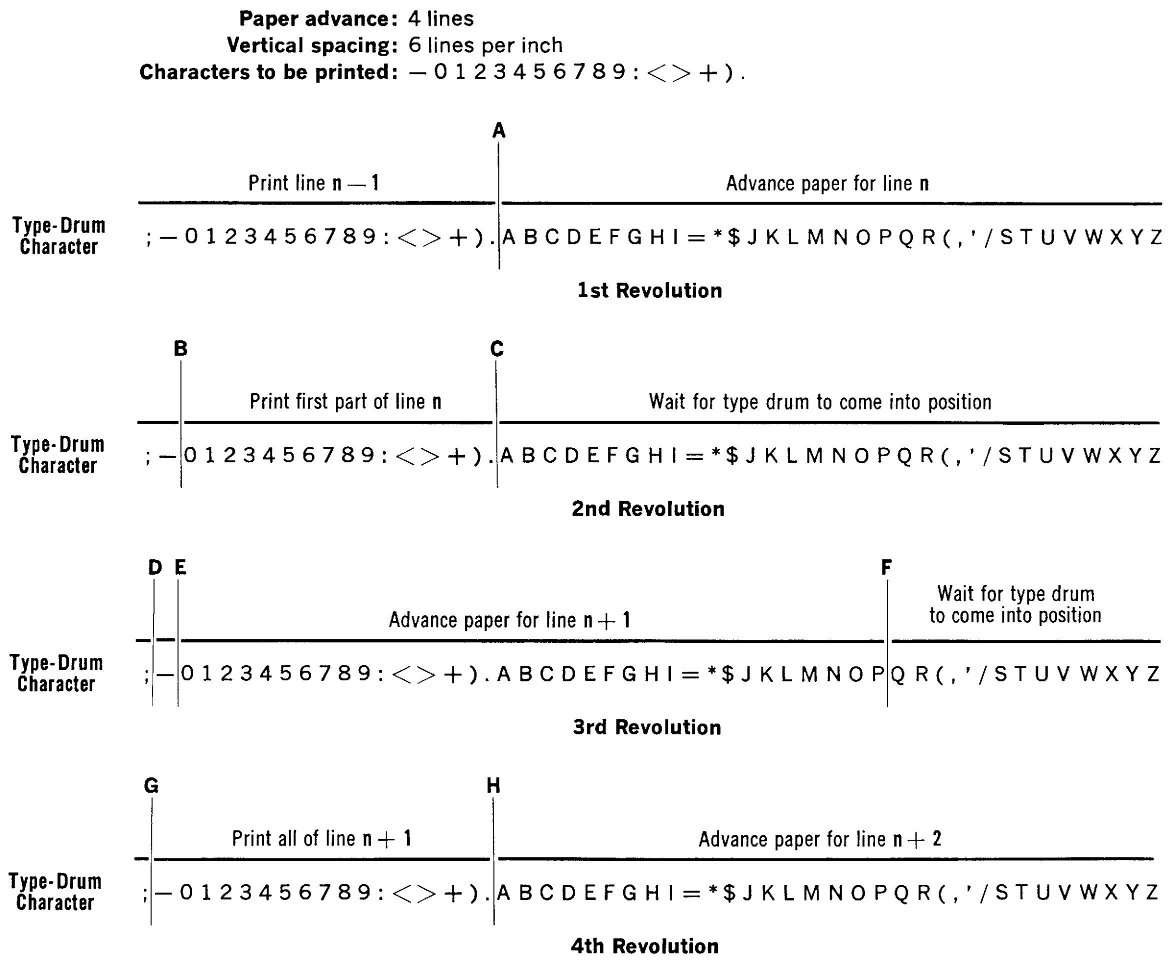

Paper advance: 4 lines
Vertical spacing: 6 lines per inch
Characters to be printed: — 0 1 2 3 4 5 6 7 8 9 : < > +) .
A
Print line n — 1
Advance paper for line n
Type-Drum Character
; — 0 1 2 3 4 5 6 7 8 9 : < > +) . A B C D E F G H I = * $ J K L M N O P Q R (, ' / S T U V W X Y Z
1st Revolution
B
C
Print first part of line n
Wait for type drum to come into position
2nd Revolution
D E
F
Advance paper for line n + 1
Wait for type drum to come into position
3rd Revolution
G
H
Print all of line n + 1
Advance paper for line n + 2
4th Revolution

Figure F-1. Continuous Printing of Partial Character Set, Timing Diagram

APPENDIX G

INPUT-OUTPUT SPECIFICATIONS

This appendix contains a summary of equipment specifications for the following input-output units of the UNIVAC III System: UNISERVO IIIA tape system, High-Speed Reader, Card-Punch Unit, and High-Speed Printer.

TAPE SYSTEM

PROGRAM-CONTROLLED FUNCTIONS:
Gather-write,
Block- or scatter-read,
Rewind with or without interlock,
Tape-error recovery, and
Load-point test.

TAPE SPEED:
Read or write—100 inches per second
Rewind—125 seconds for a 3600-foot reel of tape

RECORDING:
9 channels; 1000 frames per inch; the 27 bits of a UNIVAC III word occupy 3 frames

CHARACTER-TRANSFER RATES:
Alphanumeric characters—133,000 per second
Decimal digits—200,000 per second

BLOCK LENGTH:
Variable—under program control

INTERBLOCK GAP:
Approximately 0.6 inch

TAPE CHARACTERISTICS:
Base—MYLAR
Coating—oxide
Width—½ inch
Length—3600 feet (3500 feet recordable)
Thickness—1 mil

DIRECTION:
Read—forward or backward
Write—forward

CHECKING:
Information written onto tape is checked by a read head after being written;
Information read from or written onto tape is modulo-3 checked;
Bad spots on tape are marked with special bit configurations which are not transferred to memory; and
Load point and end-of-tape warning are detected photoelectrically.

HIGH-SPEED READER

PROGRAM-CONTROLLED FUNCTIONS:
Card feeding, stacker selection, program interrupt, memory-address selection, and automatic translation from card code to UNIVAC III character code.

CARD-FEED RATE:
700 cards per minute
After the picker knife places a card in the card-feed path, the card moves through the reader continuously until it is deposited in one of the output stackers.

INPUT-MAGAZINE CAPACITY:
2000 cards

OUTPUT-STACKER CAPACITY:
1000 cards in each of three program-selectable output stackers

SENSING STATIONS:
Two—one for checking, and one for data transfers to memory

CHECKING:
The following checks are made on each card that is read:
Hole count;
Count of number of memory accesses; and
Modulo-3 check on data transfers.

CARD-PUNCH UNIT

PROGRAM-CONTROLLED FUNCTIONS:
Card feeding, stacker selection, program interrupt, memory-address selection, and automatic translation from UNIVAC III character code to card code.

CARD-FEED RATE:
300 cards per minute

CARD MOTION:
After a card is placed in the card-feed path by the picker knife, the card is moved in fixed steps until the continuously moving output rollers deposit it in one of the output stackers.

INPUT-MAGAZINE CAPACITY:
1000 cards

OUTPUT-STACKER CAPACITY:
1000 cards in each of two output stackers

PUNCHING:
Row-by-row for both 80- and 90-column cards.

CHECKING:
The following checks are made on each card punched:
Hole count;
Count of number of memory accesses; and
Modulo-3 check on data transfers.

HIGH-SPEED PRINTER

PROGRAM-CONTROLLED FUNCTIONS:
Paper advance and printing

SPEED:
Alphanumeric Data—700 lines per minute with single spacing
Numeric Data—922 lines per minute with single spacing

CHARACTER SET:
51 printing characters—A through Z, 0 through 9, and 15 special symbols and punctuation marks:
; — : < > +) . = * $ (, ' /

CHARACTERS PER LINE:
128

SPACING:
Horizontal—10 characters per inch
Vertical—6 or 8 lines per inch (operator-controlled)

PAPER:
Continuously folded, sprocket-fed paper from 4 to 22 inches wide, with up to 22 inches between folds.

NUMBER OF COPIES:
An original and up to 5 carbons using paper between approximately 11 and 13.5 pounds in weight

RIBBON:
Nylon, self-reversing

PAPER-ADVANCE TIMING:
First Line—10 milliseconds
Succeeding Lines—8.3 milliseconds (6 lines per inch) or
6.25 milliseconds (8 lines per inch)
Paper Stabilization—10 milliseconds

CHECKING:
Paper runaway; low paper supply; modulo-3 check when either an FS or data is transferred to the synchronizer

FIXED MEMORY LOCATIONS

Table H-1 lists the fixed memory locations which are assigned special functions by the control circuitry. In all other respects, these locations do not differ from the rest of memory.

Table H-1. Fixed Memory Locations

Address	Function
	Standby Locations
0003	UNISERVO IIIA synchronizer, basic write channel
0004	UNISERVO IIIA synchronizer, basic read channel
0005	General-purpose channel 1
0006	General-purpose channel 2
0007	General-purpose channel 3
0008	General-purpose channel 4
0009	General-purpose channel 5
0010	General-purpose channel 6
0011	General-purpose channel 7
0012	General-purpose channel 8
0013	UNISERVO IIA synchronizer, read-write channel
0014	UNISERVO IIIA additional synchronizer, write channel
0015	UNISERVO IIIA additional synchronizer, read channel
	Interrupt Locations
0016	Processor error, control-counter storage
0017	Processor error, next instruction
0018	Contingency, control-counter storage
0019	Contingency, next instruction
0020	Input-output, control-counter storage
0021	Input-output, next instruction

INSTRUCTIONS AND FUNCTION SPECIFICATIONS

This appendix summarizes the central-processor instructions and the input-output function specifications of the UNIVAC III System. Table I-1 lists the central-processor instructions by functional category, giving the following information for each instruction: mnemonic code, operation code, and whether field selection and multiword operands may be used. Table I-2 lists the input-output function specifications for each input-output device, giving the following information for each function specification: mnemonic code, operation, and function code.

Table I-1. Central-Processor Instructions

Mnemonic Code	Operation Code	Operation	Options	
			Field Selection	Multiword Operands
		Operand-Transfer Instructions		
L	12	(m')⟶ ARi	Yes	Yes
LCS	13	— (m')⟶ ARi	Yes	Yes
EXT	14	Extract (m')⟶ ARi	Yes	Yes
ST	10	(ARi)⟶ m'	No	Yes
STCS	11	— (ARi)⟶ m'	No	Yes
		Arithmetic Instructions		
A	20	(ARi) + (m')⟶ ARi	Yes	Yes
AH	22	(ARi) + (m')⟶ ARi', where i' > i	Yes	Yes
S	21	(ARi) — (m')⟶ ARi	Yes	Yes
SH	23	(ARi) — (m')⟶ ARi', where i' > i	Yes	Yes
M	30	(m') × (AR1)⟶ AR2, AR3	No	No
D	31	(AR1, AR2) ÷ (m'); quotient⟶ AR2 remainder⟶ AR1	No	No
BA	24	(ARi) + (m')⟶ ARi	Yes	Yes
BAH	26	(ARi) + (m')⟶ ARi' where i' > i	Yes	Yes
BS	25	(ARi) — (m')⟶ ARi	Yes	Yes
BSH	27	(ARi) — (m')⟶ ARi', where i' > i	Yes	Yes
		Comparison Instructions		
C	54	(ARi) : (m')	Yes	Yes
CA	55	|(ARi)| : |(m')|	Yes	Yes
CONE	57	1-bits (ARi) : 1-bits (m')	Yes	Yes
CZRO	56	1-bits (ARi) : 0-bits (m')	Yes	Yes
		Logical-Branching Instructions		
TEQ	60	Test equal comparison indicator: if set, m'⟶ CC if reset, (CC) + 1⟶ CC	No	No
THI	60	Test high comparison indicator: if set, m'⟶ CC if reset, (CC) + 1⟶ CC	No	No
TLO	60	Test low comparison indicator: if set, m'⟶ CC if reset, (CC) + 1⟶ CC	No	No
TPOS	60	Test sign of AR: if +, m'⟶ CC if —, (CC) + 1⟶ CC	No	No
TUN	06	m'⟶ CC	No	No
TR	07	(CC/MAC) + 1⟶ m' ; m' + 1⟶ CC	No	No

Table I-1. Central-Processor Instructions (cont)

Mnemonic Code	Operation Code	Operation	Options: Field Selection	Options: Multiword Operands
		Sense-Indicator Instructions		
SSI	62	Set sense indicator	No	No
RSI	61	Reset sense indicator	No	No
TSI	60	Test sense indicator: if set, m′⟶ CC; if reset, (CC) + 1⟶ CC	No	No
		Initiate-Input-Output Function Instruction		
IOF	70	(m′)⟶ channel standby location; set channel standby-location interlock indicator	No	No
		Input-Output Interrupt Instructions		
TIO	64	Test I-O indicators: if set, (CC) + 1⟶ CC; if reset, (CC) + 2⟶ CC	No	No
RIO	65	Reset I-O indicators. Reset input-output interrupt-mode indicator.	No	No
PIO	62	Set inhibit-input-output interrupt indicator	No	No
AIO	61	Reset inhibit-input-output interrupt indicator	No	No
TIOP	60	Test inhibit-input-output interrupt indicator: if set, (CC) + 1 → CC; if reset, (CC) + 2⟶ CC	No	No
		Processor-Error and Contingency Interrupt Instructions		
TCI	64	Test contingency indicators: if set, (CC) + 1⟶ CC; if reset, (CC) + 2⟶ CC	No	No
RCI	65	Reset contingency indicators. Reset contingency interrupt-mode indicator.	No	No
TPE	64	Test processor-error indicators: if set, (CC) + 1⟶ CC; if reset, (CC) + 2⟶ CC	No	No
RPE	65	Reset processor-error indicators. Reset processor-error interrupt-mode indicator.	No	No
		Console-Typewriter Instructions		
ACT	66	Activate keyboard	No	No
RT	01	(ARi) + (TBR)⟶ ARi	No	No
WT	02	Typewriter on-line: 1 character (m′)⟶ TBR; (CC) + 2⟶ CC. Typewriter off-line: (CC) + 1⟶ CC	No	No
		Miscellaneous Instructions		
NOP	00	No operation	No	No
STMC	04	(MACi)⟶ m′	No	No
STCR	05	(TCWRi)⟶ m′	No	No
WAIT	77	m′⟶ CC; Stop Central Processor	No	No
DIS	03	(m′) → Display	No	No
LT	76	(Clock)⟶ ARi	No	No

Table I-2. Summary of Function Specifications

Mnemonic Code	Function-Specification Operation	Function Code							
		Bit Position							
		24	23	22	21	20	19	18	17
	UNISERVO IIIA Tape Units								
GWT	Gather-write Tn					0	0	0	1
OWT	Overlay patterns ⟶ Tn, gather-write Tn					0	1	0	1
FSR	Forward scatter-read Tn					0	0	0	0
BSR	Backward scatter-read Tn					0	0	1	0
FBR	Forward-block-read, Tn ⟶ L. . .(L + W) — 1					0	1	0	0
BBR	Backward-block-read, Tn ⟶ L. . .(L — W) + 1	Tape-unit				0	1	1	0
FCSR	Forward-contingency scatter-read Tn	number				1	0	0	0
BCSR	Backward-contingency scatter-read Tn	(0 through 15)				1	0	1	0
FCBR	Forward-contingency block-read, Tn ⟶ L. . .(L + W) — 1					1	1	0	0
BCBR	Backward-contingency block-read, Tn ⟶ L. . .(L — W) + 1					1	1	1	0
RWI	Rewind Tn with interlock (L is ignored)					0	1	1	1
RW	Rewind Tn without interlock (L is ignored)					0	0	1	1
LPT	Test Tn for load point					1	1	1	1
	W = number of words in a block								
	UNISERVO IIA Tape Units								
CWRT	Compatible write, L. . .L + 179 ⟶ Tn					0	0	0	1
CWSD	Compatible write subdivided, L. . .L + 179 ⟶ Tn					0	1	0	1
CFRH	Compatible forward-read high gain, Tn ⟶ L. . .L + 179					1	0	0	0
CFRL	Compatible forward-read low gain, Tn ⟶ L. . .L + 179	Tape-unit				0	1	0	0
CFRN	Compatible forward-read normal gain, Tn ⟶ L. . .L + 179	number				0	0	0	0
CBRH	Compatible backward-read high gain, Tn ⟶ L. . .L + 179	(0 through 5)				1	0	1	0
CBRL	Compatible backward-read low gain, Tn ⟶ L. . .L — 179					0	1	1	0
CBRN	Compatible backward-read normal gain, Tn ⟶ L. . .L — 179					0	0	1	0
CRWI	Compatible rewind with interlock					1	0	1	1
CRW	Compatible rewind					0	0	1	1
	High-Speed Reader*								
FCT	Feed card, select stacker 0, image with translation ⟶ L. . .L + 19					0	0	1	1
FCTS1	Feed card, select stacker 1, image with translation ⟶ L. . .L + 19					0	1	1	1
FCTS2	Feed card, select stacker 2, image with translation ⟶ L. . .L + 19					1	0	1	1
FC	Feed card, select stacker 0, image without translation ⟶ L. . .L + 39					0	0	0	1
FCS1	Feed card, select stacker 1, image without translation ⟶ L. . .L + 39					0	1	0	1
FCS2	Feed card, select stacker 2, image without translation ⟶ L. . .L + 39	All				1	0	0	1
CT	Select stacker 0, image with translation ⟶ L. . .L + 19	0's				0	0	1	0
CTS1	Select stacker 1, image with translation ⟶ L. . .L + 19					0	1	1	0
CTS2	Select stacker 2, image with translation ⟶ L. . .L + 19					1	0	1	0
CAD	Select stacker 0, image without translation ⟶ L. . .L + 39					0	0	0	0
CS1	Select stacker 1, image without translation ⟶ L. . .L + 39					0	1	0	0
CS2	Select stacker 2, image without translation ⟶ L. . .L + 39					1	0	0	0
	Card-Punch Unit*								
PC	Feed card, select stacker 0, image without translation L. . .L + 39 ⟶ punch					0	0	0	1
PCT	Feed card, select stacker 0, image with translation L. . .L + 19 ⟶ punch	All				0	0	1	1
PCS	Feed card, select stacker 1, image without translation L. . .L + 39 ⟶ punch	0's				0	1	0	1
PCTS	Feed card, select stacker 1, image with translation L. . .L + 19 ⟶ punch					0	1	1	1
CCS	Select stacker 1					0	1	0	0
	High-Speed Printer								
PRT	Advance paper and print L. . .L + 31	Number of lines						1	0
PAD	Advance paper (L is ignored)							0	0

* Transfer addresses given here are for 80-column card only. Both translated and untranslated 90-column card images are transferred to and from L . . . L + 23.

INDEX

UNIVAC® III REFERENCE MANUAL

UNIVAC

DIVISION OF SPERRY RAND CORPORATION

U T 2488 6-62

Printed in U. S. A.