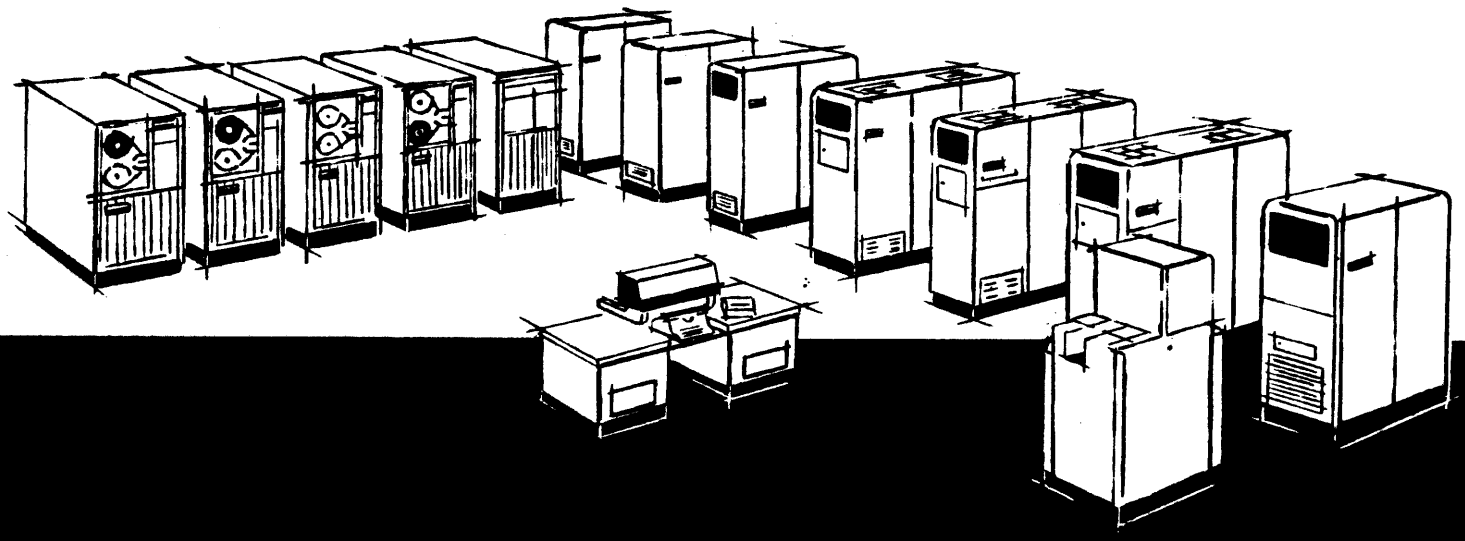
SPECIFICATIONS



UNIVAC®

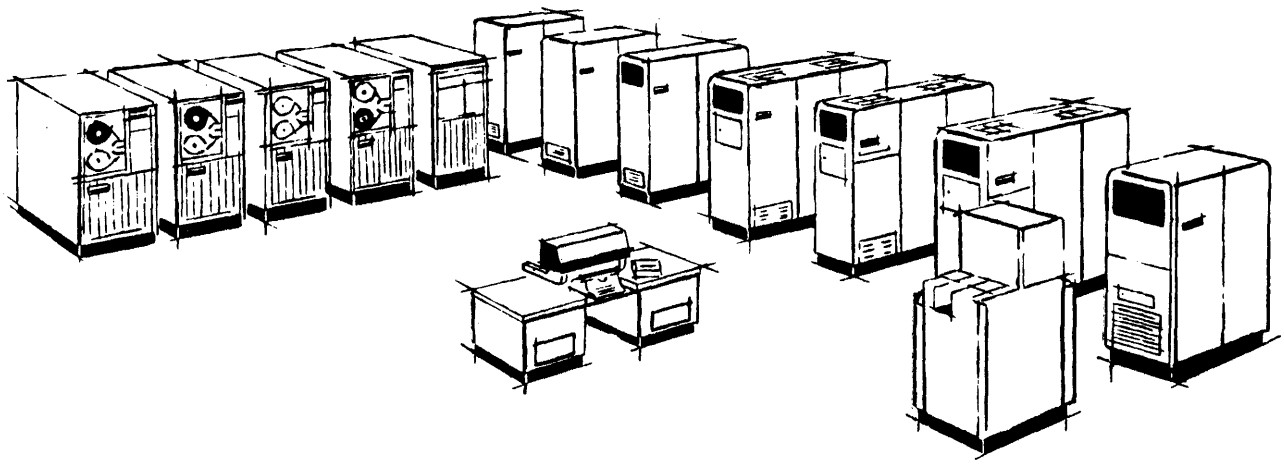
File-Computer System

MODEL 1

Central Computer and General Storage System

Remington Rand Univac

SPECIFICATIONS



UNIVAC®

File-Computer System

MODEL 1

Central Computer and
General Storage System

Remington Rand Univac
DIVISION OF SPERRY RAND CORPORATION

TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

TABLE OF CONTENTS (CONTINUED)

LIST OF FIGURES

LIST OF FIGURES (CONTINUED)

LIST OF TABLES

*Most commonly used tables.

INTRODUCTION

FEATURES OF UNIVAC FILE-COMPUTER, MODEL 1

The UNIVAC File-Computer, Model 1, is the medium sized, general purpose member of the Remington Rand family of UNIVAC computers. It possesses unique features which contribute to its data processing versatility. The most outstanding are described briefly below:

Any grouping of input/output devices up to ten in number may be connected to the central computer at the same time. These devices include the UFC-1 Console, Inquiry Typewriter, 90-Column Card Input/Output, 80-Column Card Input/Output, Paper Tape Input/Output, Magnetic Tape Input/Output, and special purpose equipments such as those used in the Airline Reservation System.

Each input device in the UFC-1 System automatically translates data from its own language (punched card, paper tape, etc.) to the language (UNIVAC code) of the central computer. Each output device translates data from the language of the central computer to its own language. Therefore, no central computer time is lost in data translation.

The UFC-1 may incorporate at the present time from one to ten large-capacity magnetic drums in a system, allowing for random access to stored data. This ability of the computer to accept data in any order eliminates the need for prior sorting of input information. When the correct address of stored data is not known, it may be located through a channel search without interruption of other computer operations.

Through the demand stations associated with each input/output device, the central computer and several input/output devices may function independently, except during intervals when control information is being exchanged by the computer and the I/O device. General storage operations, involving the large-capacity, general storage drums, may be carried on simultaneously with central computer operations and with input/output operations. The "busy" or "not busy" condition of both the general storage system and the various input/output devices may be determined without interrupting any operation currently in progress. During execution of internally stored instructions the next instruction is located and readied for execution while the current instruction is in progress.

The UFC-1 operates as an internally programmed computer through a series of sequentially stored instruction words. It operates as a plugboard-programmed computer through the wiring of program steps on a 48-step main program plugboard which incorporates other plugboard controlled devices, such as selectors, branches, etc. Most programs developed for the UFC-1 incorporate both internally stored and plugboard defined instructions in a single interrelated program to exploit the strongest features of each type of programming.

During all data transmission operations, a redundant parity check is conducted by the computer. The computer stops automatically at the point in the program where a parity error occurs. All arithmetic operations of the computer may be checked for accuracy by a reverse operation; i.e., addition is checked by subtraction. In internally defined programs, checking is performed automatically unless suppressed by the programmer. In plugboard defined programs, a check or no-check decision is affected by the wiring of the process hubs.

The UFC-1 handles any of the sixty-three UNIVAC code characters with equal facility, regardless of whether the character is a number, letter, or special symbol. No special programming procedure is required to handle alphabetic information or special symbols.

Three-address logic is the three-part principle of computer instruction which includes (a) the address (or storage location) from which the first operand is obtained, (b) the address from which the second operand is obtained and, (c) a third address where the result is to be stored.

In UFC-1 internal programming, each twelve-character instruction word may contain (a) the addresses of two operands, (b) the address at which the result is to be stored, (c) the basic process to be performed, and (d) a sub-instruction which may modify or extend the basic operation. Up to 850 of these powerful instruction words may be stored in the high speed drum. The 48-step program control plugboard also operates basically in three-address logic.

FEATURES OF UNIVAC FILE-COMPUTER, MODEL 1, SPECIFICATIONS MANUAL

This present edition, a revised reference manual, contains a thorough discussion of the specifications of this electronic computer, incorporating recent modifications, necessary corrections of the original text, and a detailed index.

Section I, "UFC Model 1 System Resume," describes briefly the versatile features of the central computer, the general storage system, and the input/output system, and lists the program step processes, instruction word processes, sub-instructions, sub-steps and storage locations.

In Section II, "Program Control," both internally stored and plugboard defined programs, together with the internal instructions and plugboard processes contained in these programs, are defined, explained and diagrammed. So also are the sub-instructions or sub-steps which expand and modify the basic operations. Rules and process times are included in the individual tables provided for each instruction word. Details on the operation of the Arithmetic Registers, which are involved in most of the instructions, are given here, rather than in Section IV.

Section III, "Central Computer Operating Memory" discusses the Program Control Storage System, the addressable memory locations which comprise the central operating memory of the computer and the data transmissions which result when these locations are referred to.

Section IV, the "Arithmetic Section" lists the specific instructions which involve the Arithmetic Section, with the initial and final contents of each of the Arithmetic Registers.

Section V, "Input/Output System," treats its subject comprehensively, and there is a brief comparison of the General Storage and Input/Output Systems, summarizing their relationships and differences. Particular emphasis is placed on the demand stations and time-sharing features of the input/output system.

Section VI, "General Storage Data Transmissions," describes the random-access large capacity external memory of the UFC-1, whose principal parts are the General Storage Drums, General Storage Address Register, and General Storage Buffer. Their operations are explained and their parts are diagrammed.

Appendix A, a fifty-two page "Glossary of Terms for the UNIVAC File-Computer, Model 1," contains definitions and explanations of UFC-1 components, functions, and terms.

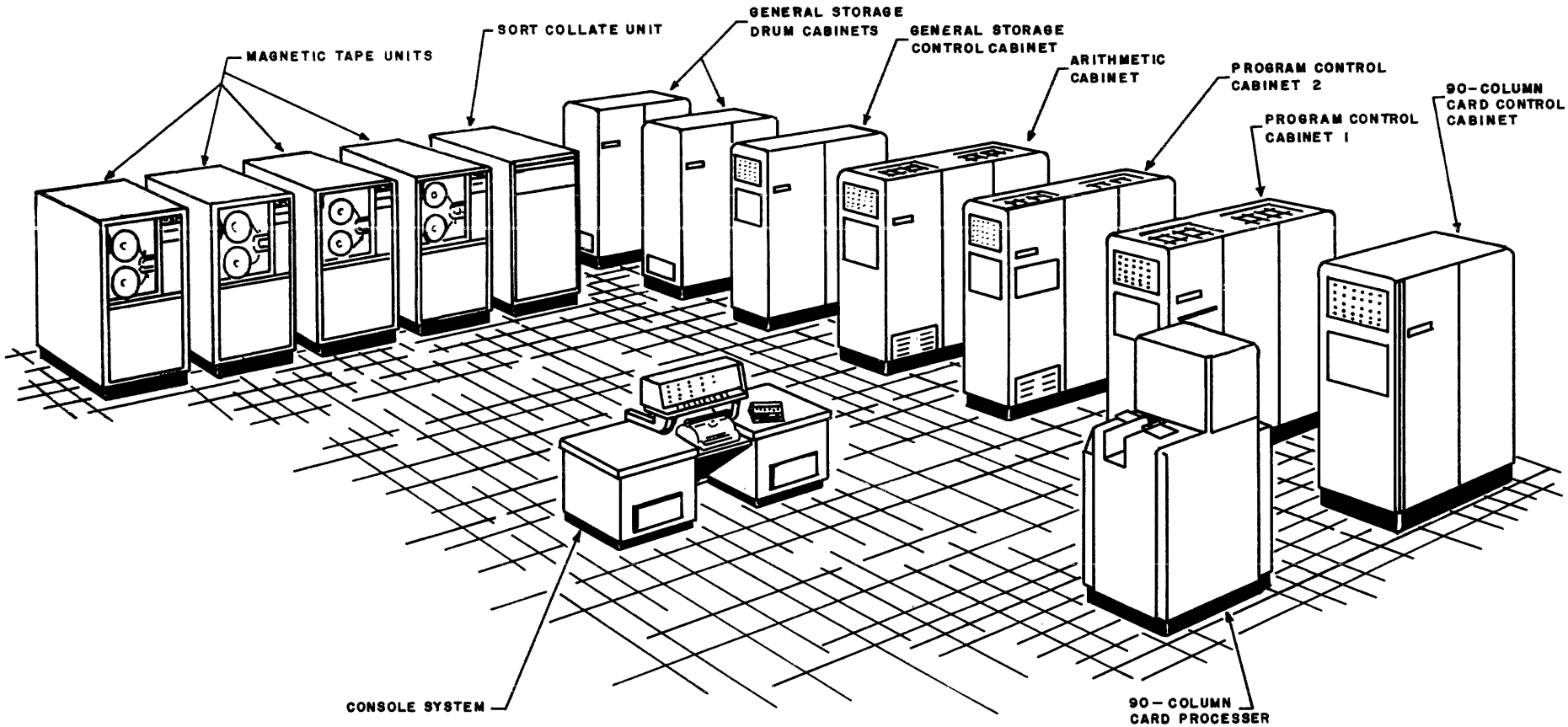
Appendix B, a "List of Frequently Used Abbreviations," provides quick reference for abbreviated proper names. Throughout the text, in diagrams, in tables, and on plugboards, abbreviations are frequently used in order to simplify exposition and to conserve space. For expanded definitions, the reader may then wish to check the glossary and the index.

Appendix C, a cross-referenced "Index," provides a simple means of locating information related to any topic, even though it may appear in various sections of the manual. Subjects are referred to by section (Roman numerals) and page (Arabic numerals). The designations "f" and "ff", following a page number, indicate that information about the subject is found on the numbered page and on the following page ("f") or on two or more following pages ("ff").

Appendix D, "Program Control Plugboard," contains a detailed drawing of the UFC-1 plugboard; opposite the drawing is an alphabetized key for locating hubs on the plugboard.

For further information regarding programming of the UFC-1 and peripheral equipment, the reader is referred to the UNIVAC File-Computer, Model 1, Basic Programming Manual, U-1474 and manuals for the Input/Output Equipment.

Figure I-1. A medium-sized UFC Model 1 System.



I UFC MODEL 1 SYSTEM RESUME

A Univac File Computer Model 1 System is an electronic data-processing system which features the simultaneous operation of:

a UFC Model 1 Central Computer;

a large-capacity, random-access, magnetic drum memory called General Storage; and

an integrated system of UFC Input/Output Units and other auxiliary devices.

Time-shared (simultaneous) operation is possible because the Central Computer, the General Storage System, and each of the UFC Input/Output Units operate independently under control of a computer program. That is, time-shared operations can be programmed for General Storage or one of the UFC Input/Output Units; and the defined operation can be initiated without causing subsequent delay to the computer's execution of logical and arithmetic operations.

Once initiated, a time-shared General Storage or UFC Input/Output Unit operation is carried out independently of the Central Computer. It is, therefore, not uncommon in the execution of Univac File Computer System programs for

the Central Computer to be executing an arithmetic or logical operation,

General Storage to be engaged in a reading or recording operation, and

a variety (perhaps all) of the UFC Input/Output Units to be operating.

A medium-sized Model 1 Univac File Computer System is shown in Figure I-1. For convenience in identifying the Central Computer, General Storage, and the UFC Input/Output Units in this sketch,

the tops of the Central Computer Cabinets have been cross-hatched,

the tops of the General Storage Cabinets have been left plain, and

the tops of the UFC I/O Unit Cabinets have been shaded.

The purpose of this section, Section I , is to summarize the principal facts relative to each portion of the UFC Model 1 System. The information given in this section is therefore repeated and expanded, as appropriate, in succeeding sections.

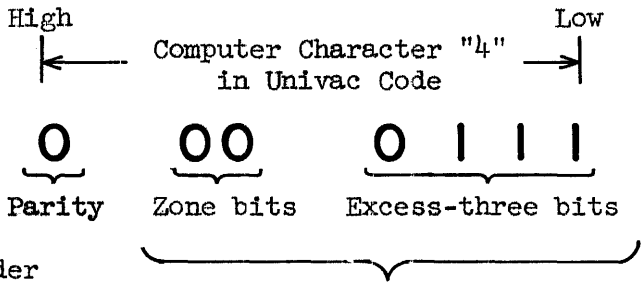
A. MODEL 1 CENTRAL COMPUTER

The Model 1 Central Computer is an alphanumeric, decimal, serial, three-address, general-purpose, digital machine that is equipped with an operating storage it can use independently of other memories in a UFC Model 1 System. In addition to carrying out the logical and arithmetic operations required in the solution of a problem, the Central Computer monitors the operation of the entire system.

1. Units of Data

a. Basic Unit

The basic unit of data is an alphanumeric character expressed in Univac Code. Each character is 7 bits (binary digits) in length:



The highest-order bit is a parity bit which is used principally for checking the accuracy of data-transmissions in the system

The lower-order 6 bits represent an actual alpha or numeric quantity in excess-three, binary-coded decimal notation (Univac code).

Table I-1 lists the Univac code for 63* computer characters. The digits 0-9 and the letters A-Z as well as many of the other characters listed have a system-wide meaning which corresponds exactly to the actual meaning of these symbols. Other symbols in the table have specialized meanings when employed in data sent to or received from certain UFC I/O Units. One example of an interpretation of these special symbols is listed, namely that for the UFC High Speed Printer.

Table I-1, Univac code

ZONE	EXCESS THREE BITS															
BITS	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	i	Δ	-	0	1	2	3	4	5	6	7	8	9	'	&	(
01	r	,	.	;	A	B	C	D	E	F	G	H	I	#	¢	@
10	t	"	|	)	J	K	L	M	N	O	P	Q	R	$	*	?
11	Σ	β	:	+	/	S	T	U	V	-W	X	Y	Z	%	=	not used*

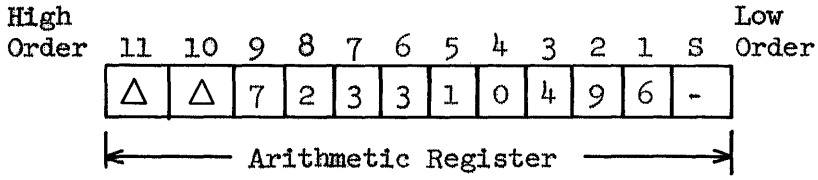
*111111 is a delete code for some UFC I/O Units; it is not used in the UFC High Speed Printer System.

UFC High Speed Printer Special Symbol Assignments:

i = Ignore code	" = Class Suppress 4
Δ = Space code, Class Suppress 1	? = Form Compensation III
r = Carriage return - multi-line symbol	¢ = Class Suppress 2
@ = Form Compensation I	Σ = Printer Stop
t = Tabulator, Class Suppress 3	β = Printer Breakpoint stop
/ = Form Compensation II	= = Form Compensation IV

b. Arithmetic Unit of Data

The arithmetic unit of data, or operand length, is 12 characters: 11 characters plus a Sign character:



Except in the multiply instructions, 12-character results are produced in all logical and arithmetic operations. (In the multiply instructions up to a 22-character product can be formed; however, only the 11 higher-order characters of the product and the Sign of the product, or the 11 lower-order characters of the product and the Sign of the product can be stored in the multiply instruction.)

c. Units of Storage

(1) Word

A Word, or 12 characters, is the standard unit of storage.

(2) Field

Quantities containing more or less than 12 (specifically, from 1 up to 119) contiguous characters are called Fields.

(3) Blockette

A 120-character quantity is called a Blockette.

The above represent the units of data associated with the following principal locations in the operating memory of the Central Computer:

High Speed Drum's Input/Output, Factor Storage, and Intermediate Storage Tracks;

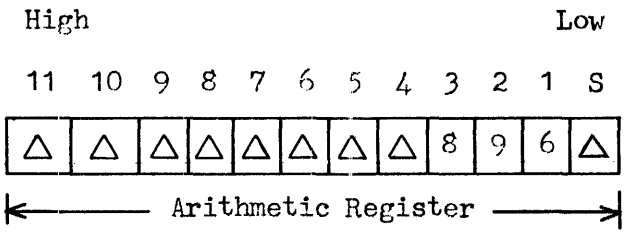
Block Transfer Buffer; and

General Storage Buffer.

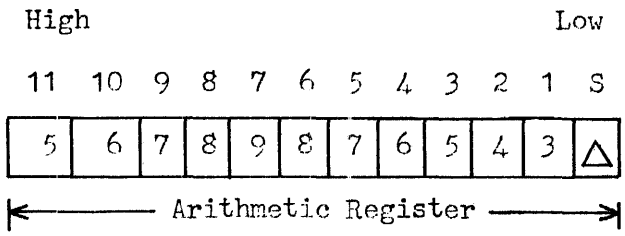
In each of these locations, 120 7-bit Univac coded characters can be stored and obtained on a Word, Field, or Blockette basis.

The Arithmetic registers (Register A, Register B, Register C and Register D) as well as several other special-purpose memory locations (as the Instruction Revolver & Shift Revolver) also store 12 7-bit characters. One memory location, the Code Distributor Register, stores but one 7-bit character. In two registers of the computer (the Program Address Counter and the General Storage Address Register) only the excess-three bits (or digit portion) of Univac coded characters are stored. When data is read out of PAK and GSAR the zone bits 00 and the appropriate parity bit are added, so that 7-bit characters are sent over the transfer Bus. In several other special-purpose memory locations (the General Storage and Block Transfer Buffer Patterns) only the parity bits of Univac coded characters are stored. There are, therefore, a variety of units of storage employed in the computer

If a stored number to be used as an operand contains less than 12 characters, as the Field 896+, Space codes are inserted as higher-order characters when the Field is shifted into an arithmetic register as an operand:

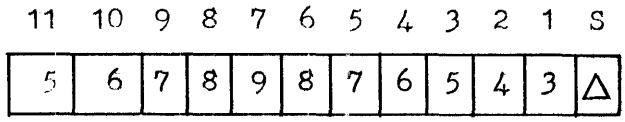


If a stored number to be used as an operand contains more than 12 characters, as the Field 123456789876543+, only the lower-order twelve characters are shifted into the Arithmetic Section:

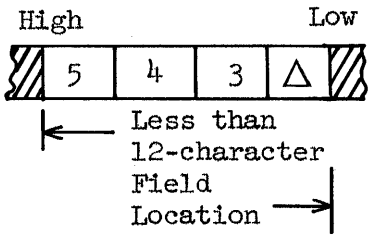


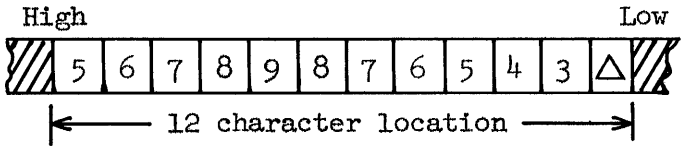
Similarly, when results are stored, the particular Arithmetic Register in which the result to be stored was formed, will make a 12-character quantity available for storage. However, what actually gets stored depends on the capacity of the location that is to receive the result:

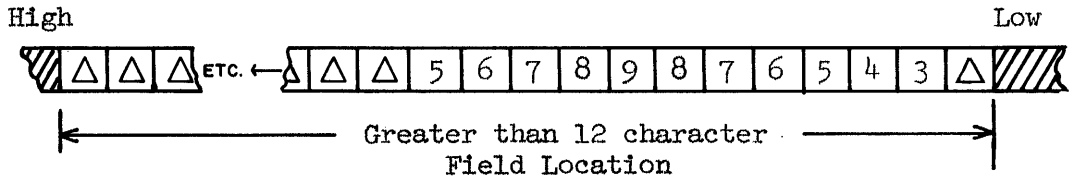
Assume this result is formed in an Arithmetic Register



When this result is stored, it is transmitted lowest-order character first; the number of characters transmitted depends on the capacity of the destination:







Complete details on the rules for all data transmissions in the Central Computer are given in Section III.

2. Three-address Instructions

Two types of Computer Instructions are available: Instruction Words (internally-stored instructions) and Program Steps (plugboard-defined instructions). Both employ three-address logic. That is, the format of both types of Computer Instruction permits specification of the location of two operands, of a particular process (or basic operation) to be performed on these operands, and of a destination at which the result of the instruction can be stored. Computer instructions can also be used to initiate other sub-operations. These operations are called Sub-Instructions or Sub-Steps, depending on whether they are defined by a coded character in an Instruction Word, or by patchcord wiring on the Program Control Plugboard, respectively. In general., Sub-Instructions and Sub-Steps enable the computer to

specially condition itself for the current operation or for subsequent operations; or

give General Storage an operation to perform; or

provide the basic control for the operation of any I/O Unit or receive control information from any I/O Unit.

The format for each type of instruction is explained below. Tables I-2 and I-3 list the name of each Process, Sub-Instruction, and Sub-Step used in the computer. (Complete details are presented in Section II.)

Computer programs can consist of (a) Instruction Words only, (b) Program Steps only, or as is usually the case (c) a combination of (a) and (b). Thus the Model 1 Univac File Computer can be operated as:

an internally-stored-program computer, or

a plugboard computer, or

a combination stored-program/plugboard computer

a. Instruction Words: internally-stored instructions defined by a 12-character word as follows:

U	V	W	OP
xxx	xxx	xxx	xxx
usually a 3-character address which specifies the location of the first value, V_1, used in the instruction	usually a 3-character address which specifies the location of the second Value, V_2, used in the instruction	usually a 3-character address which specifies the location where the result, R, of the instruction is to be stored	a 3-character operation code which specifies what the computer is to do in executing this instruction (basic operation or process, and Sub-Instructions)

Program Steps: plugboard-defined instructions, manually patched at a specific location (called a Plugboard Step 51-98) on the Program Control Plugboard. The following nine basic hubs are, in general, patched to other hubs to define a Program Step. (See Figure B-1, Appendix B). These hubs can be directly patched to their destination or they can be patched via other hubs which conditionally define a Program Step's process, addresses, and shifts, as well as which Program Step will be executed next in the program.

STEP IN hub	- patched from a pulse-source hub to initiate the Program Step
PROCESS hub	- patched to +C, ÷SR NC, etc. hubs to define the basic operation to be performed in executing the Program Step
V_1 ADDRESS hub	- patched to a hub which specifies the location of the first value, V_1, used in the Program Step
V_1 SHIFT hub	- patched to a hub which defines a shift operation for V_1, if V_1 must be shifted prior to its use in the Program Step
V_2 ADDRESS hub	- patched to a hub which specifies the location of the second value, V_2, used in the Program Step
V_2 SHIFT hub	- patched to a hub which defines a shift operation for V_2, if V_2 must be shifted prior to its use in the Program Step
R ADDRESS hub	- patched to a hub which specifies the location where the result, R, of the Program Step is to be stored
R SHIFT hub	- patched to a hub which defines a shift operation for R, if R must be shifted prior to its storage
STEP OUT hub	- patched to a variety of hubs to sequence program, initiate Sub-Steps (for the computer, General Storage System, or for I/O Control), transfer control to internally-stored program, Stop, etc.

Table I-2. List of UFC Model 1 Processes

Program Step Processes	Instruction Word Processes
Add	Add (W and W/O check)
Add and Check	Subtract (W and W/O check)
Subtract	Multiply, Store Lower (W and W/O check)
Subtract and Check	Multiply, Store Upper (W and W/O check)
Multiply, Store Lower	Divide, Store Quotient (W and W/O check)
Multiply, Store Lower and Check	Divide, Store Remainder (W and W/O check)
Multiply, Store Upper	Arithmetic Transfer
Multiply, Store Upper and Check	Buffer Transfer
Divide, Store Quotient	Mask Transfer
Divide, Store Quotient and Check	Suppress Left Zero
Divide, Store Remainder	Left Normalize
Divide, Store Remainder and Check	Load Shift
Compare	Compare
Arithmetic Transfer	Jump on Zero
Buffer Transfer	Jump on Plus
Mask Transfer	Jump on Negative
Suppress Left Zero	Unconditional Jump
Normalize	Channel Clear
Channel Clear	Load GSAR
	Test Demand In
	Demand In
	Test Incoming Control
	Channel Search Probe
	Substitute U
	Substitute V
	Substitute W
	- - - - - - - - - - - - - - - - - - - -
	Transfer Control to Plugboard (51-98)

Table I-3. List of UFC Model 1 Sub-Instructions and Sub-Steps

Sub-Steps	Sub-Instructions
Clear General Storage Buffer to Ignores	Clear General Storage Buffer to Ignores
Read Unit Record	Read Unit Record
Write Unit Record	Write Unit Record
Write Unit Record and Check	Write Unit Record and Check
Channel Search Equal	Channel Search Equal
Channel Search Unequal	Channel Search Unequal
Channel Search Probe	Special Character Out
Condition Compare	Breakpoint
Clear Block Transfer Buffer to Ignores	Stop
Branching	Suppress Check
Function Delay	Set Conditional Storage
Function Sequence	
Selector (Probe)	
Alternate Switch (Probe)	
Code Distributor Register Pulse (Probe)	
Test Demand In	
Demand In	
Track Switching	
Stop	
Next Instruction (Transfer Control to Internal Program)	
Test Incoming Control	

4. Principal Parts of the Central Computer

a. Program Control (See also Section II)

Program control is that section of the Central Computer which interprets, executes, and sequences Computer Instructions.

The principal Program Control circuits involved in the interpretation and execution of Computer Instructions are the

Operation Pulse/Enable Distributor, OED, which time-sequences the interpretation, execution, and sequencing of all Computer instructions;

Process Register, PR, whose contents in the case of Instruction Words, specifies the basic operation (or process) to be performed; and in the case of Program Steps, specifies which Plugboard Step (51-98) is being enabled, and hence which Program Step is being carried out;

Special Character Register, SR, which functions only in internally-stored programs and holds a coded character which defines the Sub-Instruction(s) to be initiated by an Instruction Word;

Shift Revolver, SRV, a special track on the High Speed Drum which holds a Shift Word that defines the type of shift (right, left, right end-around) and the number of positions (0-11) involved in shifting an operand and/or result during the execution of instructions (primarily used in internally-stored programs, but also available to plugboard defined programs); and the

Shift Counter, SK, a subtractive counter which

when a shift operation is defined by SRV, holds the appropriate section of the Shift Word from SRV during shift operations; and

when a shift operation is plugboard-defined, holds a number which specifies the type of shift and the number of shifts involved during a plugboard shift operation.

Each time a quantity is shifted one place, one is subtracted from SK. When SK = 0 the shift operation terminates.

The principal Program Control circuits involved in the acquisition and sequencing of Instruction Words are the

Program Address Counter, PAK, whose contents specify the location from which successive Instruction Words are to be obtained; and the

Instruction Revolver, IRV, a pair of revolvers on the High Speed Drum, one of which (IRVc.) holds the current Instruction Word, and

the other of which (IRVn) receives the next Instruction Word to be executed. Normally the next Instruction Word is obtained while the current Instruction Word is being executed.

PAK and IRV are not used in sequencing Program Steps. The sequence of Program Steps to be executed is determined by the programmer and (manually) patched on the Program Control Plugboard prior to a problem run. When one Program Step is completed, the pre-patched wiring defines which Program Step is executed next. The computer "remembers" what Program Step it is executing by retaining the Plugboard Step Number of the Program Step in the Process Register. When a new Program Step is initiated, the Plugboard Step Number (51-98) for that Program Step is inserted in the Process Register, etc.

b. Program Control Storage (See also Section III)

Program Control Storage is the operating memory of the Central Computer. It performs two general functions

(1) The acquisition of operands and storage of results during the execution of computer instructions (this function is similarly performed whether the program is internally or plugboard-defined).

(2) The acquisition of Instruction Words (this function performed only if stored-programs or combination stored-programs/plugboard-defined programs are being executed).

Program Control Storage consists of all those memory locations described below and the necessary circuitry for finding these locations and performing a storage reference.

The Principal Program Control Storage Locations are on the High Speed Drum:

2 Factor Storage (FS) Tracks 85 Intermediate Storage Tracks	each track can store 120 7-bit characters

The other Program Control Storage Locations, listed below, are "scratch pad" type memories, whose principal function is something other than permanent storage:

Input/Output (I/O) Tracks:	Central Computer buffers for data transmissions to and from the UFC I/O Units; 20 tracks on the High Speed Drum, two per addressable location: one used by the computer, one used by an I/O Unit assigned to that track location. Computer's track can be made available to I/O Unit and vice-versa by track switching operation. Capacity of each track is 120 7-bit characters.

Block Transfer Buffer, BTB: A 120-character magnetic core register used primarily as a buffer in Field and Blockette data transfers.

General Storage Buffer, GSB: A 120-character magnetic core register whose principal function is that of a buffer in data transmissions to and from the General Storage System. GSB not available to Program Control Storage when engaged in General Storage operations.

General Storage Address Register, GSAR: A 7-digit register used principally to hold General Storage Addresses during General Storage operations. Not available to Program Control Storage when engaged in General Storage operations.

Program Address Counter, PAK: A 3-digit register described above.

Code Distributor Register, CDR: A 1-character register used almost exclusively for program variance on the Program Control Plugboard.

Instruction Revolver (IRV): Actually 2 12-character revolvers on the High Speed Drum used principally to store the current and the next Instruction Word as described above.

Shift Revolver (SRV): A 12-character revolver used, as described above, to hold Shift Words.

Register A, RA:
Register B, RB:
Register C, RC:
Register D, RD:

} 12-character registers whose prime functions are performed during the execution of arithmetic and logical operations.

High Speed Drum Pattern:
Block Transfer Buffer Pattern:
General Storage Buffer Pattern:

} Special-purpose memory locations used to store a Field Selection Pattern which defines the fields that can be referred to in 120-character memory locations.

c. Arithmetic (See also Sections II and IV)

The Arithmetic section performs all of the arithmetic and most of the logical operations carried out in the system. Specifically it executes the process of the following instructions:

Add
Add and Check
Subtract
Subtract and Check
Multiply, Store Lower
Multiply, Store Lower and Check
Multiply, Store Upper
Multiply, Store Upper and Check
Divide, Store Quotient
Divide, Store Quotient and Check
Divide, Store Remainder
Divide, Store Remainder and Check
Compare
Mask Transfer
Suppress Left Zeros
(Left) Normalize

Arithmetic Transfer

Substitute U, V, and W

It also functions as an intermediate storage in the Jump Instruction Words.

The principal circuits in the Arithmetic section are:

The four 12-character registers

Register A
Register B
Register C
Register D

an adder/subtractor which serially (i.e. character-by-character) processes data in excess-three binary-coded decimal notation,

a comparator, and

the control circuitry and counters necessary to execute each of the above mentioned processes.

The Arithmetic operations are algebraic operations. Provision is made for psuedo-alpha and alphanumeric additions and subtractions.

d. I/O Control (See also Sections II and V)

The Input-Output Control circuitry of the UFC Model 1 Central Computer can monitor the simultaneous operation of from 1 up to 10 of any combination of UFC Input/Output Units. Except during those intervals in which control information is exchanged between the Central

Computer and an I/O Unit, the computer and I/O Unit operate independently. This is possible because part of the system's input/output circuitry is built into each UFC I/O Unit. This circuitry is called a Demand Station. When the I/O Control Instruction Words (internally-stored program) or the I/O Control Sub-Steps (plugboard program) are executed, the computer communicates with an I/O Unit's Demand Station. The computer can then test the status of an I/O Unit as well as exchange control information with that I/O Unit, without, in general, introducing any delay in Central Computer operation.

e. General Storage Control (See also Sections II and VI)

The Central Computer controls the operation of from 1 up to 10 large-capacity magnetic drums in the General Storage System by the execution of Sub-Instructions or Sub-Steps. Once a General Storage operation is initiated, it is carried out in General Storage completely independent of the Central Computer. The latter is therefore free for internal computing. Only one General Storage operation can be initiated and carried out at a time, but any General Storage Location, (or in the case of channel search operations, any series of locations) can be involved.

B. UFC MODEL 1 INPUT/OUTPUT SYSTEM (See also Section V)

The UFC Model 1 Input/Output System contains circuitry in both the Central Computer and each UFC I/O Unit. The computer circuits are those which execute the necessary computer operations to simultaneously control any combination of from 1 up to 10 UFC I/O Units. The principal circuitry contained in each UFC I/O Unit is called a Demand Station. By means of an I/O Unit's Demand Station, control information can be exchanged by the computer and a particular I/O Unit. The Demand Stations of all I/O Units are logically identical. Each I/O Unit also contains (a) the required control circuitry for interpreting the I/O Instructions from the computer that define what the I/O Unit is to do; (b) the necessary buffer memory and control circuitry for communicating with the I/O Unit's associated I/O Track on the High Speed Drum during data transmissions to and from the computer; and the control circuitry which activates the mechanisms of the I/O device and carries out an input/output operation completely independent of the central computer.

1. UFC Input/Output Units

The following UFC Input/Output Units are currently being built for use in UFC Model 1 Systems. A variety of other devices are under development.

UFC Model 1 Console System

UFC Inquiry Typewriter

UFC 90-Column Card System

UFC 90-Column Card System (With Post-Read Checking)

UFC 80-Column Card System (Bull Unit)

UFC Magnetic Tape Unit

UFC High Speed Printer

UFC High Speed Paper Tape System

A special-purpose, magnetic tape equipment, the UFC Sort-Collate System, is also available as an auxiliary device for use in UFC Model 1 Systems. This device permits off-line sorting and collating operations to be performed, and also allows combination collating and up-dating operations to be carried out in a psuedo on-line mode of operation.

2. UFC Input/Output Unit Operation

When a UFC Input/Output Unit is physically connected to the Computer, cabling from its Demand Station is plugged into one of ten Demand Station Positions (0-9) in the computer. Any UFC Input/Output Unit can be plugged into any Demand Station Position. Once plugged, however, it is designated as I/O Unit "b" (where b = 0-9), and it communicates only with a correspondingly numbered I/O Track (00-09) on the High Speed Drum in data transmissions.

Each UFC Input/Output Unit's Demand Station enables the computer to test the status of the Unit at any time to determine whether the Unit is READY or NOT READY for subsequent use. A UFC Input/Output Unit is READY if it is capable of receiving an I/O Instruction* from the computer. It is NOT READY in all other cases. By executing the appropriate Demand Test In (0-9) operation, the computer can immediately determine the status of a particular UFC Input/Output Unit. If the unit tested is NOT READY, the computer can resume its internal computation without delay. If the Unit tested is READY, the computer can then execute an appropriate Demand In (0-9) operation. This operation can (a) give the UFC Input/Output Unit a function to perform, or (b) receive (program variance) control information from the UFC Input/Output Unit, or in special cases, perform (a) and (b). In any case, no delay is introduced into central computer operation.

If a Unit is given a function to perform, it places itself in a NOT READY status, and begins the defined operation. If the operation specified by the computer program calls for input or output, the UFC Input/Output Unit then carries out the required data transmissions. In the case of input operations, data on an input medium is sent to the I/O Unit's buffer memory, and from there to the I/O Unit's associated I/O Track. In the case of output units the data flow is I/O Track → I/O Unit Buffer Memory → Output Medium. When a UFC Input/Output Unit is capable of receiving another I/O Instruction, it places itself in a READY condition. It can then, when again "demanded", receive another instruction from the computer, give control information to the computer, or both.

Each I/O Track on the High Speed Drum is actually a pair of tracks. At any given time at any of the ten I/O Track locations, the computer is connected

* An I/O Instruction is a specific operation the I/O Unit is to perform or an operating condition that is to be set up in the I/O Unit.

to one track of the pair, and the UFC Input/Output Unit is connected to the other track of the pair. The computer and UFC Input/Output Unit can thus time-share operations at the same I/O Track Location. While the UFC Input/Output Unit is loading or unloading one track, the computer can be loading or unloading the other. Track-switching circuitry in the UFC Input/Output Unit can be operated by the computer program to reverse the track assignments: the track formerly connected to the computer can be made available to the UFC Input/Output Unit; and, what was the UFC Input/Output Unit's track, can be made available to the computer. When the computer refers to an I/O Track location, it always communicates with the track (of the pair) to which it is connected at that time. When data transmissions to and/or from a UFC Input/Output Unit occur, they take place to and/or from the track to which the UFC Input/Output Unit is connected at that time. All data transmissions to and from each UFC Input/Output Unit take place independently of the Central Computer.

3. Principal Features of the Model 1 Input/Output System

a. Simultaneous, i.e., time-shared operation with the Central Computer and General Storage; also time-sharing of the same I/O Track by the Central Computer and the I/O Unit associated with that track.

b. Common Language versatility: each I/O Unit contains the necessary translators that enable it to transcribe or record one (or more) code(s) on the media it manipulates, and yet communicate with the computer only in Univac code. This completely eliminates the need for data conversion in the Central Computer.

c. Flexibility in the number and kind of UFC I/O Units to be included in the System. Any combination of from 1 up to 10 I/O Units can be employed; and the combination "on line" can be varied as required.

d. Facility for causing variance in the computer program either by interpreting special control data used in input media, or by interpreting special control information sent to the computer, apart from the actual data transmission operations.

e. Facility for assigning priority to Control information received from I/O Units.

f. Facility for scanning all I/O Units in a random or sequential fashion to keep every I/O Unit ready for operation in operation.

C. UFC MODEL 1 GENERAL STORAGE SYSTEM (See also Section VI)

Model 1 General Storage is the large-capacity, random-access, permanent memory of the UFC Model 1 System.

1. Units of Data

General Storage information is formated by two units of data: the Unit Record and the File. A Unit Record is a collection of

from 1 up to 10 computer words. That is, a Unit Record is a collection of contiguous characters each of which is at least 12 characters long, and each of which is an integral multiple of 12 characters in length (120 characters, maximum). A File is a collection of related Unit Records. Files are separated by End of File codes, 6 characters (----'0), which are stored on the General Storage Drum in special Search Control Locations.

The Unit Record Length for each operation is programmed. Files are also established by programming. Usually, a collection of related Unit Records are recorded from some predetermined starting location, and an End of File code is recorded in the next Search Control Location that appears after the last Unit Record of the File has been recorded.

2. General Storage Operations

Clear General Storage Buffer to Ignores
Write Unit Record
Write Unit Record and Check
Read Unit Record
Channel Search =
Channel Search ≠

3. Principal Features of UFC Model 1 General Storage System

a. A format for storing data which is similar (and, in many cases, identical) to the form of the business transaction. This basic, typical format is the individual Unit Record.

b. A capacity for storing internally thousands of Unit Records; the number of Unit Records and the length of each Unit Record being flexible. From 1 up to 10 General Storage Drums can be included in an installation, each drum capable of storing 180,000 alphanumeric characters. The Unit Record Length for each operation is programmed.

c. A random-access storage feature which permits the Model 1 Univac File Computer System to keep a current balance for a large number of records with high volume activity. Specifically, it allows

entry of input data into General Storage in the random sequence of its arrival; and

random-access to any Unit Record Area in General Storage, either to selectively obtain data from that Unit Record Area or to selectively alter data in that Unit Record Area on a Unit Record basis.

d. Two Search features which enable General Storage to look for a certain Unit Record in accordance with a key, or Unit Record Identifier, even though that Unit Record's address is not known. In Channel Search = an exact match between a Unit Record and a Unit Record Identifier is sought. In Channel Search ≠ a mis-match of any set of characters examined is sought.

e. Simultaneous operation with the Central Computer. Once given an operation to perform, Model 1 General Storage carries that operation out independently of the Central Computer.

II PROGRAM CONTROL

Program Control is that section of the Central Computer which interprets, executes and sequences Computer Instructions. It is therefore, the main control for the entire UFC Model I System.

The principal Program Control circuits are the:

Operation Pulse-Enable Distributor (OED)
Program Address Counter (PAK)
Instruction Revolver (IRV)
Process Register (PR)
Program Control Translator (PCT)
Special Character Register (SR)
Storage Address Register (SAR)
Transfer Address Control (TAC)
Shift Revolver (SRV), and
Shift Counter (SK)

Two types of Computer Instructions are interpreted, executed and sequenced by these circuits: Instruction Words (12-character coded quantities which are stored internally in the Computer), and Program Steps (plugboard-defined instructions which are manually patched on the Program Control Plugboard).

Each type of Computer Instruction involves a basic operation, as Add, Subtract, Multiply, etc., that the computer is to carry out; and each can also specify one or more auxiliary operations that modify or extend the basic operation.

The basic operation specified by a Computer Instruction is called a Process. The auxiliary operations are called Sub-Instructions or Sub-Steps, depending on whether they are specified by Instruction Words or Program Steps, respectively.

In the UFC Model 1, a computer program (or collection of Computer Instructions designed to achieve some computational result) can consist of:

Instruction Words only (internally-stored program)

Program Steps only (plugboard-defined program), or

a combination of Instruction Words and Program Steps
(combination internally-stored/plugboard-defined program)

Normally, the combination-type program is most efficient, because it enables the programmer to exploit the advantages of both types of program, and yet does not involve unusual or complicated programming techniques to use one type of program, then the other. That is, with but few exceptions, any function performed by the system can be defined either by an Instruction Word or by a Program Step; and it is a relatively simple matter to program certain portions of a problem using one type of Computer Instruction, and other portions of the problem using the other type.

Paragraph IIA below discusses internally-stored programs: Instruction Words, their format, their interpretation and execution, as well as the manner in which they are sequentially obtained in the computation of internally-stored programs.

Paragraph IIB below discusses plugboard-defined programs: the Program Control Plugboard, the definition of a Program Step, the interpretation, execution and sequencing of Program Steps, and the function of certain hubs which are used (or can be used) in all operations of the computer.

Paragraph IIC outlines the manner in which combination internally-stored/ plugboard-defined programs are carried out.

Paragraph IID is a detailed analysis, in tabular form, of every Computer Instruction.

A. INTERNALLY-STORED PROGRAMS

An internally-stored program is one in which appropriately coded Instruction Words are executed in the required sequence to achieve some computational result. Although actually initiated via patchcord wiring on the Program Control Plugboard, and although they can employ other plugboard-defined functions, internally-stored programs can completely define all the necessary system operations for the solution of any computable problem.

The principal advantages of the internally-stored program are these:

> the Instruction Words which compose it can be treated as operands by the computer (internally-stored programs can, therefore, be designed with practically unlimited versatility in self-modification); and
>
> either "loop" or "straight line"programming techniques can be applied, as convenient, since an adequate number of Jump Instruction Words are provided (for complex "loop" programming if this is desired), and the system's memory has been designed to allow programs of practically any size to be stored so as to be randomly accessible, even those involving a large number of instructions (as long "loop" programs, or "straight-line" programs).

1. Instruction Words

An Instruction Word is an internally-stored Computer Instruction consisting of 12 7-bit characters. All Instruction Words have the following format (x represents one character):

U	V	W	OP
xxx	xxx	xxx	xxx

where U, V and W are usually storage addresses, and OP is the Operation Code which defines what the computer is to do in executing the instruction.

a. Operation Code, OP

The Operation Code, OP, has two parts: the two leftmost characters define the basic operation the computer is to perform in carrying out the instruction; the rightmost character is a Special Character that defines an operation called a Sub-Instruction which extends or modifies the basic operation specified by the two leftmost characters.

The two leftmost characters can represent a range of numbers from 00 to 99. When these two characters specify one of the 26 numbers (< 50) that are listed in Table II-1, Page II-4, they are called a PROCESS code, PR. These numbers each define a basic arithmetic or logical operation for the computer. Each PR is listed in Table II-1 with both a numeric and a mnemonic code. The letters employed in the mnemonic code suggest the name of the Instruction Word and have the same "excess-three" bits in the Univac code as the numbers they represent. The numeric version of each PR is actually interpreted by circuitry in the computer; the mnemonic code is used in programming.

When the two leftmost characters of OP form a number > 50 but < 99 (i.e., are any of the integers 51-98) they are called a TRANSCOP (Transfer-Program Control-to-Plugboard) Code, TC, because when they are detected in the OP of an Instruction Word:

> the internally-stored program is interrupted; and
>
> Program Control is referred to a specific location on Program Control Plugboard where a Program Step, the next Instruction to be executed, is "located".

The specific location on the plugboard is called a Plugboard Step. TC and the Plugboard Step to which Program Control is referred are numerically the same; i.e., if the leftmost two characters of an Instruction Word's OP is 51, Program Control will be transferred to Plugboard Step #51 when that Instruction Word is executed. Transcop Instruction Words are thus a means of automatically switching Program Control from the internally-stored program to a plugboard-defined program. The sequence of Program Steps so initiated is called a Transcop sequence.

The rightmost or low-order character of all Operation Codes is called a Special Character, S. Each Special Character employed (see Table II-2, Page II-5) defines one or more Sub-Instructions which either

> modify the basic operation specified by an Instruction Word, or
>
> interrupt the internally-stored program (in a different manner than Transcop Instruction Words), or
>
> extend the activity of the system in executing an Instruction Word beyond the basic operation specified by PR or TC, and allow other operations to be initiated "between" Instruction Words.

Table II-I. PROCESS Codes

PR		Name of Process Instruction Word
Numeric Code	Mnemonic Code	
*13	AT	Arithmetic Transfer
*14	AD	Add
15	JN	Jump on Negative
17	JP	Jump on Plus
19	JZ	Jump on Zero
*22	SB	Subtract
*23	BT	Buffer Transfer
24	SU	Substitute U
25	SV	Substitute V
26	SW	Substitute W
27	SP	Channel Search Probe
*29	SZ	Suppress Left Zeros
31	LA	Load GSAR
32	LS	Load Shift
*33	CC	Channel Clear
34	TD	Test Demand In
*35	LN	Left Normalize
*37	CP	Compare
39	TI	Test Incoming Control
41	UJ	Unconditional Jump
*42	MK	Mask Transfer
*43	ML	Multiply, Store Lower
*44	MU	Multiply, Store Upper
45	DE	Demand In
*48	DQ	Divide, Store Quotient
*49	DR	Divide, Store Remainder

*These processes marked with an asterisk are also available in Program Steps by appropriately patching a Program Step's PROCESS hub as described in Paragraph IIB. The effect of most of the processes not marked with an asterisk above can be obtained in plugboard-defined programs without using a Computer Instruction; i.e., except for Load GSAR and Load Shift which can be duplicated during a Program Step, the effect of the processes that are not marked by an asterisk above is obtained "between" Program Steps in plugboard-defined programs.

Table II-2. Special Character Codes

ALTERNATE VALUES OF S	SPECIAL CHARACTER	BREAKPOINT #1	BREAKPOINT #2	BREAKPOINT #3	SET CONDITIONAL STORAGE	SUPPRESS CHECK	CLEAR GSB	READ UR	WRITE UR	CS =	CS ≠	WRITE UR AND CHECK	SPECIAL CHARACTER OUT	STOP
'	2	▨			▨									
&	3		▨		▨									
(	4			▨	▨									
i, Δ, 0	5	Note that 5, 0, Δ, or i can be used to ignore S in an IW												
	6	■												
-	7		■											
	8			■										
1	9				■									
#	B	▨			▨	▨								
¢	C		▨		▨	▨								
@	D			▨	▨	▨								
r	E					■								
SC ONLY ,	F	▨				▨								
.	G		▨			▨								
SC ONLY ;	H			▨		▨								
A	I				▨	▨								
$	K						■							
*	L							■						
?	M								■					
t	N									■				
"	O										■			
!	P											■		
‡	Q-Y												■	
/	Z													■

Note: Certain values of S specify but one Sub-Instruction; others specify two or three Sub-Instructions. For ready reference, each Sub-Instructions unique value of S is indicated by a shaded area, and cross-hatching is employed when two or three Sub-Instructions are specified by a value of S.

‡ The alternates for Q-Y are, respectively:),J,%,=, delete ,Σ,β,:, and +.

In summary, two basically different types of Instruction Words are specified by OP:

				OP
Process Instruction Words:	U	V	W	PRS
	xxx	xxx	xxx	xxx
Transcop Instruction Words:	U	V	W	TCS
	xxx	xxx	xxx	xxx

Complete details on each Computer Instruction are given in Paragraph IID. The first page of each instruction's analysis lists the sequence of events that occur when the instruction is executed, and describes in detail the basic operation specified by the instruction. The reader should examine these first pages at this point, and acquaint himself with the fundamental repertory of instructions available in the UFC Model 1. The following paragraph, particularly Tables II-3 and II-4, will also prove useful in understanding the purpose of each instruction. Paragraph IIAlc, Sub-Instructions, outlines the manner in which the system's basic repertory can be modified or extended.

b. U, V, W-sections

Since the instruction format of Univac File Computer Model 1 contains three address*-sections, the computer is said to employ three-address logic. In Instruction Words, the three address-sections are designated U, V, and W; in Program Steps, they are designated V_1 ADDRESS, V_2 ADDRESS, and R ADDRESS. The address-sections of an instruction are used in several ways.

In most instructions, the address-sections actually contain addresses, and these addresses are used to obtain the data to be used in the instruction and to store the data generated by the instruction. In other instructions, the instruction itself, or a portion thereof, is used as an operand. When the address-sections are used as operands, the section itself (rather than the data in the storage location the section might specify) is manipulated in the instruction. The specific uses of each address-section for each instruction are summarized in Table II-3, Pages II-8 through II-10. Additional comments on U, V, and W are made below.

In most Process Instruction Words, the U-section is a 3-character address that specifies the location from which the first operand, V_1, is to be obtained; the V-section is a 3-character address that specifies the location from which the second operand, V_2, is to be obtained; and the W-section is a 3-character address that specifies the location at which the result, R, is to be stored. As shown in Table II-4, Page II-11, V_1 is usually placed in RA; V_2 is placed in RB; and R is obtained from either RD or RC

* An address, as used herein, is a numeric or alphanumeric coded quantity that specifies a particular memory location in the computer.

In other Process Instruction Words, the U, V, and W-sections themselves are used as operands or as part of an operand contained in the Process Instruction Word. In these instructions, the section itself rather than data in any storage location it might specify is manipulated when the instruction is executed. The intermediate storages associated with U, V, and W, when these sections are so employed, are given in Table II-4, Page II-12.

The U, V and W-sections of Transcop Instruction Words are also each a three-character address; however, they are used differently. When Transcop Instruction Words are executed, a sequence of one or more Program Steps is initiated; and Program Control, beginning with the Program Step defined by TC, continues the program from the plugboard. Meanwhile, the U, V and W-sections, as well as the Special Character, of the Transcop Instruction Word are retained in Program Control. As explained in Paragraph IIB below, the address hubs (V_1 ADDRESS, V_2 ADDRESS, and R ADDRESS) of each Program Step in Transcop sequences can be patched to define an address either via the Plugboard Addressing System, or via the U, V or W-address of the Transcop Instruction Word. In short, the U, V and W-addresses of Transcop Instruction Words are available to any Program Step in Transcop plugboard sequences. When so used, U, V and W do not necessarily signify the locations from which V_1 and V_2 are obtained or the address of the location at which R is stored; e.g., the R ADDRESS hub on the plugboard could be wired to the U ADDRESS hub. In such a case, U of the Transcop Instruction Word would be used as the address of the location at which the result of a Program Step is stored.

The following general rules apply to the coding of U, V, and W when these sections are used as the addresses associated with V_1, V_2, and R:

the particular value given U, V, or W must be uniquely interpretable by Program Control Storage; and

the locations specified by U, V, and W must be valid locations for the process specified by the instruction

If either of these rules is violated in the coding of U, V, and W, the computer will hang up when it attempts to use the incorrect value of U, V, or W. The Program Control Storage Addresses which are interpretable are listed in Table III-1, Page III-3. The valid Sources (i.e., the locations that can be specified by U and V) and the valid Destinations (i.e., the locations that can be specified by W) for each Computer Instruction are listed under the rules for each Computer Instruction that are given in Paragraph IID.

Table II-3 Uses of U, V, and W

The following comments concerning the use of U, V, and W also apply to the addresses defined by patchcord wiring from the V_1 ADDRESS, V_2 ADDRESS, and R ADDRESS hubs, respectively, in all Program Steps which have the same process as the Instruction Words marked with an asterisk below.

	U	V	W
*(AT) Arithmetic Transfer *(BT) Buffer Transfer	specifies the Source of the data to be transferred	The V-section of this Instruction Word is ignored	specifies the Destination for the data transferred.

	U specifies the Source from which the following V_1 operands are obtained:	V specifies the Source from which the following V_2 operands are obtained:	W specifies the Destination at which the following results, R, are stored:
*(AD) Add, Add & Check	augend	addend	sum
*(SB) Subtract, Subtract& Check	minuend	subtrahend	difference
*(ML) Multiply, Store Lower Multiply, Store Lower & Check	multiplicand	multiplier	11 lower-order digits, and sign of product
*(MU) Multiply, Store Upper Multiply, Store Upper & Check	multiplicand	multiplier	11 upper-order digits and sign of product
*(DQ) Divide, Store Quotient Divide, Store Quotient & Check	dividend	divisor	quotient
*(DR) Divide, Store Remainder Divide, Store Remainder & Check	dividend	divisor	remainder

	U	V	W
(SU) Substitute U (SV) Substitute V (SW) Substitute W *(MK) Mask Transfer	specifies the Source of V_1, the operand to be modified	specifies the Source of V_2, the operand used in modifying V_1	specifies the Destination at which the modified V_1 operand is stored

*(CP) Compare: U and V specify the Sources for the two operands that are to be compared; the W-section is ignored.

	U	V	W
*(SZ) Suppress Left Zeros *(LN) Left Normalize	specifies the Source of V_1, the operand modified by these Instruction Words.	the V-section of these Instruction Words is ignored	specifies the Destination at which the modified V_1 operand is stored

*(CC) Channel Clear: the U and V-sections of this Instruction Word are ignored; the W-section is always a Blockette Address that specifies the particular buffer or track to be filled with Space codes.

Table II-3 Uses of U, V, and W (continued)

		U and V	W
(JN) Jump on Negative	If a Jump occurs:	U is used as a 3-character operand, and is usually* placed in the W-section of the Word location specified by V	is used as the address for the next Instruction Word
(JP) Jump on Plus			
(JZ) Jump on Zero			
	If no Jump occurs:	The U, V, and W-sections are ignored.	

(UJ) Unconditional Jump: U is used as a 3-character operand, and is usually* placed in the W-section of the Word Location specified by V; W is used as the address for the next Instruction Word.

(SP) Channel Search Probe:	U	V	W
Channel Search not completed	U is used as the address for the next Instruction Word	the V and W-sections are ignored	
Channel Search completed, and: Channel Search Storage = MINUS	the U-section is ignored	is used as the address for the next Instruction Word	the W-section is ignored
Channel Search Storage = ZERO	the U and V-sections are ignored		is used as the address for the next Instruction Word
Channel Search Storage = PLUS	the U, V, and W- sections are all ignored		

(LS) Load Shift: The U, V, and W-sections of this Instruction Word are called a Shift Word. They define the programmed shifts for V_1, V_2, and R, respectively, and are placed in the Shift Revolver when this Instruction Word is executed.

(LA) Load GSAR: The two lower-order digits of U, the three digits of V, and the two highest digits of W form A General Storage Address, and these digits are sent to GSAR when this Instruction Word is executed.

(51-98) Transcop: The U, V, and W-sections of Transcop Instruction Words are not used in the execution of those Instruction Words; however, they are available as V_1, V_2, or R addresses to Program Steps in Transcop sequences.

* Usually V refers to a Word Location in BTB, or GSB, or on the I/O, FS, IS Tracks. Conditional Jumps are frequently employed to enter a sub-routine via W and automatically return to the main program when the Instruction Word at V is executed. The following variation can be programmed for the manner in which U is stored:

if V is Register C, the jump occurs, but is non-returnable; i.e., no provision is made to return the program to any particular point.

Table II-3. Uses of U, V, and W (continued)

	U	V	W
(TD) Test Demand In	U=(-b-) and specifies which I/O Unit's Demand Station ("b") is tested	If I/O Unit "b" is READY, V is used as the address for the next Instruction Word and W is ignored.	
		If I/O Unit "b" is NOT READY, W is used as the address for the next Instruction Word and V is ignored.	

	U	V	W
(DE) Demand In	U= (abc) "a" is the conditional track-switch digit. "b" specifies the I/O Unit that is placed "on demand", "c" is the SPECIAL OUT control digit.	V= (xxx) and specifies an I/O Instruction. V is ignored if a SPECIAL OUT is produced and "c"≠ 1.	is the address used for the next Instruction Word if a SPECIAL OUT is produced. W is ignored if a DEMAND OUT is produced.

	U	V	W
(TI) Test Incoming Control	U= (abc) "a" is the conditional track switch digit "b" is the I/O Unit whose associated I/O Tracks are conditionally switched "c"= W, X, Y, or Z (the condition to be tested.)	the V-section of this Instruction Word is ignored.	is used as the address for the next Instruction Word if the (HS) I/O-to-Computer Control Line Storage condition (W,X,Y, or Z) tested is found. This section is ignored if the condition tested is not found.

Table II-4. Intermediate Storages Associated with U, V, and W.

RA = Register A
RB = Register B
RC = Register C
RD = Register D
BTB = Block Transfer Buffer

Instruction Words in which U and V specify the location of operands	Intermediate Storages		
	for V_1	for V_2	from which R is obtained
Add	RA	RB	RD
Add & Check	RA	RB	RD
Subtract	RA	RB	RD
Subtract & Check	RA	RB	RD
Multiply, Store Lower	RA	RB	RD
Multiply, Store Lower & Check	RA	RB	RD
Multiply, Store Upper	RA	RB	RC
Multiply, Store Upper & Check	RA	RB	RC
Divide, Store Quotient	RA & RC	RB	RD
Divide, Store Quotient & Check	RA & RC	RB	RD
Divide, Store Remainder	RA & RC	RB	RC
Divide, Store Remainder & Check	RA & RC	RB	RC
Arithmetic Transfer	RD	--	RD
Buffer Transfer	BTB	--	BTB
Mask Transfer	RA	RB	RD
Compare	RA	RB	--
Suppress Left Zeros	RA	--	RD
Left Normalize	RA	--	RD
*Substitute U, V, & W	RA	RB	RD
Channel Clear	Space Code Padder → W		

Table II-4 is continued on the next page.

* Except for the Substitute U, V, and W Instruction Words, the process specified by each Instruction Word on this page can be directly duplicated on the plugboard in a Program Step. The same intermediate storages are used for V_1 and V_2 in Program Step processes as in the correspondingly-named Instruction Word processes. Similarly, R is obtained from the same intermediate storages.

Table II-4. Intermediate Storages Associated with U, V, and W (Continued)

IRVc = IRV track containing the current IW
SRV = Shift Revolver
GSAR = General Storage Address Register
PAK = Program Address Counter
SAR = Storage Address Register
C-I/O (A-J) = Computer-to-I/O Control Lines (A-J)

Computer Instructions in which U, V, and W are operands or portions of an operand	Intermediate Storages associated with U, V, and W.
Load Shift	IRVc → SRV
Load GSAR	IRVc (7 characters) → GSAR
Jump on Plus Jump on Negative Jump on Zero	JUMP: U of IRVc → RC → W-section of V; W of IRVc → PAK NO JUMP: Ignore U, V and W
Unconditional Jump	U of IRVc → RC → W-section of V W of IRVc → PAK
Channel Search Probe	JUMP: U, V, or W of IRVc → PAK NO JUMP: Ignore U, V and W
Demand Test In	U of IRVc → SAR I/O Unit "b" READY: V of IRVc → PAK I/O Unit "b" NOT READY: W of IRVc → PAK
Demand In	U of IRVc → SAR DEMAND OUT: V of IRVc → SAR → C-I/O (A-J) SPECIAL OUT: (c = 1): V of IRVc → SAR → C-I/O (A-J); W of IRVc → PAK (c ≠ 1): W of IRVc → PAK
Test Incoming Control	U of IRVc → SAR Condition tested found: W of IRVc → PAK
Transcop	None as part of this IW. However, if the U, V, or W section of this IW is used by a Program Step, then an intermediate storage becomes involved, because an operand is to be obtained or a result is to be stored. The particular intermediate storage associated with U, V, or W can be found on Page II-11 by determining (a) which ADDRESS hub (V_1, V_2, or R) is patched to the U, V, or W ADDRESS hub; and (b) what process is defined in the Program Step.

c. Sub-Instructions

As explained on Page II-3, Sub-Instructions are operations, defined by specific values of S in the OP of Instruction Words, that extend or modify the basic operation specified by PR or TC. Sub-Instructions are of three general types:

Sub-Instructions for the Central Computer:
- Set Conditional Storage
- Suppress Check
- Stop

Sub-Instructions that initiate General Storage operations:
- Clear General Storage Buffer to Ignores
- Read Unit Record
- Write Unit Record
- Write Unit Record and Check
- Channel Search Equal
- Channel Search Unequal; and

Sub-Instructions that can produce a pulse on the Program Control Plugboard:
- Breakpoint 1
- Breakpoint 2
- Breakpoint 3
- Special Character Out (Q-Y)

The particular values or codes given S to initiate each Sub-Instruction are listed in Table II-2, Page II-5, and each Sub-Instruction is discussed in detail below. Because of the manner in which the Special Character is interpreted, most Sub-Instructions can be defined by a pair of alternate Special Characters. For simplicity only the principal values of S are mentioned below.

In general, the Sub-Instruction(s) specified by S is (are) interpreted the same regardless of the type of Instruction Word (PRS or TCS) by which it is initiated. However, Suppress Check is meaningful only in arithmetic Process Instruction Words, and the following should be noted regarding the time at which Sub-Instructions are initiated:

Process Instruction Words: When the Suppress Check Sub-Instruction is programmed, it is initiated and carried out during the following Process Instruction Words: Add, Subtract, Multiply Store Lower, Multiply Store Upper, Divide Store Quotient, and Divide Store Remainder. All other Sub-Instructions are initiated at the end of the execution of Process Instruction Words; i.e., just before Program Control begins execution of the next instruction.

Transcop Instruction Words: The Sub-Instruction(s) specified by S in Transcop Instruction Words, is (are) never carried out until the entire sequence of Program Steps initiated by the Transcop Instruction Word is

completed, and Program Control returns to the internally-stored program. That is, the entire Transcop sequence of plugboard operations is executed, and then (just prior to the execution of the next Instruction Word) the Sub-Instruction(s) specified by S in the Transcop Instruction Word is (are) initiated. If a Transcop sequence does not revert Program Control to the internally-stored program again, the Sub-Instruction of the Transcop Instruction Word is not initiated.

A Sub-Instruction may be an extremely short operation (as Set Conditional Storage) or it may be an extended sequence of events whose execution is time-shared with that of other operations in the system (as Channel Search Equal). Unless a Sub-Instruction by its very nature (as Stop, or one of the Breakpoints 1, 2, or 3 which result in a stop) interrupts the program, no delay in Central Computer operation is involved in initiating Sub-Instructions.

(1) Sub-Instructions for the Central Computer

(a) Set Conditional Storage.

Each time an arithmetic operation or comparison (of V_1 and V_2) operation* is executed, a special memory in the computer, called Branch Storage, is set to +, or 0, or -, depending on whether:

in arithmetic operations, the result of the operation is > 0, $= 0$, or < 0, respectively; or

in comparison operations, $V_1 > V_2$, $V_1 = V_2$, or $V_1 < V_2$, respectively.

Branch Storage can only be examined directly by plugboard-defined programs. Furthermore it is only a temporary memory for the result of arithmetic or comparison processes, since Branch Storage is always cleared (set to 0) just before the process of each instruction is initiated.

If the Set Conditional Storage Special Characters: 2, 3, 4, 9, B, C, D, or I are used in the OP of an Instruction Word, another memory in the computer, Conditional Storage, is set to +, 0, or -, depending on whether Branch Storage is +, 0, or -, respectively, at the time the Special Character of the Instruction Word is examined. (Note that S = 9 is the value used if only the Set Conditional Storage Sub-Instruction is desired.) Each setting of Conditional Storage lasts until another Set Conditional Storage Sub-Instruction is executed. It is thus a more permanent storage than Branch Storage, and it can be directly examined by the internally-stored program. (See the Jump on Plus, Jump on Zero, and Jump on Negative Instruction Words, Paragraph IID)

* Specifically, Instruction Words with the following processes: AD, SB, ML, MU, DQ, DR, or CP.

If a Process Instruction Word contains one of the above mentioned Special Characters and performs an arithmetic or comparison operation, the following will occur:

> if PR = AD, SB, ML, MU, DQ, or DR, Set Conditional Storage will be set to +, 0, or -, depending on whether the result of the arithmetic operation of that Process Instruction is > 0, $= 0$, or < 0, respectively;
>
> if PR = CP, Set Conditional Storage will be set to +, 0, or -, depending on whether $V_1 > V_2$, $V_1 = V_2$, or $V_1 < V_2$, respectively.

If an Instruction Word contains one of the Set Conditional Storage Special Characters, but does not specify an arithmetic or comparison operation as its process, 0 will be set up in Conditional Storage when the Instruction Word's Special Character is examined. In short, the Set Conditional Storage Sub-Instruction sets Conditional Storage to the current contents of Branch Storage, and Conditional Storage retains that setting until another Set Conditional Storage Sub-Instruction is executed.

(b) Suppress Check.

When Process Instruction Words involving an arithmetic operation are executed, the arithmetic operation specified is performed; and then unless S = B, C, D, E, F, G, H, or I, the result is always automatically checked by an inverse arithmetic operation. Addition and Subtraction are checked before the result is stored. Multiplication and Division are checked after the result is stored.* However, if the Special Character in a Process Instruction Word involving an arithmetic operation is B, C, D, E, F, G, H, or I, automatic checking of the arithmetic operation is suppressed. (Note that E is the Suppress Check Special Character to be used if Suppress Check is the only modification of the basic operation that is required.)

(c) Stop: If S = Z in the OP of an Instruction Word, the computer completes the execution of that Instruction Word, sets up the OP of the next Instruction Word in Program Control, and then stops. Computation of the internally-stored program can be resumed by pressing the START button.

(2) Sub-Instructions Which Initiate General Storage Operations.**

---

* If the arithmetic operation checks, Program Control continues execution of the program without delay. If the arithmetic operation does not check, the program is immediately interrupted and the ARITHMETIC ERROR CHECK hub on the plugboard emits a pulse. ARITHMETIC ERROR CHECK hub patching is discussed in Paragraph IIBln.

** Each of these Sub-Instructions is discussed in detail in Section VI, Pages VI-14 through VI-29.

(a) If S = K: Clear General Storage Buffer to Ignores.
(b) If S = L: Read Unit Record.
(c) If S = M: Write Unit Record.
(d) If S = P: Write Unit Record and Check.
(e) If S = N: Channel Search Equal.
(f) If S = O: Channel Search Unequal.

(3) Sub-Instructions which can produce a pulse on the Program Control Plugboard.

(a) Breakpoint 1, 2, and 3.

The Breakpoint Sub-Instructions are generally programmed when it is desirable to optionally interrupt a program at the end of any particular Instruction Word (i.e., stop the execution of Instruction Words) either to continue the program from the plugboard, or to stop the computer. The functions of Breakpoint are thus similar to those of the Transcop Instruction Word and the Stop Sub-Instruction, respectively. However, the Transcop Instruction Word and the Stop Sub-Instruction are always executed if detected by Program Control. Breakpoints 1, 2, and 3 on the other hand, even though coded, are conditionally initiated. That is, the following must be done, prior to a program run to initiate a Breakpoint Sub-Instruction:

I. Set the BREAKPOINT SELECTOR (on either Program Control Cabinet #1 or on the UFC Console), as follows, prior to the program run:

Press the BREAKPOINT 1 button to allow Breakpoint 1;
Press the BREAKPOINT 2 button to allow Breakpoint 2;
Press the BREAKPOINT 3 button to allow Breakpoint 3;

Any combination of Breakpoints can be allowed. The appropriate Sub-Instruction (Breakpoint 1, 2, or 3) will be initiated when the corresponding Breakpoint Special Character is detected. The particular Breakpoint(s) selected for use at any given time can be visually checked, because each button in the BREAKPOINT SELECTOR contains a light which illuminates when that button has been pressed.

II. Include the appropriate Special Character in OP of the Instruction Word after which the internally-stored program is to be interrupted:

2, 6, B, F for Breakpoint 1 (where 6 is employed if only Breakpoint 1 is to be specified by S)

3, 7, C, G for Breakpoint 2 (where 7 is employed if only Breakpoint 2 is to be specified by S)

4, 8, D, H for Breakpoint 3 (where 8 is employed if only Breakpoint 3 is to be specified by S)

Only one Breakpoint (1, 2, or 3) can be initiated per Instruction Word.

When Process Instruction Words which contain one of the Breakpoint Special Characters are executed and the BREAKPOINT SELECTOR is appropriately set, the program is interrupted at the end of the execution of these Process Instruction Words and the corresponding BREAKPOINT hub (1, 2, or 3) on the Program Control Plugboard will emit a pulse. When Transcop Instruction Words which contain one of the Breakpoint Special Characters are executed and the BREAKPOINT SELECTOR is appropriately set, the entire sequence of plugboard operations initiated by the Transcop Instruction Word is executed. Then the program is interrupted, and the corresponding BREAKPOINT hub (1, 2, or 3) on the Program Control Plugboard will emit a pulse.

III. Patch the BREAKPOINT 1, 2, and 3 hubs as required:

BREAKPOINT hub-to-STEP IN hub (to continue operation from plugboard)
BREAKPOINT hub-to-STOP hub (to stop computer operation)
BREAKPOINT hub-to-NEXT INSTRUCTION hub (to continue internally-stored program)

If any of these hubs emits but is not patched, the computer hangs up.

If the CLEAR button on the BREAKPOINT SELECTOR is pressed, all Breakpoints are ignored.

(b) Special Character Out (Q-Y):

If S = Q - Y in the OP of an Instruction Word, a pulse is emitted from the correspondingly-named SPECIAL CHARACTER OUT hub (Q-Y) on the Program Control Plugboard, just prior to the execution of the next Instruction Word. By appropriately patching the SPECIAL CHARACTER OUT hub which emits, this Sub-Instruction can be used to initiate an operation which is not directly available to the internally-stored program, as Condition Compare or Clear Block Transfer Buffer to Ignores (See Paragraph IIB1j).

Since this Sub-Instruction does not interrupt the internally-stored program, the SPECIAL CHARACTER OUT (Q-Y) hub which emits must not be patched to any STEP IN hub on the Program Control Plugboard, or to any other hub which results in a STEP IN hub being pulsed.

2. Interpretation, Execution and Sequencing of Instruction Words.

The following Program Control circuits are discussed below in explaining the interpretation, execution, and sequencing of Instruction Words:

Operation Pulse/Enable Distributor (OED)
Program Address Counter (PAK)
Instruction Revolver (IRVc and IRVn)
Process Register (PR)
Program Control Translator (PCT)
Special Character Register (SR)
Storage Address Register (SAR)
Transfer Address Control (TAC)
Shift Revolver (SRV)
Shift Counter (SK)

a. Operation Pulse/Enable Distributor, OED.

The Operation Pulse/Enable Distributor, OED time-sequences the execution of every Computer Instruction, and supplies the principal control pulses and/or enables required to carry out each part of an instruction.

OED operates in steps, and during the execution of most Instruction Words cycles either from 0-13 or from 2-13. At each OED setting one or more parts of the instruction are carried out.

A typical OED cycle for an Instruction Word is listed in Table II-5, Page II-19. In reading Table II-5, apply the following rule whereby OED operates: when the event specified by one OED step is initiated, OED is set to the next step; as each step is completed, the next is automatically begun and OED is set to the next-highest step, etc.

Table II-5, however, is merely a typical OED cycle. The specific sequence of OED steps that is carried out during the execution of each Instruction Word is listed in tabular form in Paragraph IID as the latter part of each Computer Instruction's analysis. The predicted* timing involved for each OED step in the execution of each Computer Instruction is also listed. These tables illustrate that not only do the process times (OED 8) for the various Instruction Words differ, but also (a) that different OED steps (as PAK → SAR and Load IRVn) require different times, and (b) that the same OED step (as PAK → SAR) may also require different times to carry out.

Each event listed in Table II-5 is discussed, as appropriate in the following paragraphs.

* These times have not all been machine checked. However, care has been taken in determining each time (or range of times) and it is not expected that the predicted and actual times will vary significantly.

OED	PRINCIPAL EVENTS	EXPLANATION
*0	PAK → SAR	Place this IW's address in SAR
*1	Load IRVn	Obtain this IW
2	Switch IRV's	Cause IRVn to become IRVc, and vice versa
	From OP of IRVc: Load PR Load SR	Set up OP of this IW in Program Control
3	U of IRVc → SAR	Place address for V_1 in SAR
	U of SRV → SK	Place shift for V_1 in SK
4	Load V_1	Obtain V_1 and place it in RA
5	V of IRVc → SAR	Place address for V_2 in SAR
	Shift V_1	Shift V_1 in RA
6	Load V_2	Obtain V_2 and place it in RB
	V of SRV → SK	Place shift for V_2 in SK
7	Advance PAK, PAK → SAR	Advance the count in PAK by one; then place next IW's address in SAR
	Shift V_2	Shift V_2 in RB
8	Load IRVn	Obtain the next IW
	W of SRV → SK	Place shift for R in SK
	Initiate Process	Carry out process (& check in Addition and Subtraction)
9	W of IRVc → SAR	Place address for R in SAR
	Shift R	Shift R in RD (or RC)
10	Store R	Store R
11	Check	Check (in Multiplication & Division)
12	--	(Used only in execution of Program Steps)
13	Sub-Instruction	Initiate Sub-Instructions

* If this Instruction Word is already loaded into IRVn, OED cycles from 2-13; if it is not, OED cycles from 0-13 (i.e., OED causes Program Control to first acquire this Instruction Word: OED 0-1; and then begin execution of this Instruction Word: OED 2-13.)

Table II-5. A Typical OED Cycle for an Instruction Word.

b. Program Address Counter, PAK*.

The Program Address Counter, PAK, is a three-digit, addressable (997) register that is used to obtain the address of each Instruction Word, and to sequence internally-stored programs. In short, PAK is the device used by Program Control to "remember" where to get the next (or in the case of a start, the first) Instruction Word in the program.

PAK receives each address it holds in one of the following ways:

(1) PAK can be manually set, prior to a program run, to any address in the range 000-999**. A MASTER CLEAR automatically sets PAK to 000. If the first Instruction Word in the program is to be obtained from any other location than 000, the correct address of the first Instruction Word must be set up in PAK after the MASTER CLEAR button is pressed.

(2) PAK will then normally be advanced by one (See OED 7, Table II-5) as each succeeding Instruction Word is executed; and the next Instruction Word will be acquired (on OED 8) during the execution of the current Instruction Word. In this type of operation, PAK acts like a counter and the successive addresses it stores will cause Program Control to obtain successive Instruction Words in the program from locations with consecutively-numbered addresses. The count in PAK (if uninterrupted) will then cycle as follows:

> From the number held in PAK at the time the first Instruction Word is obtained***, PAK is advanced by one as each Instruction Word is executed. If PAK is initially set to 130, for example, it will be advanced to 131 on OED 7 of the execution of the first Instruction Word (and the Instruction Word at 131 will be obtained on OED 8.) When the Instruction Word that was stored at 131 is executed, PAK is advanced to 132, etc. When, and if, the count in PAK reaches 979, PAK automatically resets to 130. (130-979 are the addresses of the Intermediate Storage Tracks on the High Speed Drum; it is in this area of the computer memory that Instruction Words are normally stored. PAK will continue to cycle through these addresses unless otherwise modified by the program as described below.)

---

* Reference is made in the following discussion to various addresses, such as 000, 130, 997, etc. See Table III-1, Page III-3 to identify the memory locations to which these addresses refer.

**Since PAK is a digit register, any of its digit positions can be set to 0-9. However, as explained later in this paragraph, the normal operating cycle for PAK is 130-979. Because PAK is a digit register, Field Addresses (each of which involves alpha) cannot be formed in PAK; hence Field Locations, as such cannot be used as the Source of an Instruction Word.

***This is generally the initial setting of PAK. It would not be the initial setting of PAK, however in combination-type programs if the program began on the plugboard and PAK was sent data (different from initial setting of PAK) by any Program Step in the initial plugboard portion of the program.

(3) PAK can also be sent data by the program in either of the following ways:

(a) When the Unconditional Jump or Test Demand In Instruction Words are executed; or when a "jump" occurs* in the execution of the following Instruction Words:

Channel Search Probe (and Channel Search Storage is not +)
Jump on Plus (and Conditional Storage is +)
Jump on Negative (and Conditional Storage is -)
Jump on Zero (and Conditional Storage is 0)
Demand In (and a SPECIAL OUT is produced)
Test Incoming Control (and the W, X, Y, or Z condition tested is found)

In each of these cases the U, V, or W-section of the Instruction Word being executed is sent to PAK on OED 5; PAK is not advanced on OED 7; but the correct next Instruction Word is obtained on OED 8 (i.e., it is acquired as the current Instruction Word is executed.) To verify this, see OED cycle for each of these Instruction Words in Paragraph IID.

The following caution must be observed in coding the address placed in PAK by these instructions: (i) Suppose PAK is loaded with 0'3, the next 1W will be obtained from the Word 3 location of the I/O Track address corresponding to the unit currently "on demand." (ii) As that 1W is executed, PAK is advanced to 0'4 and the next 1W is obtained from the word 4 location of the I/O track corresponding to the I/O Unit currently "on demand," etc. (iii) If PAK continues to be advanced in this manner, the 1W at 0'8 must be a jump or it must address PAK as a Destination, or the 1W at 0'9 must be a jump. In short, PAK must not be advanced when it contains 0'9, or an address which cannot be uniquely interpreted will be set up in SAR. (See page III-17.)

(b) When PAK is used as a Destination in an instruction, i.e., when W = 997 or R ADDRESS-PAK patching is employed, the result of the instruction is stored in PAK, provided PAK is a valid Destination for the process specified by the Instruction Word. When PAK is so used, the next Instruction Word's OED cycle will begin at OED 0, not OED 2. To see the need for this refer to Table II-5 and analyze the following:

On OED 7, PAK is advanced and the address of what normally would be the next Instruction Word is set up in SAR;

On OED 8, the "next Instruction Word" is obtained. This, however is not the correct next Instruction Word, since the program is sending an address to PAK on OED 10.

Therefore, when PAK is addressed as a Destination (OED 10 time) in any instruction, Program Control "remembers" this, and at the next OED 13 time sets OED to 0** rather than to 2, so that the correct next Instruction Word can be acquired.

---

* If a jump does not occur in these Instruction Words, the counting of PAK (and normally sequencing of the program) is uninterrupted.

** OED is also set to 0 when the internally-stored program starts (so that the first Instruction Word can be acquired.) In all other cases OED is set to 2.

PAK thus has a normal counting procedure for sequencing internally-stored programs and there are a variety of conditional and unconditional ways in which the sequencing of internally-stored programs can be changed.

Assuming no interruption, PAK's normal cycle is effective whenever the number in PAK is in the range 000-979.

If PAK is given a setting (manually or by the program) in the range 980-988, the following occurs:

assuming that each Word Location referred to in GSB contains a valid Instruction Word, the count in PAK will advance to 989, and the next Advance PAK will set PAK to 130.

If PAK is given a setting (manually or by the program) > 989, special care must be taken in the program. There are several cases to consider:

If set to 990, PAK can advance to 994 provided Registers A, B, C, and D (990-993), when referred to, each contains an interpretable Instruction Word.

994, however, is an illegal PAK address since a valid Instruction Word cannot be obtained from the Code Distributor Register. If 994 is generated in PAK by counting, or if 994 is manually set up in PAK, or if 994 is sent to PAK by the program, the Code Distributor Register will be referred to. However, the computer will not be able to interpret the "Instruction Word" it obtains. The computer, therefore, hangs up.

Basically, the same situation is created if 995 (address of the General Storage Address Register), or 997 (address of PAK itself) is set up or sent to PAK. Note that 995-999 cannot be generated in PAK by counting.

If 996 is set up or sent to PAK, IRVc is specified as the location from which the next Instruction Word is to be obtained. This will produce the same Instruction Word as the one being executed*, i.e., the current Instruction Word will be repeated.

If 998 (address of the Shift Revolver) or 999 (which is not assigned to any memory location) are set up or sent to PAK, the computer hangs up when it attempts to interpret these addresses. (998 is an illegal Source address, and 999 is uninterpretable ** in SAR.)

---

* As explained in Section III, the acquisition of the next Instruction Word and the procurement of operands are both storage references that refer to a Source. When IRV is addressed as a Source, IRV_C (the Instruction Revolver containing the current Instruction Word) supplies the data.

** If any characters other than those whose excess three bits are the same as the digits 0-9 are sent to any stage of PAK, the address so formed in PAK is also either uninterpretable in SAR or cannot be uniquely translated.

c. Instruction Revolver, IRV.

The Instruction Revolver, IRV, is a special-purpose device which makes use of a pair of tracks on the High Speed Drum to store and re-circulate the Instruction Word now being executed, and the Instruction Word to be executed next. IRV is thus the equivalent of a pair of high-speed (12-character capacity) memory locations. One revolver (IRVc) contains the current Instruction Word; the other revolver (IRVn) holds the next Instruction Word in the program. This pair of revolvers is referred to in the singular since both have the same address (996)*.

Program Control uses IRVc to obtain the U, V, W, and OP-sections of the current Instruction Word as it executes that Instruction Word. (See Table II-5, OED 2, 3, 5 and 9.) It uses IRVn to load in the next Instruction Word. (OED 8 of the current Instruction Word's OED cycle, or OED 1 of the next Instruction Word's cycle.)

The assignment of one revolver as IRVc and the other as IRVn changes as the execution of each new Instruction Word begins on OED 2: the revolver that was IRVn for the previous Instruction Word (and hence is the revolver that contains the Instruction Word to be executed next) becomes IRVc and supplies OP, U, V, and W to Program Control; the revolver that was IRVc for the previous Instruction Word becomes IRVn, and can be loaded with the next Instruction Word as the current Instruction Word is executed.

d. Process Register, PR.

Each time Program Control switches IRV's on OED 2 (i.e., causes IRVn to become IRVc and vice versa) the OP portion of the now current Instruction Word is set up in Program Control. Specifically,

> the excess-three bits of the two-highest-order characters in OP are sent to a two-digit register, called the Process Register, PR; and
>
> the value of the special character, S, is set up in a one-character register, called the Special Character Register, SR.

In internally-stored programs, the Process Register holds, therefore, a PR or TC code; and the Special Character Register stores S.

If the Process Register contains a valid PR or TC code (See Table II-7, Page II-25), it determines the OED cycle to be carried out in executing the Instruction Word. That is, the contents of the Process Register specify:

---

* If U or V = 996, or if V_1 ADDRESS or V_2 ADDRESS to IRV patching is employed, IRVc supplies the data. If W = 996, or if R ADDRESS-IRV patching is employed, IRVn receives the data to be stored. (The current Instruction Word can therefore be an operand for itself; and the next Instruction Word executed in a program can be the result of the current Instruction Word, without interrupting the count in PAK.)

the basic operation or process to be carried out in executing the Instruction Word

the intermediate storages to be used for the operands as well as those from which the result is to be obtained; and

the shifts which can occur in the execution of the Instruction Word.

If the Process Register contains a TC code (51-98) at OED 3 time, the internally-stored program is interrupted (after the completion of OED 3) and the execution of the Program Step at the Plugboard Step numerically equal to TC is initiated with OED set at 4. As explained in Paragraph IIB, the effect of a TC code in the Process Register at OED 3 time is the same as if that Program Step's STEP IN hub were pulsed. A Transcop sequence is then begun.

The manner in which the digits in the Process Register are interpreted is shown in Table II-6, Page II-25. The combinations of digits that produce valid Process and Transcop codes are listed in Table II-7, Page II-25.

e. Program Control Translator, PCT.

The Program Control Translator, PCT, is the integrating control circuitry in Program Control. PCT has the following principal functions:

PCT interprets the basic operation specified by an instruction and initiates that basic operation; or if a valid process or transcop is not specified, OED 3 is carried out, and PCT stops computer operations with OED set to 4.

PCT combines enables indicating what basic operation is to be performed with outputs from OED so that the events that must occur on each OED step of an instruction are carried out; and

PCT determines what intermediate storages are to be used for each process and sets up the necessary circuitry so that the required storage references involved in an instruction can be interpreted by Transfer Address Control.

In internally-stored programs PCT is notified of the basic operation to be performed via (combined) outputs from the Process Register. In plugboard-defined programs PCT is notified of the basic operation to be performed by the particular process hub (MK, +C, + NC, etc.) that is enabled when a Program Step's STEP IN hub is pulsed.

Table II-6. Translation of the Process Register

In both stages of the Process Register only the excess-three bits of characters are stored.		
Only digits are interpreted	These bits are translated to recognize each digit	The following will translate as a digit (all other characters are uninterpretable in PR)
0*	(--0011)	0, ;,), or +
1	(--0100)	1, A, J, or /
2	(--0101)	2, B, K, or S
3	(--0110)	3, C, L, or T
4	(--0111)	4, D, M, or U
5	(--1000)	5, E, N, or V
6	(--1001)	6, F, O, or W
7	(--1010)	7, G, P, or X
8	(--1011)	8, H, Q, or Y
9	(--1100)	9, I, R, or Z

* Interpretable only in lower-order stage.

Table II-7. Translation of Valid Process (and Transcop) Codes

The valid combinations of digits in PR (i.e., those which will enable a process to be performed) are listed below and each combination is identifiable by the mnemonic code associated with the process it specifies. If any but these combinations appear in PR, the computer stops with OED set to 4.

					LOW ORDER DIGIT IN PR									
					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
HIGH ORDER DIGIT IN PR	0	;	)	+										
	1	A	J	/				AT	AD	JN		JP		JZ
	2	B	K	S			SB	BT	SU	SV	SW	SP		SZ
	3	C	L	T		LA	LS	CC	TD	LN		CP		TI
	4	D	M	U		UJ	MK	ML	MU	DE			DQ	DR
	5	E	N	V		TC	TC	TC	TC	TC	TC	TC	TC	TC
	6	F	O	W	TC	TC	TC	TC	TC	TC	TC	TC	TC	TC
	7	G	P	X	TC	TC	TC	TC	TC	TC	TC	TC	TC	TC
	8	H	Q	Y	TC	TC	TC	TC	TC	TC	TC	TC	TC	TC
	9	I	R	Z	TC	TC	TC	TC	TC	TC	TC	TC	TC	

f. Special Character Register, SR.

The character held in the Special Character Register, SR, determines what Sub-Instruction(s) is (or are) to be initiated by an Instruction Word. As noted above, SR receives the Special Character from the OP-section of IRVc at OED 2 time. SR is usually examined at OED 13 time. However, during arithmetic Instruction Words, it is examined as follows:

Add and subtract, on OED 8
Multiply, on OED 11
Divide (examined on OED 7, check performed on OED 11)

The Sub-Instructions which can be initiated by the various values of S are listed in Table II-2, Page II-5. As shown in Table II-8, Page II-27, only certain combinations of bits in SR are interpreted. For this reason alternate values of S can be used to define most Sub-Instructions. Note from Table II-2 that each of the 64 characters in Univac code has been assigned a meaning in connection with S, and that 5, i, Δ, or 0 must be employed if no Sub-Instruction is to be initiated by an Instruction Word.

g. Storage Address Register, SAR

The Storage Address Register, SAR is a three-character (non-addressable) register which has the following principal functions:

SAR holds the addresses of the locations from which V_1 and V_2 are obtained (See Table II-5, OED 3 and 4 and OED 5 and 6 respectively); and the address of the location to which R is to be sent (OED 9 and 10);

SAR holds the address of the location from which the next Instruction Word is obtained during the storage reference that acquires each new Instruction Word (OED 7-8 of the current Instruction Word's OED cycle or OED 0-1 of the next Instruction Word's OED cycle);

SAR holds the U-section (abc) of the I/O Control Instruction Words:

Test Demand In (TD)
Demand In (DE)
Test Incoming Control (TI)

during OED 4 of the execution of these Instruction Words when a, b, and c are examined; and

SAR holds the V-section (xxx) of the Demand In Instruction Word (during OED 9 and 10) when an I/O Instruction is to be sent to the I/O Unit "on demand".

Section III discusses the use of SAR in the procurement of operands, storage of results, and acquisition of Instruction Words. The use of the SAR during the execution of the I/O Control Instruction Words

Table II-8 . Translation of Special Character Register*

To get this Sub-Instruction	These bits are translated	The following will translate to produce the required Sub-Instruction
Breakpoint #1	(0--101) or (0-1-01)	2, ', 6, B, #, F
Breakpoint #2	(0---10)	3, &, 7, -, C, ¢, G, or period
Breakpoint #3	(0-1-11) or (0--111)	4, (, 8, D, @, or H
Set Conditional Storage	(0--1--)	2, ', 3, &, 4, (, 9, 1 , B, #, C, ¢, D, @, A, or I
Suppress Check	(01 ----)	B, #, C, ¢, D, @, E, r, F, comma, G, ., H, ;, A or I
Clear GSB	(10-101)	K, or $
Read UR	(10-110)	L, or *
Write UR	(10-111)	M, or ?
CS =	(10-000)	N, or t
CS ≠	(10-001)	O, or "
Write UR & Check	(10-010)	P, or |
Special Character Out Q	(10-011)	Q, or)
" R	(10-100)	R, or J
" S	(11-101)	S, or %
" T	(11-110)	T, or =
" U	(11-111)	U or delete
" V	(11-000)	V, or Σ
" W	(11-001)	W, or β
" X	(11-010)	X, or :
" Y	(11-011)	Y, or +
Stop	(11-100)	Z, or /

* In Table II-2, Page II-5, the coding for S in OP is presented in a more useable form. This table is included to indicate how Table II-2 was formulated.

is discussed in Section V, and illustrated in Paragraph IID (OED cycles for I/O Control Instruction Words).

h. Transfer Address Control, TAC.

When a reference to memory for data is initiated, Transfer Address Control, TAC, on the basis of information received from PCT determines whether a valid Source or Destination is involved in the reference; and if a valid reference can be made carries out the required data transmission.

When TAC functions in programmed storage references (procurement of operands, acquisition of next Instruction Word, or storage of results), SAR holds the programmed address, and the address of the intermediate storage involved is automatically determined by PCT. In other storage references (as PAK → SAR, and U, V, or W of IRVc → SAR, and U, V, or W of SRV→ SK, etc.) PCT determines Source and Destination automatically as part of the execution of the instruction.

i. Programmable Shifts

Some shift operations in the computer occur automatically as part of the execution of an instruction; others must be programmed to occur.

An example of an automatic shifting operation is the manner in which most data transmissions take place to and from the various registers of the machine. When most registers in the computer receive (or supply) data, that data is shifted in (or shifted out), and the shifting operation is part of the mechanics of the data transfer. In these shifting operations, no (ultimate) re-arrangement of characters in the data transmitted takes place.

Certain shift operations, however, are not automatic; i.e., they occur only if programmed. Such shift operations always take place in the arithmetic registers (Registers A, B, C or D), and they have two principal functions, both of which involve re-arrangement of the data shifted:

> to align one operand's decimal point with that of another operand's so as to successfully perform an arithmetic operation (this is necessary because the machine treats all numbers as if they were integers, and would, for example, add "tenths" to "hundreds" if appropriate shifting of one or both operands is not programmed for these quantities prior to their use in an add operation.); or
>
> to permit <u>limited</u> "packing" and "unpacking" techniques to be employed in connection with Registers A, B, C, and D.

The following discussion deals only with shifts that must be programmed to occur.

The principal circuits involved in programmed shift operations are:

the particular register (RA, RB, RC, or RD) whose contents are to be shifted;

the Shift Revolver (SRV); and

the Shift Counter (SK).

SRV and SK are discussed in Paragraph IIA2j below. The following paragraphs are a general discussion of programmed shifts.

Table II-9, Page II-30, lists each of the instructions in which shifts can be programmed to occur, specifies what quantities in each instruction can be programmed for shifting, and indicates the particular register (RA, RB, RC, RD) in which these shifts take place when the instruction is executed. Note in Table II-9 that:

in most instructions in which shifts are permissible, three quantities can be shifted during the execution of the instruction:

V_1 (after it has been placed in RA, or in RA and RC, or in RD);
V_2 (after it has been placed in RB); and
R (before it is stored from RC or RD); and that

in certain instructions only two quantities (V_1 and V_2, or V_1 and R) can be shifted.

Three types of shift operations can be programmed for each quantity that can be shifted in an instruction:

right (end-off) shift
left (end-off) shift
right end around shift.

The effect of each of these types of shifts is illustrated in Figure II-1, Page II-31, and is discussed on Page II-32.

If shifts are programmed for any of the quantities in instructions not listed in Table II-9, the programmed shifts are simply ignored. Similarly if a shift is programmed for a quantity in one of the instructions listed in Table II-9, but a shift for that quantity is not listed as permissible in Table II-9, the programmed shift is ignored. (Table II-9, footnote** explains the only exceptions to this rule.)

In each type of shift, the maximum number of places the contents of the register(s) involved should be shifted is 11.

The time required for a shift operation, any type, is 0.042 (n+1) milliseconds, where n is the number of places shifted.

Table II-9. Instructions in which Shifts can be Programmed to Occur.

Computer Instructions in which the following Processes are executed:	Quantities for which Shifts can be Programmed	Arithmetic Registers Involved:	
Add	V_1, V_2, and R	V_1 in RA	
Add & Check			
Subtract			
Subtract & Check			
Mask Transfer		V_2 in RB	
Substitute U			
Substitute V			
Substitute W		R in RD	
Multiply Store Upper	V_1, V_2, and R*	V_1 in RA	
		V_2 in RB	
Multiply Store Lower		R in both RC & RD or in RC or RD*	
Multiply, Store Upper & Check	V_1 and V_2**	V_1 and RA	
Multiply, Store Lower & Check		V_2 in RB	
Divide, Store Quotient	V_1, V_2, and R	V_1 in RC***	
		V_2 in RB	
		R in RD	
Divide, Store Quotient & Check	V_1, V_2,**	V_1 in RC***	
		V_2 in RB	
Divide, Store Remainder	V_1, V_2, and R	V_1 in RC***	
		V_2 in RB	
		R in RC	
Divide, Store Remainder & Check	V_1, V_2**	V_1 in RC***	
		V_2 in RB	
Compare	V_1 and V_2	V_1 in RA	
		V_2 in RB	
Suppress Left Zeros	V_1 and R	V_1 in RA	
Left Normalize		R in RD	
Arithmetic Transfer	Three separate shifts can be programmed. The Source Data (or the Source Data plus Space codes) are manipulated.	All shifting done in RD	V_1 Shift time
			V_2 Shift time
			R Shift time

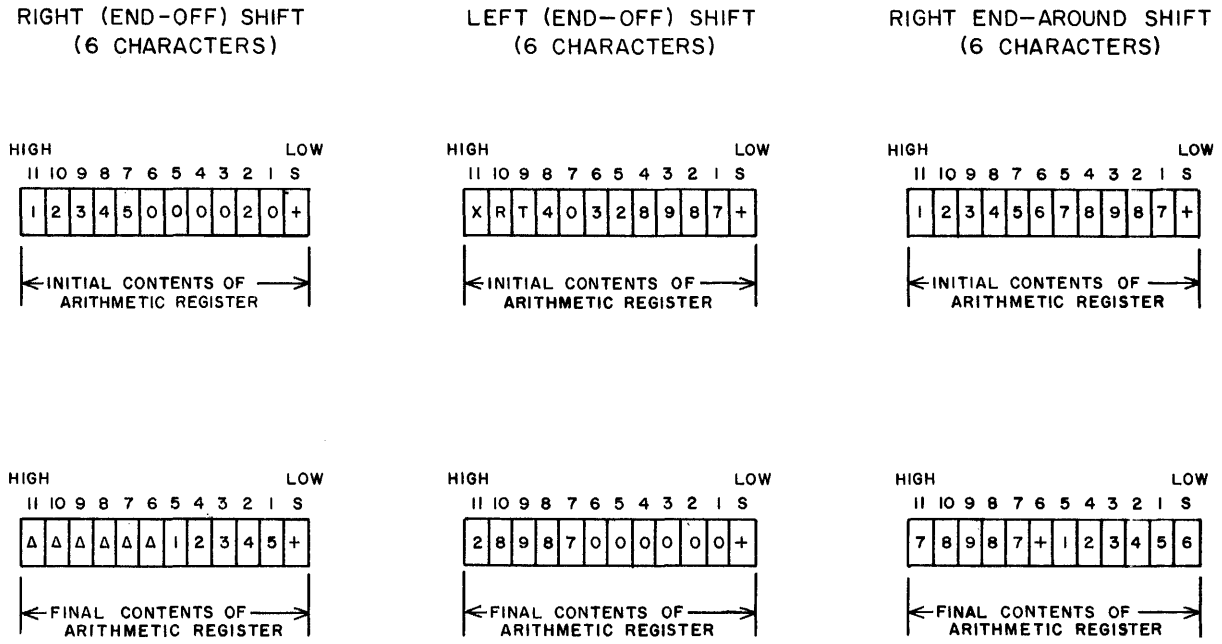
* In left and right shifts for R in Multiply processes, Registers C and D are shifted as a single (22-character) register. In right end around shifts for R in Multiply processes, however, only the register from which the result is stored is shifted.

** If a shift for R is programmed, the shift will be performed but will cause an ARITHMETIC CHECK ERROR.

*** V_1 is not actually shifted in RA, although the effect in the Divide algorithm is the same as if V_1 were also shifted in RA. The shift for V_1 is performed in RC to provide a correct remainder should the $(u-v+n+1)<1$ condition be detected at process time. (See DIVISION RULES, Paragraph IID)

Note: No shift can be programmed either by u,v, or w = 0, or by xx,yy, or zz = 00.

(a) All Shifts other than Right and Left Shifts for R in Multiply Processes.



(b) Left and Right Shifts for R in Multiply Processes.

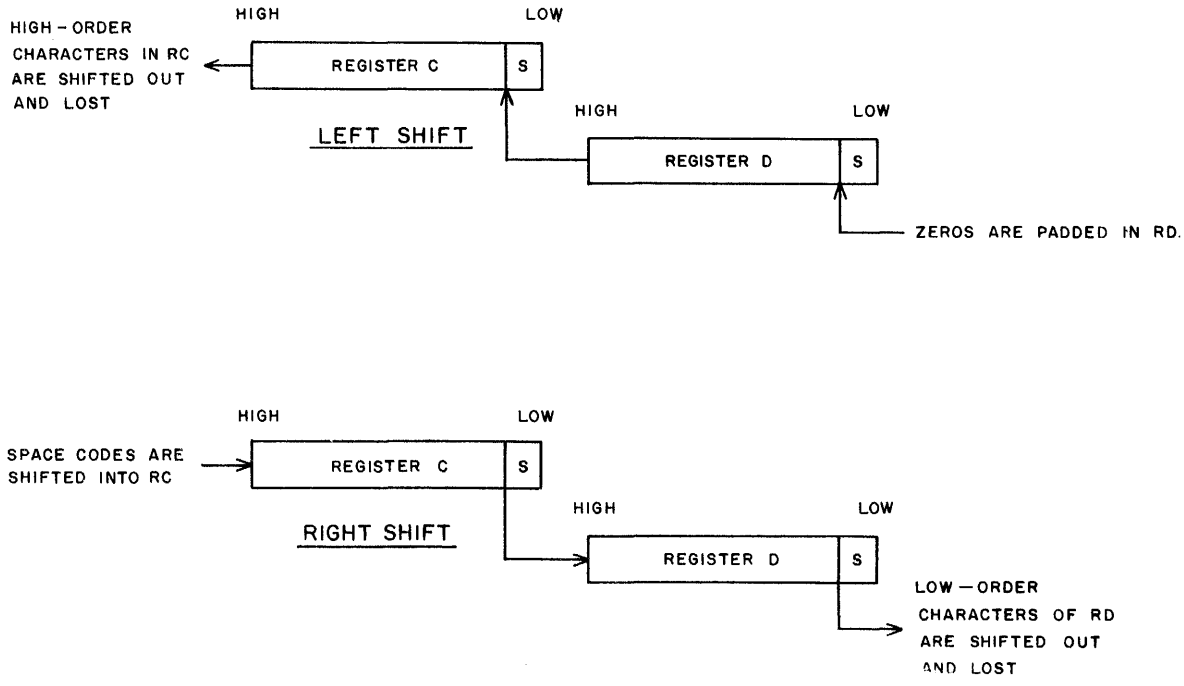


Figure II-1. Effects of Programmed Shifts

Figure II-1a (Page II-31) gives a numerical example of each of the three standard types of shifts, and illustrates the following basic facts that must be remembered in programming shifts:

Right (end-off) and left (end-off) shifts do not involve the register's Sign position, whereas the right end-around shift does;

During right (end-off) shifts, if "n" is the number of places a register is shifted, the lower-order "n" characters originally held in the register (Sign excluded) are lost, and "n" Space codes are shifted into the "n" higher-order character positions of the register;

During left (end-off) shifts, if "n" is the number of places a register is shifted, the higher-order "n" characters originally held in the register are lost, and "n" Zeros are shifted into the lower-order character positions of the register, beginning with the register's Character 1 position.

In Add and Subtract Instruction Words, care must be taken that V_1 and V_2 shifts are programmed so that correspondingly-significant digits are added or subtracted, and yet no sum or difference greater than 11 characters plus sign is generated. If a sum or difference greater than 11 digits and sign is generated, a +/- OVERFLOW error* will occur.

Instructions in which a Multiply process is carried out have special rules regarding shift operations:

V_1 and V_2 can be shifted as in any other instruction; however, if Multiply (Store Upper or Store Lower) and Check is programmed, a shift for R must not be programmed. If an R-shift is programmed, it will be performed, but a CHECK error* will occur since the verification of the result will be impossible.

If a Check is not programmed as part of the instruction, R can be shifted as follows:

right end-off shift (involving both RC and RD)
left end-off shift (involving both RC and RD)
right end-around shift (involving RC or RD)

When R is shifted right or left in Multiply instructions, RC and RD operate as one register (See Figure II-1b):

A left (end-off) shift will shift the eleven non-Sign positions of both RC and RD to the left. The output from the left side of RD will be entered into the right side (not the Sign position) of RC. Zeros will be padded into the right side (not the Sign position) of RD, and RC will be shifted off its left end.

* Errors are discussed in Paragraph IIB1n.

A right (end-off) shift will shift the eleven non-Sign positions of both RC and RD to the right. The output from the right end (not the Sign position) of RC will be entered into the left end of RD, Space codes will be entered into the left end of RC, and the output from the right end of RD (not the Sign position) will be shifted off the end.

When R is shifted right-end around in Multiply instructions only the register from which the result is to be stored is shifted.

Special rules must also be applied when programming shifts for V_1, V_2, and R in Divide instructions. Specifically,

V_1 and V_2 Shifts must be programmed to insure the proper number of decimal places in the quotient and yet not cause a DIVIDE OVERFLOW error (this error occurs when it is detected that a quotient > 11 characters and sign would be generated if a divide process were to be performed with the specified V_1 and V_2.) The equations which can be employed to properly program shifts for V_1 and V_2 in Divide instructions are given under DIVISION RULES in Paragraph IID.

In the Divide Store Quotient (or Store Remainder) and Check Instructions, no shift should be programmed for R. If a shift is programmed for R, it will be performed, but a CHECK error will occur since the verification of the result will be impossible.

j. Shift Word, Shift Revolver and Shift Counter

Programmed shifts are defined for the quantities involved in an Instruction Word by loading an appropriate Shift Word in the Shift Revolver prior to the execution of the Instruction Word.

(1) Shift Word

A Shift Word is a collection of 12 characters which specify what type of shift (left, right, right and around, or no shift) is to be performed on each quantity involved in an Instruction Word; and how many places in an arithmetic register each quantity in the Instruction Word is to be shifted. The following format is employed for Shift Words:

U	V	W	---
uxx	vyy	wzz	---

where:

the U-section (uxx) defines the shift for V_1:

u = 0	no shift	xx is the
u = 1	right end-around	number of
u = 2	left (end-off)	places V_1 is
u = 3	right (end-off)	to be shifted

the V-section (vyy) defines the shift for V_2:

v = 0	no shift	yy is the
v = 1	right end-around	number of
v = 2	left (end-off)	places V_2 is
v = 3	right (end-off)	to be shifted

the W-section (wzz) defines the shift for R:

w = 0	no shift	zz is the
w = 1	right end-around	number of
w = 2	left (end-off)	places R is
w = 3	right (end-off)	to be shifted

depending on how SRV is loaded, the lower-three order characters (---) of a Shift Word can contain:
the OP of the Load Shift Instruction Word, or
any computer characters (SRV loaded as a Destination);
in any event, these three characters are not used in shifting operations.

Shift Words are also used in plugboard-defined programs to define shifts for Program Steps. This subject is developed in Paragraph IIB.

(2) Shift Revolver, SRV

The Shift Revolver SRV, is a special-purpose 12-character memory location on the High Speed Drum. SRV is used to store (and constantly recirculate) the current Shift Word. Although provision is made for storing 12 characters in SRV, only the higher-order nine characters (the U, V, and W-sections of the Shift Word) are ever used.

SRV can be loaded with a Shift Word by the execution of an appropriately coded Load Shift Instruction Word (LS).

SRV can also be addressed as a Destination and sent a Shift Word as the result of an instruction (W = 998, or R ADDRESS-SRV patching).

Shift Words held in SRV cannot be read-out and sent to another Program Control Storage location, since SRV can function as a memory location only as a Destination. However, SRV is automatically referred to (as a Source) by Program Control to perform the U, V, and W of SRV→SK transmissions that occur during the execution of Computer Instructions.

NOTE: Care must be taken that SRV is loaded at the proper times in each program. The rules for doing this are outlined in Paragraph IID, 'SHIFT REVOLVER RULES.

(3) Shift Counter, SK

The Shift Counter, SK is a three-stage (non-addressable) register whose higher-order stage specifies the type of shift to be performed, and whose lower-order two stages store the number of places to be shifted. The lower-order two stages of SK function as a subtractive counter.

Table II-5, Page II-19 illustrates the different points in the OED cycle of an Instruction Word at which the U, V, and W-sections of the Shift Word in SRV are obtained, and placed in SK.

Note that to save time, the loading of SK, as well as the performance of the shift operations, is time-shared wherever possible with other events that occur in the execution of the instruction. The U-section is obtained first (OED 3). At V_1 Shift time (OED 5), SK is referred to by Program Control to determine the type of shift (no shift, right end off, left end off, or right end around) for V_1. If no shift has been specified, V_1 remains unaltered in RA (or RC or RD). If a shift is to occur, the particular type specified is initiated. Each time RA (or RC or RD) is shifted one place, the count (xx) in the lower-order-two digit positions of SK is decreased by one. When the count in SK = 0, the required shift is completed. At the same time that V_2 is being placed in RB (OED 6), the V-section (vyy)

of the Shift Word is placed in SK. While the next Instruction Word is being obtained (OED 7), the same sequence of events occur to shift V_2 as for V_1, except that RB, v, and yy are involved. Similarly, while the process specified by the Instruction Word is being carried out (OED 8), the W-section (wzz) of the Shift Word is placed in SK. When the process is completed, R is shifted. (R is shifted on OED 9, i.e., before it is stored).

The relationship between the various sections of Shift Words and what actually gets stored in SK is outlined in Table II-10, Page II-37.

Table II-10. Relationships between Shift Word and SK

Highest Order Stage of SK is translated to determine type of shift (This stage stores only the two lower-order bits of characters)			
These bits are translated	Type of shift	Permissible values for u, v, and w	
(----11)	no shift	0	;,), +, 4, D, M, U, 8, H, Q, Y, (, @, ?, or delete
(----00)	rt.end around	1	A, J, /, i, r, t, Σ, 5, E, N, V, 9, I, R, or Z
(----01)	left	2	B, K, S, 6, F, O, W, ', #, $, %, △, comma, ", or β
(----10)	right	3	C, L, T, 7, G, P, X, &, ¢, *, =, -, ., |, or :

Middle Order Stage of SK receives the tens digit of the shift amount (This stage stores only the two lower-order bits of characters)			
This stage can receive the following pairs of bits	Initial tens digit setting that is thereby defined for SK	x-, y-, or z- can be any* of the following characters	
(----11)	0	0	;,), +, 4, D, M, U, 8, H, Q, Y, (, @, ?, or delete
(----00)	1*	1	A, J, /, i, r, t, Σ, 5, E, N, V, 9, I, R, or Z
(----01)	2*		not used*
(----10)	3*		not used*

Lowest Order Stage of SK receives the units digit of the shift amount (This stage stores only the excess-three bits of characters)		
Only the excess-three bits of the digits 0-9 should be used	-x, -y, or -z can be any* of the following characters	
(--0011)	0*	;,), or +
(--0100)	1*	A, J, or /
(--0101)	2*	B, K, or S
(--0110)	3*	C, L, or T
(--0111)	4*	D, M, or U
(--1000)	5*	E, N, or V
(--1001)	6*	F, O, or W
(--1010)	7*	G, P, or X
(--1011)	8*	H, Q, or Y
(--1100)	9*	I, R, or Z

* If a shift amount (xx, yy, or zz) > 11 is programmed it will be performed and the following must be noted:

	Contents of register involved	
	Characters 1-11	Sign
Right Shift	all Space codes	undisturbed
Left Shift	all Zeros	undisturbed

The effect of all shift amounts > 11 can be produced with smaller values of xx, yy, or zz.

B. PLUGBOARD DEFINED PROGRAMS

A plugboard-defined program is one that is set up manually by appropriately patching various hubs on the Program Control Plugboard. It differs from an internally-stored program primarily in that the instructions, called Program Steps, which make it up are manually patched instructions (not stored, coded quantities.) When the computer executes a Program Step, therefore, it does not look for an OP, U, V, and W. It recognizes what to do (carry out a process, obtain operands, store results, sequence itself, etc.), because when the Program Step was defined, certain circuits that terminate in hubs on the Program Control Plugboard were connected by patchcords; and these circuits will automatically function when signalled to do so in an OED cycle.

Plugboard-defined programs have many useful features. Among these are:

the execution time of a plugboard-defined program is faster than the execution time of a corresponding internally-defined program. Specifically,

the shift operation OP of IRVc →PR and SR is not needed to set up the process and Sub-Steps to be performed. Similarly the U, V, and W of IRVc→SAR and the U, V, and W of SRV→SK transmissions are not required to set up addresses in SAR and shifts in SK unless the ADDRESS or SHIFT hubs are patched to U, V, and W ADDRESS or SHIFT hubs;

the access time of the next instruction is zero, unless a delay is programmed into the sequencing of Program Steps; and

the "loading" of a plugboard-defined program is a simple matter; the time required is merely that necessary to remove one plugboard and mount another.

a wide variety of program variance devices are available on the Program Control Plugboard. Any Program Step can, therefore, perform many functions in a given program, since each Program Step can be conditionally defined, and the times at which it is executed in the program can be conditionally determined. System-control programming as well as "decision" (or conditional jump) programming is particularly efficient in plugboard-defined programs. No instructions are required for these operations; they are performed between Program Steps. In plugboard operations direct variance of the system program is possible by using control data (received along with the actual input data) from an input device.

1. Program Control Plugboard

The Program Control Plugboard (Figure B-1, Appendix B) is a removable connection panel which is mounted on Program Control Cabinet #1 (Figure I-1, Page I-2). It serves as a terminal board (or collection of hubs) which a programmer can use to connect various circuits in the computer by patchcords. Depending on how these patchcords are manually wired, various

plugboard-defined programs can be designed. Each of the hubs on the Program Control Plugboard is discussed below in Paragraphs II-B1a through II-B1o, under one or more of the following subjects:

Program Step
Basic Program Step Patching
Plugboard Addressing System (Address-defining hubs)
Shift-Defining hubs
Selectors
Alternate Switches
Decision Elements for Enables
Sub-Step Patching (General Information)
Decision Elements for Pulses
Central Computer Sub-Step Patching
General Storage Sub-Step Patching
I/O Control Sub-Step Patching
Error hub Patching
Hubs which emit under control of the internally-stored program
Miscellaneous hubs

In Paragraph II-B1a through II-B1o, therefore, the purpose and proper use of each hub in the design of plugboard-defined programs is presented. Paragraph II-B1p lists the fundamental rules which must be observed to avoid improper use of patchcord wiring on the plugboard.

The interpretation, execution and sequencing of Program Steps is explained in Paragraph II-B2.

a. Program Step

A Program Step is a plugboard-defined Computer Instruction. The location of each Program Step is identified by a number 51-98, called a Plugboard Step. Each Plugboard Step contains nine basic hubs: STEP IN, PROCESS, V_1 ADDRESS, V_1 SHIFT, V_2 ADDRESS, V_2 SHIFT, R ADDRESS, R SHIFT, and STEP OUT. These hubs are patched to various other hubs on the plugboard to define a Program Step, to sequence plugboard-defined programs, to initiate General Storage or I/O Unit operations (and obtain control information from those portions of the system), to stop the computer, to specially condition the computer for subsequent operations, or to transfer Program Control to the internally-stored program. Direct patching can be employed from these hubs, or each plugboard function can be conditionally defined by patching via other hubs.

b. Basic Program Step Patching

The general rules for patching each of the nine basic hubs used in the definition of a Program Step are reviewed below.

STEP IN (pulse in):

When this hub receives a pulse, OED is set to 3, and a Program Step is initiated. The STEP IN hub can receive its input via patch-cord wiring from a variety of hubs. The pulse-sources for the signal sent a STEP IN hub are:

START hub	START-STEP IN patching will initiate the program with the Program Step whose STEP IN hub is pulsed. (By employing Alternate Switches and Selectors, described below, a number of conditional "starting points" can be defined on each Plugboard.)
STEP OUT hub of a Program Step	Direct STEP OUT-STEP IN patching unconditionally defines this Program Step as next in the program. STEP OUT-STEP IN patching via the (bussed) hubs associated with one or more non-decision Substeps will unconditionally define this Program Step as the next in the program and will initiate one or more non-decision Sub-Steps as Program Control passes from the previous Program Step to this Program Step. STEP OUT-STEP IN patching via the hubs associated with "decision" Sub-Steps can be employed to conditionally define this Program Step as the next in the program.
BREAKPOINT 1, 2, or 3 hubs	BREAKPOINT 1, 2, or 3 - STEP IN patching initiates a Breakpoint plugboard sequence when the BREAKPOINT 1, 2, or 3 hub emits. The plugboard operations begin with the Program Step whose STEP IN hub is pulsed.
STEP CLEAR OUT	STEP CLEAR OUT-STEP IN patching is usually employed to initiate an error-analysis plugboard sequence.

The above do not represent the only ways in which the STEP IN hub can be patched. As suggested above, each of the pulse sources can be indirectly patched (i.e., patched via other hubs) and a variety of combinations can be devised for the manner in which the STEP IN hub is pulsed.

When the execution of a Transcop Instruction Word is initiated, the effect is the same as if the STEP IN hub at the Plugboard Step numerically equal to TC were pulsed.

PROCESS (enable out):

The basic arithmetic or logical operation specified by a Program Step is called a process. There are 19 processes available on the plugboard and hubs are associated with each. These processes are the same as the correspondingly-named processes for Instruction Words.

Add (+ , NC hub)
Add and Check (+, C hub)
Subtract (-, NC hub)
Subtract and Check (-, C hub)
Multiply Store Lower (XSL, NC hub)
Multiply Store Lower and Check (XSL, C hub)
Multiply Store Upper (XSU, NC hub)
Multiply Store Upper and Check (XSU, C hub)
Divide Store Quotient (÷ SQ, NC hub)
Divide Store Quotient and Check (÷ SQ, C hub)
Divide Store Remainder (÷ SR, NC hub)
Divide Store Remainder and Check (÷ SR, C hub)
Compare (COMP)
Arithmetic Transfer (AT)
Buffer Transfer (BT)
Mask Transfer (MASK T)
Suppress Left Zeros (SLZ)
Left Normalize (NORM)
Channel Clear (CH CL)

When the PROC hub in a Program Step is patched to one of the above hubs, the corresponding process is carried out as the Program Step is executed. Actually the process specified by a Program Step is determined when that Program Step's STEP IN hub is pulsed (OED 3); the process is not initiated, however, until later (OED 8).

Patching the PROC hub of a Program Step to one of the above mentioned hubs is analogous to coding PR in the OP of a Process Instruction Word. Direct patching is employed if the process is to be unconditionally defined. Indirect patching, i.e., patching via decision-making devices for enables, is employed if the process is to be conditionally defined. By varying the process specified by a Program Step in accordance with certain conditions that exist in the program, a single Program Step can perform the function of several instructions at different points in the program. The decision-making devices for enables which permit this are discussed in Paragraph II-B1g below.

V_1 ADDRESS, V_2 ADDRESS, R ADDRESS (enable outs):

These hubs are used to define the addresses required in a Program Step. They are patched (directly or via decision-making devices for enables) to hubs in the Plugboard Addressing System (See Paragraph II-B1c). Patching these hubs in plugboard-defined programs is analogous to coding U, V, and W for Instruction Words in internally-stored programs.

The V_1 ADR hub is patched to define the address of the storage location of the first value, V_1, which the process is to manipulate. As shown in Table II-3, Page II-8, every Program Step's V_1 ADDRESS hub, except those in which the Channel Clear Process is specified, must be patched. (If the V_1 ADR hub is patched for the Channel Clear Process, the patching is ignored.)

The V_2 ADR hub is patched to define the address of the storage location of the second value, V_2, which most processes require. As noted in Table II-3, Page II-8, no V_2 is involved in Program Steps in which the following processes are specified: Arithmetic Transfer, Buffer Transfer, Suppress Left Zeros, Left Normalize, or Channel Clear. (If V_2 patching is employed in these Program Steps, the V_2 patching is ignored by the computer.)

The R ADR hub is used to define the storage location in which the result, R, of the process is to be stored. R ADDRESS patching is employed in all Program Steps except those in which the Compare process is specified. (In the Compare process, the "result" is set up in Branch Storage, not stored in an addressable memory location. If R ADDRESS patching is employed in the Compare process, therefore, it is ignored.)

Direct patching of these ADDRESS hubs to a hub in the Plugboard Addressing System unconditionally defines the address of a computer memory location. Indirect patching, i.e., patching via decision-making devices for enables (Paragraph II-B1g), permits the program to select the address of the memory location to be used from a group of possible locations, each of which is chosen only if certain conditions in the program exist.

The rules for patching the ADDRESS hubs of a Program Step are given in Paragraph II-B1c. The intermediate storages associated with the ADDRESS hub used in each process are listed in Table II-4, Page II-11.

V_1 SHIFT, V_2 SHIFT, R SHIFT (enable outs):

Each SHIFT hub of a Program Step is patched to a shift-defining hub (See Paragraph II-B1d) to define the programmed shift required for the correspondingly-named quantity in the Program Step. Patching these hubs in plugboard-defined programs is analogous to coding the U, V, and W sections of the Shift Words used in internally-stored programs.

In those Program Steps in which the operand V_1 is obtained and placed in Register A (See Notes 1, 2, 3, below) a shift operation can be programmed for V_1; i.e., the V_1 SH hub can be patched to a shift-defining hub to specify the type of shift and number of places V_1 is to be shifted in Register A prior to the use of V_1 in the process.

Notes (See Table II-9, Page II-30):

1. V_1 is placed in both Registers A and C in all Program Steps involving division. The programmed shift for V_1 in such processes is carried out only in RC.

2. V_1 is sent directly to BTB in Buffer Transfer processes, and cannot be shifted.

3. V_1 is sent directly to RD in Arithmetic Transfer processes, but a programmed shift at V_1 shift time for the Source data in RD is permissible.

In those Program Steps in which an operand V_2 is obtained and placed in Register B, (See note 4 below), a shift operation can be programmed for V_2; and the V_2 SH hub can be patched to a shift-defining hub to define the type of shift and number of places V_2 is to be shifted in Register B prior to the use of V_2 in the process.

Note 4. If a process involves no V_2, any programmed V_2 shift is simply ignored, except in the Arithmetic Transfer process. A V_2 shift is permissible in the Arithmetic Transfer process, but RD, not RB, is involved.

In most Program Steps, the result of a process can be shifted prior to its storage in the memory. When it is permissible to shift R (See Table II-9, Page II-30), the R SH hub can be patched to a shift-defining hub to define the type of shift and number of places R is shifted prior to its storage.

Direct patching of these SHIFT hubs to a shift-defining hub (Paragraph II-B1d) unconditionally defines the shift to be used for the correspondingly-named quantity in the Program Step. Indirect patching, i.e., patching via decision-making devices for enables (Paragraph II-B1g) permits the program to select a shift from a number of possibilities, each of which is chosen only if certain conditions in the program exist.

STEP OUT (pulse out):

A pulse from the STEP OUT hub of a Program Step signifies the completion of the basic process defined by the Program Step. This hub must always be patched to some other hub or hubs.* The hub or hubs to which the STEP OUT is patched depends on what is desired next in the program.

If the plugboard-defined program is to continue, patching must be provided so that the STEP OUT pulse will directly or ultimately cause the STEP IN hub of another Program Step to be pulsed. Any Program Step's STEP IN hub, including that of the Program Step whose STEP OUT hub emits, can be pulsed. The more complex the system program, the less direct STEP OUT-STEP IN patching is employed, since at STEP OUT time not only is the central computer's program to continue, but a variety of Sub-Steps, as those listed below can be initiated via patchcord wiring:

* If the STEP OUT hub is not patched (or if a PROCESS is not patched, or if an ADDRESS that must be patched is not patched) computer operation stops, since the pulse (or enable condition) necessary to continue operations is lost.

Non-Decision Sub-Steps:

Sub-Steps (as Condition Compare and Clear BTB to Ignores or those having to do with the operation of Selectors) which specially condition the central computer for certain (subsequent) operations;

Sub-Steps which initiate General Storage operations (as Read Unit Record, Write Unit Record, etc.)

Sub-Steps (as Track Switch, or certain Demand In Sub-Steps, etc.), which provide for input/output management or are needed to subsequently refer to the I/O Track associated with a particular I/O Unit.

Decision-Making Sub-Steps:

Sub-Steps (as Branch, Function Sequence, Demand Test In, certain Demand In Sub-Steps, Channel Search Probe, CDR Pulse In, etc.) that make a selection as to what is to be done next in the program on the basis of certain conditions existing in the system, or on the basis of control data received from an input device.

If the plugboard-defined program is to stop in such a manner that the plugboard-defined program can be resumed merely by pressing the START button, the STEP OUT hub must be "Y-wired"* to a STOP and to a STEP IN hub. When the STEP OUT hub emits, OED is set to 13, and the pulse entering the STOP hub will condition Program Control to Stop. The pulse entering the STEP IN hub will set the Process Register to the Plugboard Step Number (51-98) of the Program Step whose STEP IN hub is pulsed, and will reset OED to 3. Program Control can thus "remember" where it stopped on the plugboard, and resume operation from that point when, and if, the START button is pressed. If STEP OUT is patched only to STOP, the next Instruction Word is set up in Program Control before computer operations stop. When the START button is pressed after a Stop of this type, that Instruction Word is executed. (A plugboard Stop of this type can thus be used to transfer Program Control to the internally-stored program and then stop operation.)

Normally Program Control is transferred to the internally-stored program in plugboard operations by patching the STEP OUT hub to a NEXT INSTRUCTION (NI) hub. When an NI hub is pulsed, Program Control automatically transfers to the internally-stored program and begins execution of Instruction Words. As in the case of all pulse routing, many possibilities exist for the manner in which the STEP OUT pulse can result in an NI hub being pulsed.

Note that STEP IN and STEP OUT are pulse hubs. STEP IN receives a pulse; STEP OUT emits a pulse. The other seven hubs are enable out hubs.

* Y-wiring is actually accomplished by patching an out hub to one of the BUS hubs of a bus. Two other BUS hubs of the bus are then patched to the two ins that are to be signalled.

The time at which the STEP IN hub receives a pulse is determined by the manner in which it is patched; for example, if patched from the START hub, the STEP IN hub receives a signal when the MASTER CLEAR and START buttons are pressed; if patched directly or indirectly from the STEP OUT hub of another Program Step, it is not pulsed until after that Program Step's STEP OUT hub emits, etc. In any event, when the STEP IN hub of a Program Step is pulsed, OED is set to 3 and the execution of that Program Step begins. The times at which the other hubs are used during the Program Step's OED cycle are listed below. These times can be verified by examining any of the OED cycles for Program Steps in Paragraph II-D.

V_1 ADDRESS	OED 3
V_1 SHIFT	OED 3
V_2 ADDRESS	OED 5
V_2 SHIFT	OED 6
R SHIFT	OED 8
PROCESS (initiate)	OED 8
R ADDRESS	OED 9
STEP OUT	OED 12

CHAIN WIRING:

The PROCESS, V_1 ADDRESS, V_2 ADDRESS, R ADDRESS, V_1 SHIFT, V_2 SHIFT, and R SHIFT hubs of a Program Step are each bussed pairs. The destination hubs to which these hubs are patched and which these hubs (ultimately) enable are also bussed. When the same process (address or shift) is specified by several Program Steps, the PROCESS (ADDRESS or SHIFT) hubs of those steps can be patched together so as to form a bussed "chain"; and one of the PROCESS (ADDRESS or SHIFT) hubs at either end of the "chain" can be patched to the commonly specified destination. If this is done, none of the regular buses need be employed to tie several of the same type enable outs of Program Steps to a common destination, and the amount of patchcord wiring required is greatly reduced.

The PROCESS, ADDRESS and SHIFT hubs are bussed pairs to permit such "chains" between Program Steps to be formed; the address-destination hubs and the right, left, and right end around shift-destination hubs occur in 3-hub groups to permit separate "chains" for the common V_1 ADDRESSES (SHIFTS), the common V_2 ADDRESSES (SHIFTS), and the common R ADDRESSES (SHIFTS) to be formed and be readily identifiable, if this is desirable. (A "chain" to these destination hubs need not involve the same address hub (or shift hub) in each Program Step, however.) There are two sets of 2-hub groups for the process-destination and the U, V, and W shift-destination hubs. The CDR GROUP IN and CDR ALPHANUMERIC IN hubs are bussed in 4-hub groups. Four separate "chains" can therefore be formed to each of these hubs without employing regular buses.

If the regular buses are employed, of course, more than 3 or 4 "chains" can be formed to each destination hub.

c. Plugboard Addressing System

The Plugboard Addressing System is the collection of all those hubs to which the V_1 ADDRESS, V_2 ADDRESS, and R ADDRESS hubs of a Program Step can be patched to define a programmed storage reference from the plugboard. The names of these hubs identify the particular memory location in the computer that is referred to when these hubs are enabled via patchcord wiring from the ADDRESS hubs in a Program Step. Each hub in the Plugboard Addressing System is listed in Table II-8, Page II-47, together with the address these hubs set up in SAR when enabled.

As in the case of Instruction Words, the following general rules apply to addresses defined in a Program Step:

1. The address set up in SAR by patching the V_1, V_2, or R ADDRESS hubs to a hub in the Plugboard Addressing System must be interpretable.

When the V_1, V_2 or R ADDRESS hubs are patched to any of the hubs in the Plugboard Addressing System, except the U, V, W ADDRESS and I/O WORD (0-9),I/O FIELD (A-V) and 1/O-Z hubs, no problem exists. When such hubs in the Plugboard Addressing System are enabled, they automatically set SAR to an interpretable address.

If the V_1, V_2, and R ADDRESS hubs are to be patched to I/O WORD (0-9), I/O FIELD (A-V) or I/O-Z hubs, the I/O Unit associated with the I/O Track to be referred to must previously have been placed "on demand". (If this is not done, and no I/O Unit is "on demand", the computer will hang up when it attempts the storage reference initiated by enabling an I/O WORD (0/9), I/O FIELD (A-V) or I/O-Z hub. If the wrong I/O Unit is previously placed "on demand", an address will exist in SAR, and that address will be interpretable, but will not be correct because its associated I/O Track, not the one required, will be specified.)

If the V_1, V_2, or R ADDRESS hubs are patched to a U, V, or W ADDRESS hub, the corresponding U, V, or W-section of IRVc is shifted into SAR. These sections, therefore, must be coded so that an interpretable address (See Table III-1, Page III-3) ends up in SAR. If an uninterpretable address is coded for these sections, the computer will hang up when it attempts to use the uninterpretable address.

2. The address set up in SAR must be one that is permissible in the process defined by the Program Step. Each address in a Program Step specifies either a Source or Destination. The permissible Sources and Destinations for each process are given under the Rules for each instruction in Paragraph II-D.

As noted above, the V_1 ADDRESS, V_2 ADDRESS or R ADDRESS hubs of a Program Step can be patched directly to any hub in the Plugboard Addressing System that defines a valid Source or Destination for the process specified by the Program Step, or (if an address for an operand or a result is to be conditionally defined) they can be patched via other hubs as described in Paragraph II-B1g below. Employ chain-wiring wherever possible.

Table II-11. Plugboard Addressing System.

(See Table B-1, Appendix B, for the location of these hubs on the Program Control Plugboard, Figure B-1, Appendix B)

Hubs in Plugboard Addressing System which set SAR when enabled	Address to which SAR is set
RA (Register A)	990
RB (Register B)	991
RC (Register C)	992
RD (Register D)	993
GSAR (General Storage Address Register)	995
CDR (Code Distributor Register)	994
PAK (Program Address Counter)	997
Block Transfer Buffer Word 0-9	100-109
General Storage Buffer Word 0-9	980-989
Factor Storage Track #1 Word 0-9	110-119
Factor Storage Track #2 Word 0-9	120-129
Input/Output Track "on demand" Word 0-9	0b0-0b9*
BTP (Block Transfer Buffer Pattern)	99X
GSP (General Storage Buffer Pattern)	99Y
ISP (High Speed Drum Pattern)	99W
IRV (Instruction Revolver)	996
SRV (Shift Revolver)	998
BTB-Z (Block Transfer Buffer - entire contents)	10Z
GSB-Z (General Storage Buffer - entire contents)	98Z
FS #1-Z (Factor Storage Track #1 - entire contents)	11Z
FS #2-Z (Factor Storage Track #2 - entire contents)	12Z
I/O-Z (Input/Output Track "on demand"-entire contents)	0bZ*
Block Transfer Buffer Field A-V	10A-10V (not I and 0)
General Storage Buffer Field A-V	98A-98V (not I and 0)
Factor Storage Track #1 Field A-V	11A-11V (not I and 0)
Factor Storage Track #2 Field A-V	12A-12V (not I and 0)
Input/Output Track "on demand" Field A-V	0bA-0bV* (not I and 0)

Special hubs in Plugboard Addressing System	Resulting SAR address when those hubs are enabled
U Address	U of IRVc →SAR
V Address	V of IRVc →SAR
W Address	W of IRVc →SAR

*No particular I/O Track is specified by the I/O WORD or FIELD hubs. To refer to an I/O Track (Ob-to Ob-) via the Plugboard Addressing System, the I/O Unit (0-9) associated with the particular I/O Track desired must be placed "on demand". When the appropriate I/O WORD or FIELD hub is enabled, the highest-order stage of SAR is set to 0, the lowest-order stage of SAR is set to 0-9 or A-V, depending on the hub enabled, and a test of the demand status of each I/O Unit is made. The particular I/O Unit "on demand" then sets the middle stage of SAR to "b".

d. Plugboard Shift Patching

The shifts that can be programmed to occur in Program Steps are listed (on the basis of the process defined by the Program Step) in Table II-9, Page II-30. The same rules apply for plugboard-defined shifts as for shifts that occur in the internally-stored program. (See Paragraph II-A2i above.) However, plugboard shifts are defined differently for the computer.

Usually in Program Steps, the V_1 SHIFT, V_2 SHIFT and R SHIFT hubs are patched to the appropriate LEFT SHIFT (0-11), RIGHT SHIFT (0-11) or RIGHT END AROUND SHIFT (0-11) hubs to define the programmed shifts for V_1, V_2, and R, respectively. The time at which such plugboard-defined shifts occur and the manner in which these shifts are carried out is the same as that described above (Paragraph II-A2j) for Instruction Words. The basic difference is that when the above hubs are employed, the Shift Word in SRV is not used to define the shift operation. That is, the particular LEFT SHIFT (0-11), RIGHT SHIFT (0-11) or RIGHT END AROUND SHIFT (0-11) hub that is enabled sets SK when that hub is enabled.

If it is desirable to use the Shift Word in SRV to define a shift in a Program Step, the V_1 SHIFT, V_2 SHIFT or R SHIFT hubs of the Program Step must be patched to a U SHIFT, V SHIFT, or W SHIFT hub. The following will then occur, depending on the availability of the Shift Word in SRV:

> No Shift Word in SRV, or Shift Word in SRV not available* (i.e., as far as the program is concerned, SRV is in a "cleared" condition): In this case, no shifts will occur in the Program Step for the quantity whose SHIFT hub is plugged to a U, V, or W SHIFT hub.
>
> Shift Word in SRV is available*: In this case, the U, V, or W-section of the Shift Word (depending on whether the U SHIFT, V SHIFT, or W SHIFT hub is enabled) will be referred to, and will define the shift to be performed. For example, if the R SHIFT hub is patched to the U SHIFT hub and SRV contains an available Shift Word, uxx defines the shift for R. When the U SHIFT hub is enabled, U of SRV is shifted into SK. Any one of the shift hubs in a Program Step can be patched to either the U SHIFT, V SHIFT, or W SHIFT hubs.

The V_1 SHIFT, V_2 SHIFT, and R SHIFT hubs can be patched directly to the above-mentioned shift-defining hubs, or they can be patched to those hubs as described in Paragraph II-B1g below. Employ chain-wiring wherever possible.

* See SHIFT REVOLVER RULES , Paragraph II-D, for the times when SRV does and does not contain an available Shift Word.

e. Selectors

A Selector is a relay which is used as an electrically operated two-position (Select or Non-Select) switch. It allows a programmer to route pulses, enables, B+, or ground in one or the other of two directions.*

A Selector's "routing" (or decision-making) circuit terminates on the Program Control Plugboard in COMMON, SELECT, and NON-SELECT hubs. When a pulse or enable (or a B+ or ground supply) is applied via patchcord wiring to a Selector's COMMON hub, the SELECT or NON-SELECT hub will emit depending on how the Selector is set. Usually a Selector is set to Non-Select. The plugboard program must set it to Select and make provision to hold it in its Select state as long as required.

The control (or coil) circuit for operating a Selector terminates on the Program Control Plugboard in SELECTOR PICKUP and SELECTOR GROUND hubs. These hubs are numbered the same as the Selectors. A particular Selector is set to Select as long as (1) B+ is applied to its associated SELECTOR PICK-UP hub, and (2) its associated SELECTOR GROUND hub is connected to a ground supply.

There are thus two types of programming to be done to employ Selectors: the Selector must be operated (set to Select, and dropped out, as required); and the Selector must be probed (i.e., tested to achieve variance on the basis of how it is set).

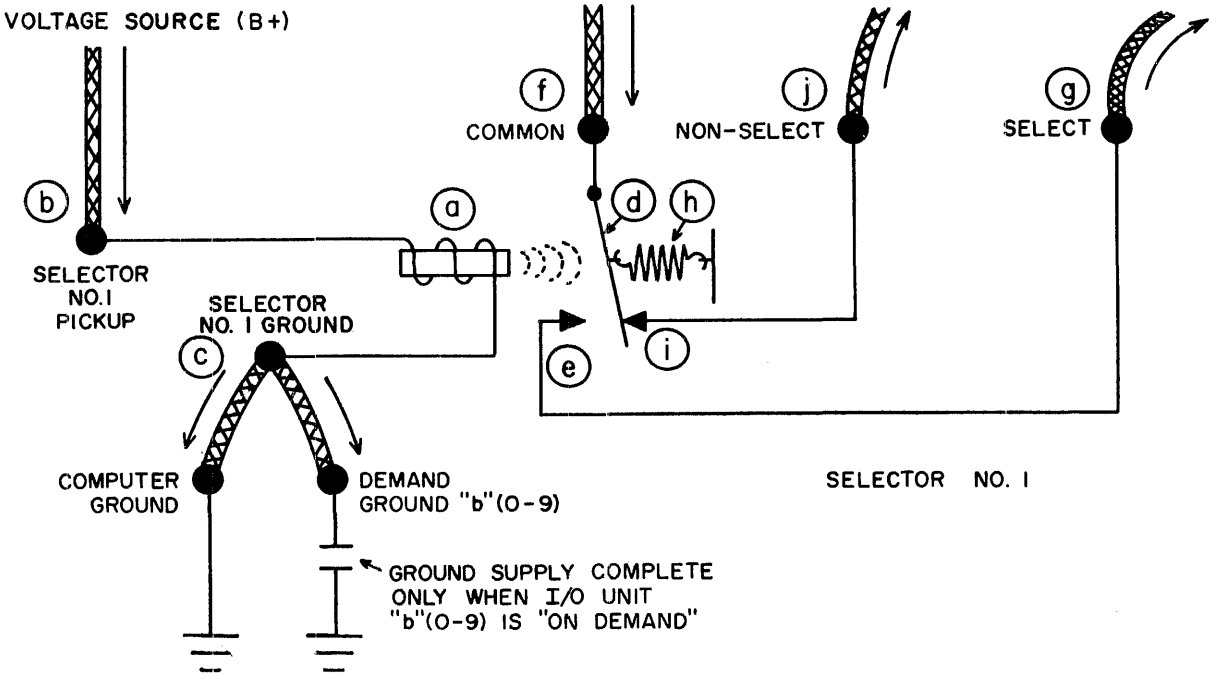
Figure II-2, Page II-50 illustrates the operation of a single-pole Selector, and the general manner in which its associated hubs are patched to achieve bi-directional routing. Note in the explanation of the operation of Selectors given in Figure II-2 that the elements (a), (b), (c), (d), and (h) have to do with control or operation of the Selector; whereas elements (e), (f), (g), (i), and (j) have to do with the program variance function of the Selector.

The UFC Model 1 has 48 Selectors. These are of two types: single pole (as described in Figure II-2 with 1 metal bar and one set of COMMON, SELECT and NON-SELECT hubs) or multi-pole (with two or four metal bars, and two or four associated sets of COMMON, SELECT, and NON-SELECT hubs). An example of a four-pole Selector is given in Figure II-3, Page II-51. When the PICKUP of a multi-pole selector is energized, more than one metal bar is affected. Thus, more than one choice may be made based upon the condition operating the Selector. The 48 Selectors that are available are divided as follows:

16 single-pole selectors
24 two-pole selectors, and
8 four-pole selectors

A brief summary of the points to note about Selectors as "decision making" devices is given on Page II-51.

* Selectors can also be used for "on-off" control. See Paragraphs IIB1e(1)(b), IIB1e(1)(c) and IIB1p.



The iron core (a) around which the wire is coiled can become magnetized when

voltage is applied to Selector #1's PICKUP hub (b); and

Selector #1's GROUND hub (c) is connected to a ground supply.

Thus when current enters Selector #1's PICKUP hub, the metal bar (d) is drawn to the left and makes contact with the SELECT bar (e). The metal bar (d) will remain in this position as long as current flows through the coil, and any pulse or enable entering Selector #1's COMMON hub (f) will emit from Selector #1's SELECT hub (g) when the metal bar is so held.

When no current is flowing through the coil, the iron core is not magnetized and spring (h) will keep the metal bar in contact with the NON-SELECT bar (i). Any pulse or enable entering Selector #1's COMMON hub will emit from Selector #1's NON-SELECT hub (j) with the metal bar in this position.

Figure II-2. Operation of a Single Pole Selector.

Brief Summary of Points to Remember About Selector "Routing" Circuits.

1. A Selector is merely a switch. It does not produce a pulse, an enable, B+, or ground of itself.
2. A Selector will remain on the Select side only as long as B+ is entering its PICKUP hub, and its coil circuit is grounded; when its B+ is lost or the ground supply is interrupted, the Selector will return to the Non-Select side.
3. Any factor, or combination of factors (as a process, address, shift, Program Step sequencing, etc.) can be changed through use of Selectors.
4. If only one factor is to be changed because of a certain condition, a single-pole Selector is used; if more than one is to be changed, a multi-pole Selector is used.
5. The factor which is to be changed by a Selector is wired to the Selector's COMMON hub. The two variations are wired from the Selector's SELECT and NON-SELECT hubs.
6. In simple application of Selectors, the normal use of a factor is patched from a Selector's NON-SELECT hub, and the special use of the factor is patched from the SELECT hub. In more complex applications, wherein the end result is produced via a network of program variance devices, this general rule cannot always be applied. In such cases, the patching is determined solely by the interrelationships between dependent conditions that must be established.
7. The same factor can be routed through several Selectors, if necessary.

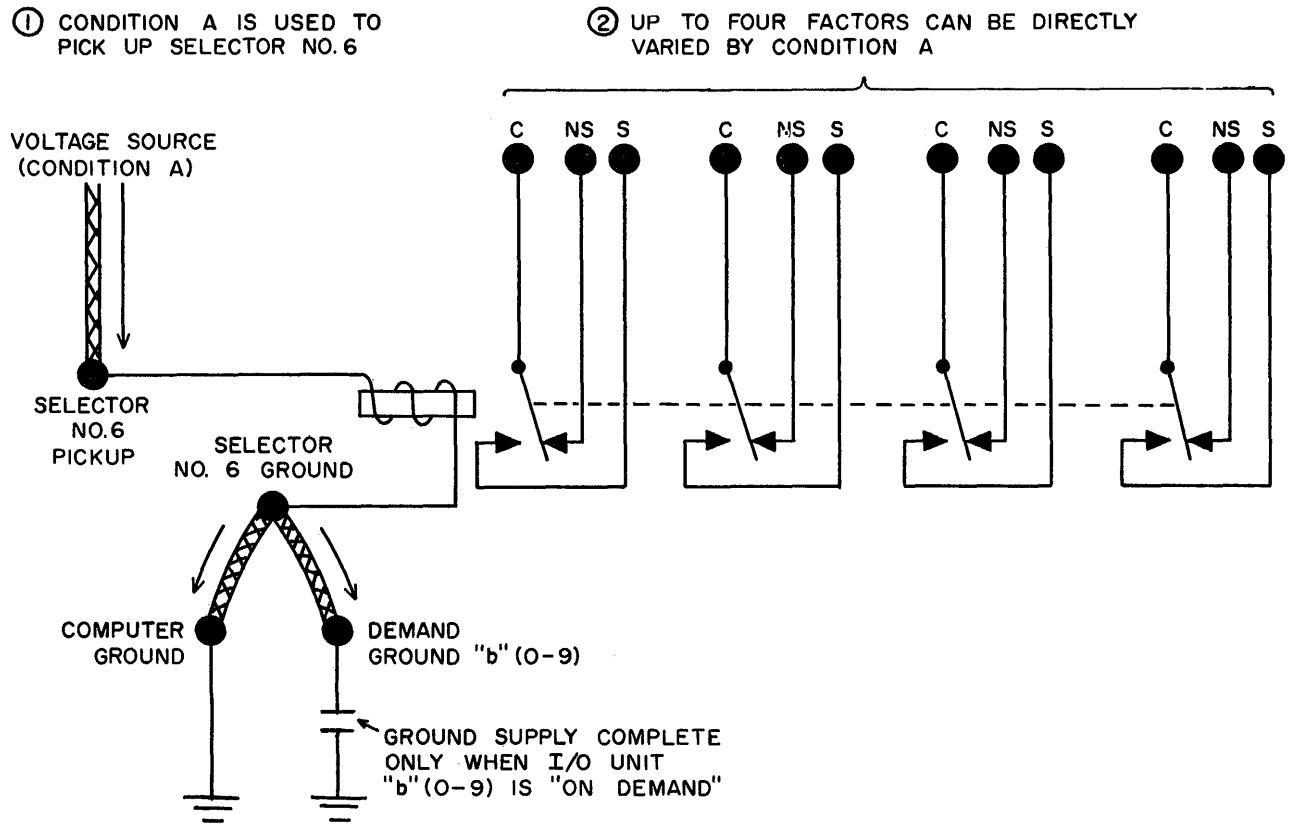


Figure II-3. An example of a Four-pole Selector.

(1) SELECTOR PICKUP (1-48) Patching

Each Selector has an associated SELECTOR PICKUP hub. As noted previously, B+ must be constantly applied to this hub to keep the Selector in a Select state; and when no current enters this hub the Selector is in a Non-Select state. The following are normally used as sources of current for the SELECTOR PICKUP hubs:

Program Select (B+)
Selector Hold (B+)
(LS) I/O-COMPUTER CONTROL LINE hubs (a-l)
CONSOLE (B+)

(a) From Program Select (B+)

Program Selects are used to pick up Selectors at a specific time in a program and to exercise control over the length of time the Selector remains picked up. A Program Select is the only practical means of picking up a Selector by a pulse. Program Selects can also be used to delay a STEP OUT or other pulse; or they can be used to operate a Selector and delay a pulse.

There are 16 Program Selects: PS 1 through PS 16. Each has 4 hubs:

IN: When a pulse is received at the IN hub, the OUT hub becomes a source of B+. The IN hub is usually wired from a STEP OUT hub, the +, —, or 0 hubs of Branching, or some other similar pulse source.

OUT: The OUT (B+) hub of a Program Select is the hub that emits a steady current to hold one or more (all 48, if necessary) Selectors on the Select side. As soon as a pulse enters the IN hub, the OUT hub will emit B+. The OUT hub is patched to the PICKUP hub(s) of the Selector(s) to be changed to their Select state. It requires approximately 10 milliseconds for a Selector to change to its Select state. (Once picked up in this manner, a Selector will then remain Select until a pulse is received by the Program Select's DROP-OUT hub, or a CLEAR hub is pulsed as described below or the ground supply is lost.)

DELAYED OUT: In the case of Program Select #1 only, the time interval between applying current to the IN hub and the emission of the DELAYED OUT hub is 15-20 milliseconds. In all other Program Selects, the time interval is 10-15 milliseconds. This will allow any Selectors picked up by out patching to change to their Select state. The DELAYED OUT hub need not always be patched; however, there are two general cases where it must always be patched:

if the Selectors picked up by the Program Select are to be used next in the program (i.e., would be used sooner than 10 milliseconds if the program were patched to continue). In this case the source of current must be patched to IN, and the DELAYED OUT patched to continue the program; and

if the pulse source is not split-wired (i.e., bussed) to continue the program. In this case DELAYED OUT patching is the only way of continuing the program.

DROP OUT: The purpose of the DROP OUT hub is to end the current going to the Selector's coil and thereby cause the Selector to return to its Non-Select side. When a Program Select's DROP OUT hub is pulsed, the Program Select's OUT hub ceases to emit B+, and supply current to the Selector(s) PICKUP hub(s). Unless each PICKUP hub wired from the Program Select OUT also has another current source, its associated Selector will drop out. (As in the case of changing a Selector to its Select side, approximately 10 milliseconds are required to drop out a Selector.) The pulse applied to the DROP OUT hub can originate from any pulse source.

No "out" pulse is generated when a Program Select is dropped-out. The source of the drop-out pulse must therefore be appropriately patched to continue the program. This is usually done by bussing the drop-out pulse through an OUT-EXPANDER or the normal BUS hubs described later in this section.

It is possible to drop out all Selectors which have been picked up via Program Select(s) except those whose PICKUP hubs are also wired from Selector Hold B+ (or another steady B+ source). This is done by pulsing one of the CLEAR hubs associated with the Program Selects. When a pulse is sent to a CLEAR hub it has the same effect as pulsing the DROP OUT hub of each of the active Program Selects. (This would most frequently be done at the end of a program, so that all Selectors would be on the Non-Select side at the start of a new program.) CLEAR has no "out" hub; however, each of the four sets of CLEAR hubs is a bussed pair. The pulse source is patched to one hub of the pair, and the other hub of the pair is patched to continue the program.

(b) From Selector Hold (B+)

The SELECTOR HOLD (B+) hubs are a constant source of power which never stop emitting as long as the machine is on. The current these hubs supply can be used to operate one or more Selectors throughout an entire program, or if suitable patching is employed, throughout any portion thereof.

Since Selector Hold B+ has no drop out of its own, a Selector picked up via direct SEL HOLD (B+) - SELECTOR PICKUP patching, will remain on the Select side until the machine is turned off. This is usually not as efficient as other uses of Selector Hold B+, and hence direct SEL HOLD (B+)-SELECTOR PICKUP patching has restricted application. When a Selector employing Selector Hold B+ is not to be kept in its Select state throughout a program, several Program Selects, another Selector, and one pole of the Selector to be controlled can be used as shown in Figure II-4, Page II-54

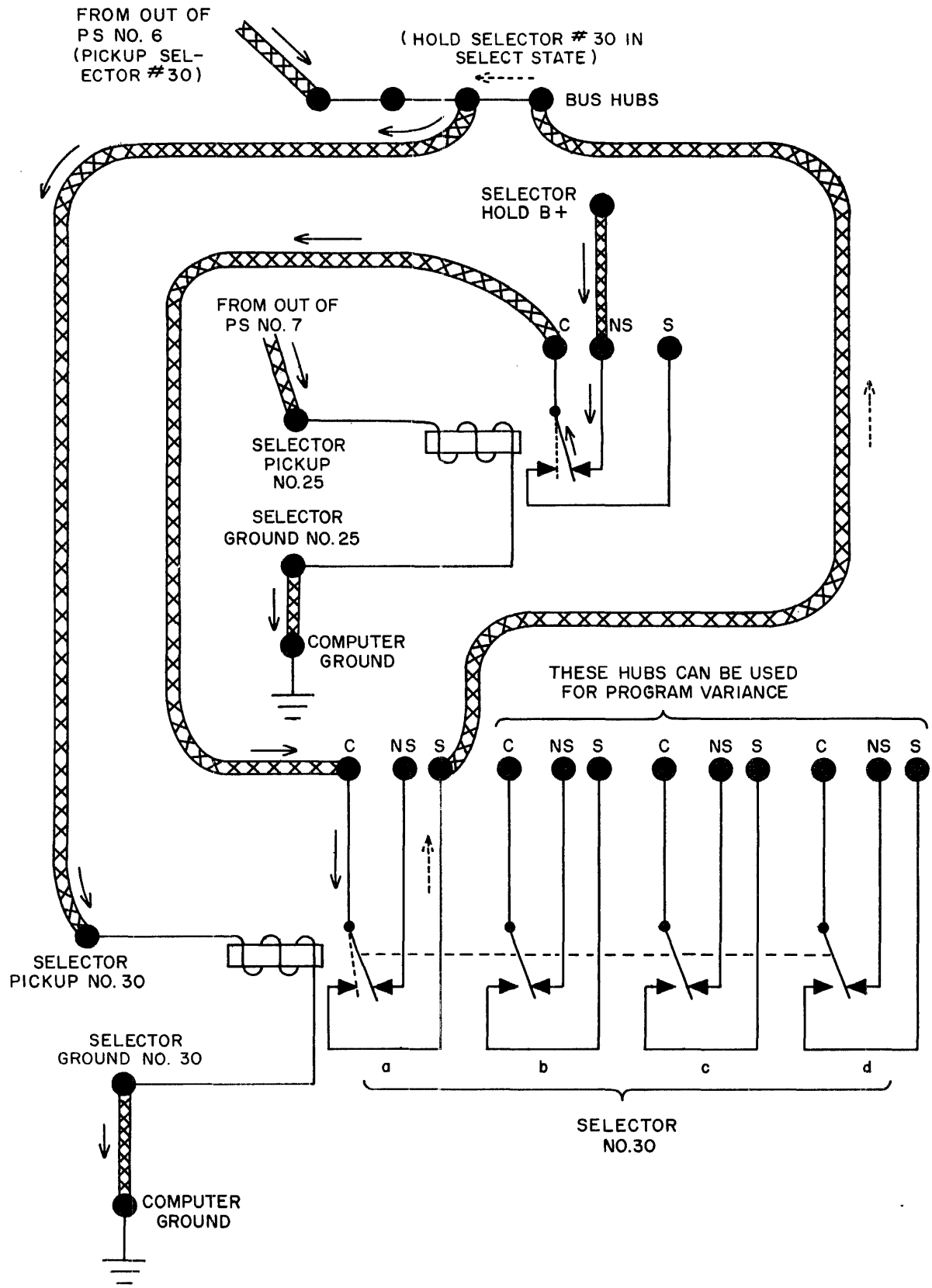


Figure II-4. Controlled Use of Selector Hold (B+).

Assume that Selector #30, a 4-pole Selector, is to have Selector Hold (B+) power applied to its PICKUP hub only for a portion of a program, but is to have that power even though the Program Selects' CLEAR hubs are pulsed. Program Selects #6 and #7, pole a of Selector #30, and a single-pole Selector #25, can be used to accomplish this by applying, and then controlling, Selector Hold (B+). (Poles b, c, and d of Selector #30 are used for their normal purpose of re-routing.)

Until Selector #30 is picked up by PS #6, Selector Hold (B+) power cannot flow to the PICKUP hub of Selector #30. It can only go from the NON-SELECT hub of Selector #25 through the COMMON of Selector #25 to the COMMON of Selector #30, Pole a. With Selector #30 on the Non-Select side, the current can go no further. However, as soon as PS #6 changes Selector #30 to the Select side, Selector Hold (B+) power will emit from the SELECT hub of Selector #30, Pole a (see dashed arrows in Figure II-4) and be applied to the PICKUP hub of Selector #30 via the BUS hubs. Now, even though PS #6 is dropped out by pulsing a CLEAR hub, Selector Hold (B+) power will keep Selector #30 on the Select side. Selector #30 will remain Select until the PS #7 IN hub is sent current and the resulting PS #7 OUT current picks up Selector #25. When Selector #25 changes to its Select state, Selector Hold (B+) power will no longer be connected to the COMMON of Selector #25 (and, therefore, cannot be maintained on Selector #30's PICKUP hub). Selector #30 will therefore revert to its Non-Select state. It will remain in this state until PS #6 OUT current again picks it up. When Selector #30 is picked up, Selector Hold (B+) power cannot again be applied to its PICKUP hub until Selector #25 is dropped out.

(c) From the (LS) I/O-COMPUTER CONTROL LINE hubs (a-l)

As explained in Section V, each I/O Unit has a set of 12* (LS) I/O-Computer control lines. These lines, when energized, can be used to notify the computer program that:

a particular condition exists in the I/O Unit , or

in the case of input I/O Units, that a control position on the input medium (as a punched card) was sensed.

Each I/O Unit's (LS) I/O-Computer control lines are B+ lines, when energized, and can thus be used to pickup Selectors.

The (LS) I/O-Computer control lines are energized as follows:

As part of the system program, specific conditions in the I/O Unit, as well as the detection of various control data (in input I/O Units), are assigned to specific (LS) I/O-Computer control lines.

* Although each I/O Unit's Demand Station contains 12 (LS) I/O-Computer control lines (a-l), not all of these control lines are used by each I/O Unit. However, each I/O Unit has a definite set and the set is lettered to correspond with the (LS) I/O-COMPUTER CONTROL LINE hubs (a-l) that will emit B+ when each control line is energized in the I/O Unit and the I/O Unit is placed "on demand".

Provision is made so that the occurrence of each condition (or each detection of control data) can be "remembered" in the I/O Unit and can apply B+ to the correct (LS) I/O-Computer control line (a-l) when the I/O Unit is placed "on demand". This is programmed for in various ways in the different I/O Units. In general, a switch setting or patchcord wiring involving Selectors in the I/O Unit are employed. If a switch setting is used, this generally will permit B+ to be directly applied to a (LS) I/O-Computer control line when the I/O Unit is placed "on demand". If patchcord wiring is employed, dc enables associated with a particular condition (or the detection of control data) in the I/O Unit are patched to operate Selectors in the I/O Unit, so that B+ (available on the I/O Unit's plugboard) can be applied via one or more Selector's routing circuit to the required (LS) I/O-Computer control lines when the I/O Unit is placed "on demand".

The meaning of each control line can be made to vary from program to program, or each control line can be assigned a constant meaning. In any event, the meaning assigned each (LS) I/O-Computer control line must be programmed in the I/O Unit (i.e., must be specified by a switch setting on the I/O Unit's Control panel, or specified by patchcord wiring on the I/O Unit's Plugboard, as part of the design of the system program), and the use to be made of each (LS) I/O-Computer control line must be programmed, both in the I/O Unit (as noted above) and on the Program Control Plugboard (as described below).

When a particular I/O Unit is placed "on demand" (See Section V) that I/O Unit's, and only that I/O Unit's, (LS) I/O-Computer control lines are connected to the (LS) I/O-COMPUTER CONTROL LINE hubs (a-l) on the Program Control Plugboard. Any of the (LS) I/O-Computer control lines that are energized at this time will cause the correspondingly-lettered (LS) I/O-COMPUTER CONTROL LINE hubs to emit B+, and these hubs will continue to emit as long as (but only as long as) the I/O Unit "on demand" remains "on demand".

The (LS) I/O-COMPUTER CONTROL LINE hubs (a-l) can thus be patched to one or more SELECTOR PICKUP hubs, and the Selectors picked up will remain in their Select state as long as the I/O Unit remains "on demand".

Since the (LS) I/O-COMPUTER CONTROL LINE hubs (a-l) are commonly shared by all I/O Units (but by each only when it is "on demand"), the following patching procedure is employed if a Selector is to be uniquely picked-up by a (LS) I/O-Computer control line from a particular I/O Unit.

The particular (LS) I/O-COMPUTER CONTROL LINE hub (a-l) is patched to a Selector's PICKUP hub;

The Selector's GROUND hub is patched to the DEMAND GROUND hub (0-9) that has the same number as that assigned to the I/O Unit.

The Selector can thus only be picked up when the I/O Unit involved is "on demand", and the particular (LS) I/O-Computer control line is energized.

If several Selectors are to be picked up when a particular (LS) I/O-COMPUTER CONTROL LINE hub (a-1) emits, that hub is connected to the desired Selectors' PICKUP hubs via BUS hubs.

If the same (LS) I/O-COMPUTER CONTROL LINE hub (a-1) is to be identically used for picking up Selector(s) if it emits while any one of several I/O Units are "on demand", the GROUND hub(s) of the Selector(s) involved is (or are) patched to each I/O Unit's associated DEMAND GROUND (0-9) via BUS hubs. (See alternate GROUND hub patching shown in Figure II-5, Page II-58).

If the detection of a particular (LS) I/O-Computer control line is to be "remembered" in the program after the I/O Unit producing it is taken "off demand", the patching illustrated in Figure II-5, Page II-58 and described on Page II-59, can be employed.

(d) Console B+ (CNSL B+)

When the CONSOLE-NORMAL switch on the UFC Inquiry Typewriter Control Panel is set to CONSOLE, the CNSL B+ hub on the Program Control Plugboard emits B+ . This hub can be patched to pick-up Selectors and its power is available during all console operations of the Typewriter.

(2) SELECTOR GROUND (1-48) PATCHING

(a) To Computer Ground:

When the GROUND hub of a Selector is patched to a COMPUTER GROUND hub, the Selector's coil circuit is unconditionally grounded. The Selector will change to its Select state whenever B+ is applied to its associated PICKUP hub.

(b) To Demand Ground:

When the GROUND hub of a Selector is patched to a DEMAND GROUND (0-9) hub, the Selector's coil circuit has a ground supply only when the corresponding I/O Unit (0-9) is "on demand". Applying current to a Selector whose GROUND hub is so patched will not set the Selector to its Select state unless (or until) the appropriate I/O Unit is "on demand".

f. Alternate Switches (A-F)

An Alternate Switch is a manually operated (single pole) Selector whose state (Select or Non-Select) is determined by the setting of an associated toggle switch on the Program Control #1 Cabinet. If the toggle switch associated with an Alternate Switch is set to SELECT, the Alternate Switch is set to its Select state and remains in this state until the toggle switch is set to NON-SELECT. The Alternate Switch then assumes its Non-Select state and remains in this state until the position of its associated toggle switch is again changed.

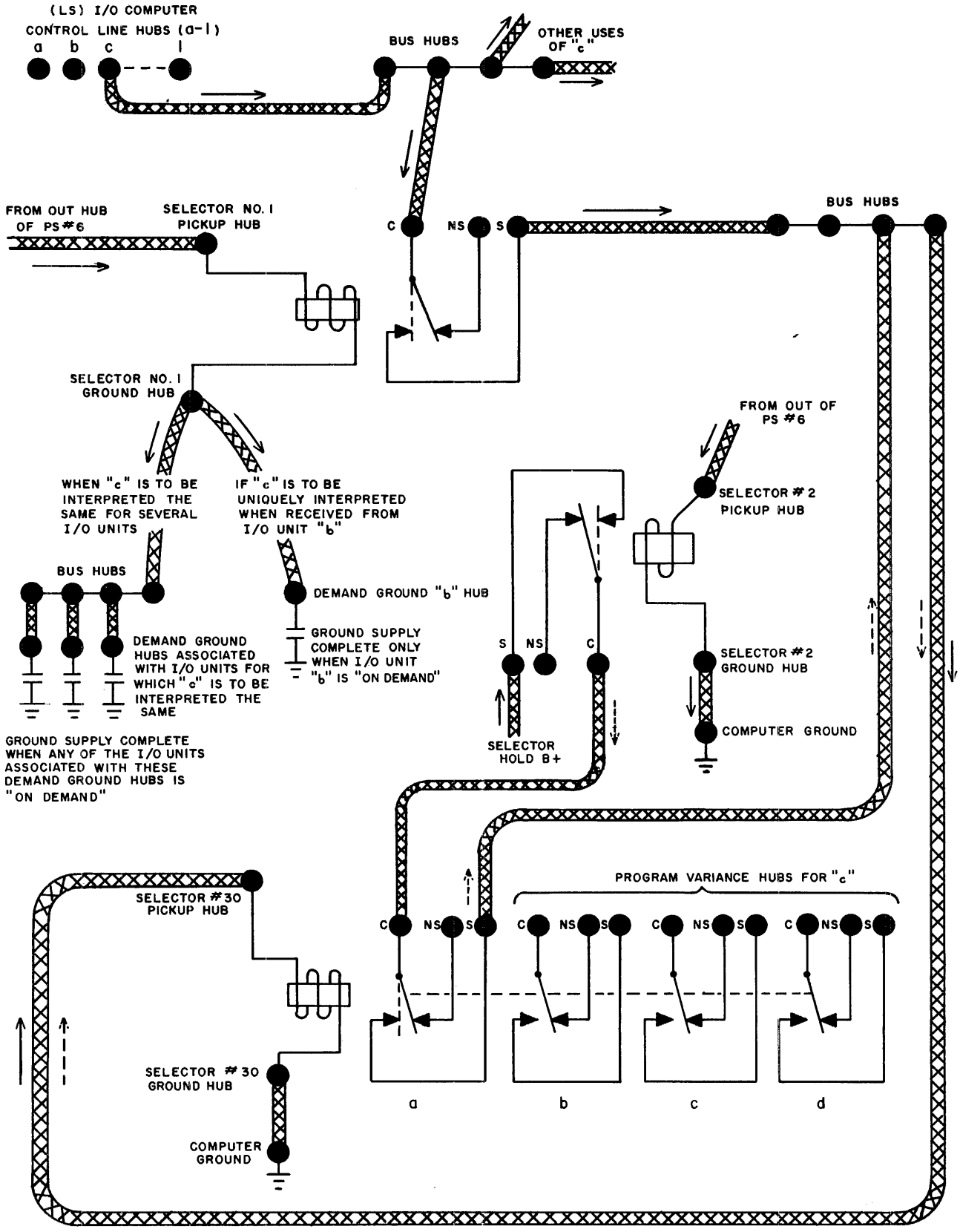


Figure II-5. An Example of how a (LS) I/O-Computer Control Line can be "remembered" in a Program.

Explanation of Figure II-5

The IN hub of PS #6 is pulsed prior to putting the I/O Unit "on demand". This picks up Selector #2 and permits Selector #1 to be picked-up when the appropriate I/O Unit is placed "on demand".

When the appropriate I/O Unit is placed "on demand" Selector #1 picks up. If the I/O Unit has energized (LS) I/O-Computer Control line "c", the (LS) I/O-COMPUTER CONTROL LINE hub "c" will emit B+ and will, (among other things) pick-up Selector #30.

When Selector #30 is picked up, Selector Hold (B+) can be routed to Selector #30's PICKUP hub as follows: from SELECT hub of Selector #2, out the COMMON of Selector #2, to the COMMON of Selector #30, out the SELECT of Selector #30 to Selector #30's PICKUP via the bus.

Even though the I/O Unit "on demand" is taken "off demand", Selector #30 remains picked up.

Selector #30 can be dropped out by pulsing the DROP OUT hub of PS #6. If this is done, Selector #30 cannot again be picked up until the following sequence takes place: PS #6's IN hub is pulsed, the appropriate I/O Unit is placed "on demand" and the (LS) I/O-COMPUTER CONTROL LINE hub "c" emits B+.

There are 6 Alternate Switches (A-F) and each has a set of COMMON, SELECT, and NON-SELECT hubs on the Program Control Plugboard. The factor which is to be changed is usually patched to the COMMON hub of the Alternate Switch, and the variations are patched from the SELECT and NON-SELECT hubs, just as for Selectors. The main differences between a Selector and an Alternate Switch are (a) the operator, rather than the plugboard program, determines whether each Alternate Switch is to be Select or Non-Select; and (b) the setting for an Alternate Switch remains the same (even from program to program) until the operator changes it.

If there are two programs patched on one plugboard, and these programs are not being handled simultaneously, an Alternate Switch could be used to determine which program is to be used at a particular time.

g. Decision Elements for Enables

As noted in Paragraph IIB1b, the enable out hubs of a Program Step (PROCESS, V_1 ADDRESS, V_1 SHIFT, V_2 ADDRESS, V_2 SHIFT, R SHIFT, and R ADDRESS) can be directly or indirectly patched to their destination. When patched directly, each process, address, or shift is unconditionally defined. When patched via other hubs, a process, address, or shift can be conditionally defined. The usual manner in which these enable out hubs are patched to conditionally define a process, address or shift is described below. The following also apply to enable-routing, in general. (Selectors and Alternate Switches can also be used, as noted previously, to route B+; however, the CDR ENABLE hubs must not be patched to or from B+).

(1) Selectors and Alternate Switches

In these devices, the process, address or shift hubs are usually patched to the COMMON hub of the Selector or Alternate Switch. Patchcord wiring from the SELECT and NON-SELECT hubs then provides the required variance (two choices per Selector pole, or Alternate Switch are available). If the enable is patched to a SELECT or NON-SELECT hub, only the COMMON will emit, and it will emit only when the Selector is appropriately set.

(2) CDR ENABLE Probes

(a) Explanation of Code Distributor, CD

The Code Distributor, CD, is a multiple-position plugboard (electronic) switch which, when appropriately used, permits a programmer to conditionally define any machine function in a variety of ways.

CD has two principal parts: the Code Distributor Register (CDR), and a multi-purpose Translator. Program variance is achieved by storing a valid control character in CDR and then probing a particular group of circuits in the Translator to receive a "route" or path for the probing signal (pulse or d-c enable).

CDR is thus a one-character register that is used to set the "switch". It receives the character it stores when it is

addressed as a Destination in either an Instruction Word (W = 994) or a Program Step (R ADDRESS-CDR patching). The control characters used in CDR are usually sent to the computer by an input I/O Unit as part of the input data. The valid characters that can be stored in CDR, i.e., those which can be placed in CDR and subsequently translated, are listed in Table II-12 below.

Table II-12. Characters that can be Translated in CDR

ZONE BITS	EXCESS-THREE BITS									
	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100
00	0	1	2	3	4	5	6	7	8	9
01	;	A	B	C	D	E	F	G	H	I
10	)	J	K	L	M	N	O	P	Q	R
11	+	/	S	T	U	V	W	X	Y	Z

Note that the 40 valid (or usable) characters can be thought of as 10 groups (columns) of 4 characters each; each character in a group (column) having the same excess-three bits. For example 2, B, K, and S are such a group, since they each have 0101 as their lower-order 4 bits. In some uses of the contents of CDR (as CDR PULSE Probe and CDR GROUP Probe, described below) only the four lower-order bits of the character in CDR are translated. In such cases, any one of the four characters with the same excess-three bits will be translated in the same manner (hence if any one of the four is placed in CDR, the same result is produced). In other uses of the contents of CDR (as CDR ALPHANUMERIC Probe) the entire six bits in CDR are translated. In such cases, all 40 characters are uniquely translated and each character placed in CDR will produce a different result. (The results referred to in this paragraph are the "routing" or "switching" for which CD is employed). If a valid character is not in CDR at the time of a probe, the computer program hangs up.

The Translator which interprets the character held in CDR is composed of six circuits: five interpret only the lower-order 4 bits of CDR, the sixth interprets all six bits in CDR.

Translator Circuit	Associated hubs	Bits in CDR that are Interpreted
CDR PULSE IN	One set of four (identical) IN hubs, and ten discrete OUT hubs: 0-9	only excess-three bits
CDR GROUP I (ENABLE)	one set of four (identical) IN hubs, and ten discrete OUT hubs: 0-9	only excess-three bits
CDR GROUP II (ENABLE)	one set of four (identical) IN hubs, and ten discrete OUT hubs: ";"-I	only excess-three bits
CDR GROUP III (ENABLE)	one set of four (identical) IN hubs, and ten discrete OUT hubs: ")"-R	only excess-three bits

Translator Circuit	Associated hubs	Bits in CDR that are Interpreted
CDR GROUP IV (ENABLE)	one set of four (identical) IN hubs, and ten discrete OUT hubs: "+ "-Z	only excess-three bits
CDR ALPHANUMERIC (ENABLE)	one set of four (identical) IN hubs, and 40 discrete OUT hubs: 0-Z	all six bits

The operation of each of these Translator circuits is basically the same: when the IN hub is appropriately signalled (i.e., pulsed or enabled as required by the particular circuit involved), one of the OUT hubs will emit if there is a valid or usable character in CDR at the time. The specific operation of each Translator circuit is described below, as appropriate. The Translator's CDR GROUP and ALPHANUMERIC circuits are used to route d-c enables and are accordingly discussed in the following paragraphs. The Translator's CDR PULSE circuit is discussed in Paragraph II-Bli, since it functions to route pulses (i.e., it is a decision-making device for pulses).

(b) CDR GROUP IN Probe

When a particular CDR GROUP IN hub (GROUP 1, GROUP 2, GROUP 3, or GROUP 4) is enabled via patchcord wiring from an enable source (as a V_1 ADDRESS hub) and one of the usable 40 characters is in CDR, a single d-c enable is obtained from the associated CDR GROUP circuit in the Translator. That is, a d-c enable is made available from one and only one of the ten CDR OUT hubs associated with the CDR GROUP IN hub enabled. Thus, given a usable character in CDR (See Figure II-6):

If a CDR GROUP 1 IN hub is enabled, one of the CDR OUT hubs labeled 0-9 will emit d-c;
If a CDR GROUP 2 IN hub is enabled, one of the CDR OUT hubs labeled ";" or A-I will emit d-c;
If CDR GROUP 3 IN hub is enabled, one of the CDR OUT hubs labeled ")" or J-R will emit d-c;
If a CDR GROUP 4 IN hub is enabled, one of the CDR OUT hubs labeled " + ", or "/" or S-Z will emit d-c;

depending on which usable character is in CDR at the time.

To understand the operation of CD upon receipt of this type of probe, assume that the CDR GROUP 2 IN hub is enabled and that the OUT hub "J" emits. A d-c enable from hub J does not necessarily imply that a J was actually stored in CDR, since each of the Translator's Group circuits look only at the excess three bits of the character in CDR. Thus when the J hub emits it means (a) that a usable character was stored in CDR, and (b) that specifically either 1, A, J, or / (the four characters in Univac code that have the same excess three bits) was in CDR at the time of the probe.

When CDR GROUP Probe is used to conditionally define a process, address, or shift the 40 control characters that can be

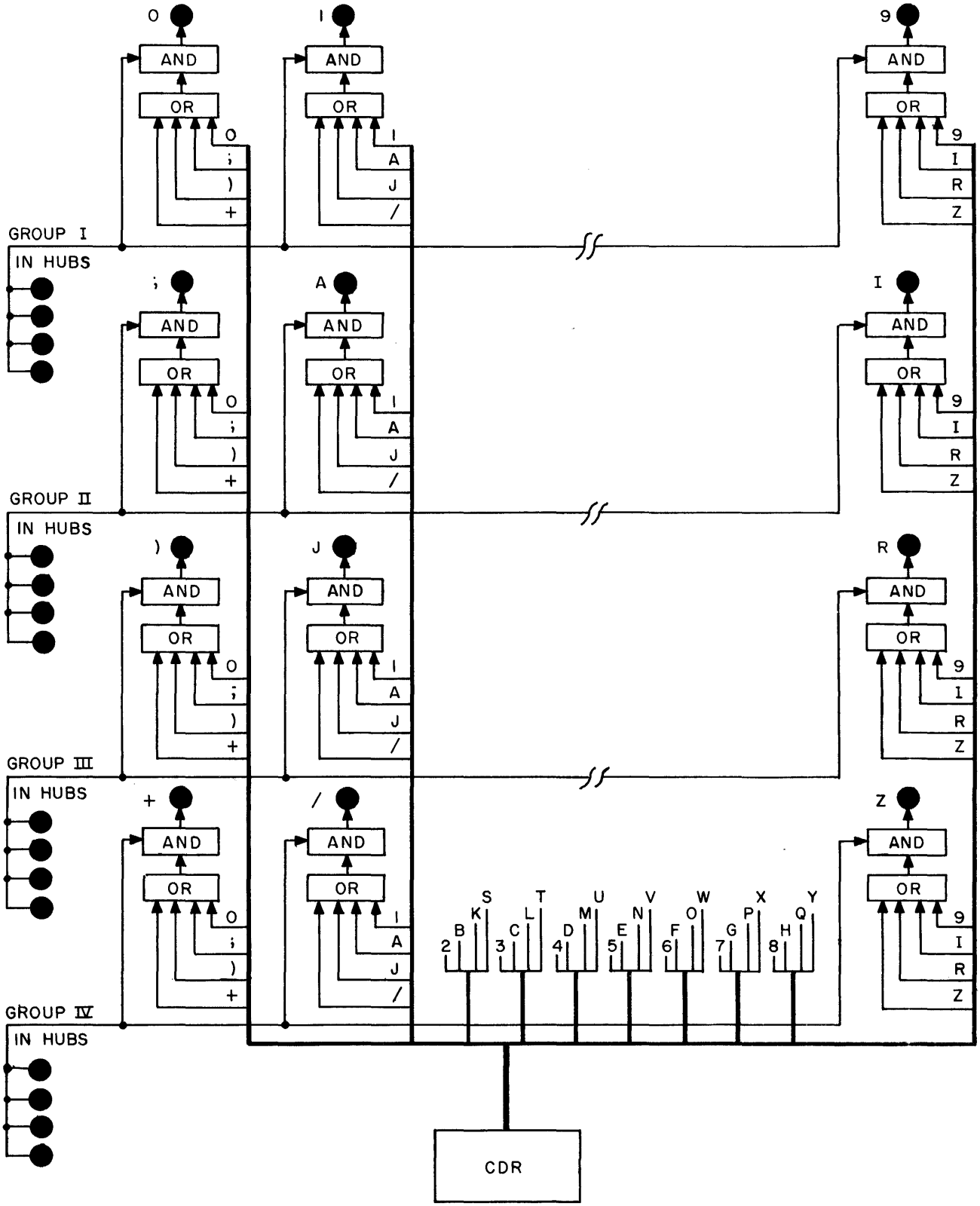


Figure II-6. CD Translator's CDR GROUP IN Circuits

sent to CDR are regarded as 10 sets of 4 characters each. Each set of four being an identical "switch" for the (probing) process, address, or shift enable; and hence one of ten distinct means of defining the process, address, or shift involved.

Since there are four CDR GROUP circuits, each set of four characters (as 1, A, J, and /) can be given four separate routing paths by appropriately patching the OUT hubs of each CDR GROUP circuit used:

one from the CDR GROUP OUT hub "1"
one from the CDR GROUP OUT hub "A"
one from the CDR GROUP OUT hub "J", and
one from the CDR GROUP OUT hub "/"

In such cases when a particular CDR GROUP IN hub (as CDR GROUP 1 IN) is enabled at one point in the program and 1, A, J, or / are in CDR, one route is specified by patchcord wiring (from hub "1") for the d-c enable used in probing; when another CDR GROUP IN hub (as CDR GROUP 2 IN) is enabled and any one of the same characters (1, A, J, or /) are in CDR, a different route can be specified from that GROUP's OUT hub, ("A") etc.

(c) CDR ALPHANUMERIC IN Probe

If any one of the CDR ALPHANUMERIC IN hubs is enabled and CDR contains one of the 40 usable characters, a d-c enable is available from only one CDR OUT hub (0-Z): the one which corresponds in label to the actual usable character held in CDR. For example, if A is in CDR, (See Figure II-7, Page II-65), CDR OUT hub "A" and only that hub emits.

From one up to 40 "routes" can thus be defined for a process, address, or shift enable (or any other enable) by appropriately loading CDR and using those enables as CDR ALPHANUMERIC probes.

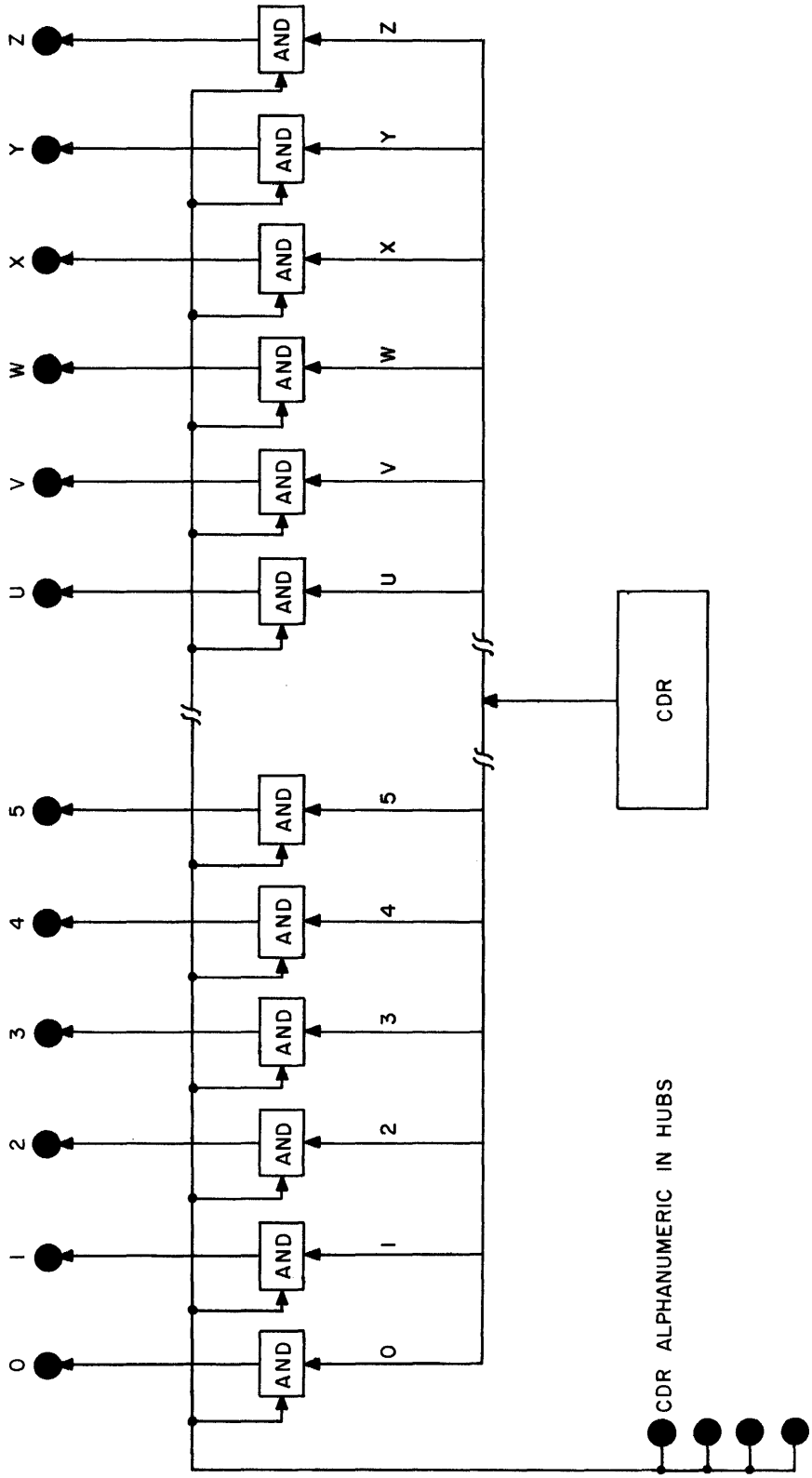


Figure II-7. CD Translator's CDR ALPHANUMERIC Circuits

h. Sub-Step Patching (General Information)

In addition to the patching that is required to completely define a basic operation for the computer, each Program Step can also be patched to initiate other operations as Program Control passes from a Program Step to the next instruction (Program Step or Instruction Word). These auxiliary plugboard-defined operations are called Sub-Steps, and each has an associated group of hubs on the Program Control Plugboard which must be appropriately signalled to initiate that Sub-Step. Some Sub-Steps, as those which initiate General Storage Operations, are identical to the correspondingly-named Sub-Instructions initiated in internally-stored programs; other Sub-Steps, as Branching, are used only in plugboard operation of the computer. Some Sub-Steps initiate time-sharing operations in the system, others are used for making decisions in the system program, or for specially-conditioning the Central Computer.

Usually Sub-Steps are initiated via patchcord wiring from a Program Step's STEP OUT hub. Sub-Steps can also be initiated, however, during execution of an internally-defined program via the Special Character Out and Breakpoint Sub-Instructions, provided the hubs these Sub-Instructions cause to emit are appropriately patched. This is common procedure, for example, if the Condition Compare or Clear BTB to Ignore Sub-Steps (discussed below) are needed in the internal program.

For convenience, Sub-Steps are discussed below under the following headings:

Decision Elements for Pulses
Central Computer Sub-Steps
General Storage Sub-Steps
I/O Control Sub-Steps

i. Decision Elements for Pulses

One of the commonest "decisions" made by plugboard-defined programs is the choice of which Program Step is to be executed next. Program Steps are sequenced by STEP OUT-STEP IN patchcord wiring: a STEP OUT can be wired to any STEP IN, directly or indirectly. When direct patching is employed, the next Program Step is unconditionally defined; when indirect patching (i.e., when patching via decision-making devices for pulses) is employed, Program Step sequencing can be varied. This variation

can be a conditional definition of which Program Step is to be executed next, as when one or more of the following Sub-Steps are initiated by (or from) the STEP OUT pulse:

Branching
Channel Search Probe
Channel Search Probe and Wait
Selector or Alternate Switch "Probes"
Function Sequence
CDR (PULSE) IN; or it

can be merely a delay in initiating the next program Step as when the Function Delay Sub-Step is employed (or, if this Sub-Step is used in connection with the above Sub-Steps, both delay and variance can be achieved); or it

can terminate a sequence of Program Steps, and cause Program Control to go to the internally-stored program for the next Computer Instruction, as when the Next Instruction Sub-Step is carried out, etc.

There are thus a variety of "decision making" devices for pulses on the plugboard, and a number of combinations of decisions that can be made between Program Steps (i.e., without carrying out a Computer Instruction). Each decision-making device for pulses is discussed below. The comments made are not restricted to STEP OUT pulses; these devices can route any pulse.

(1) Branch Sub-Step

Branching is a means of determining the next operation, depending on whether the result of a Program Step is a +, 0, or - (minus) quantity. It is also used after Program Steps in which the Compare process is carried out to determine the next operation on the basis of whether $V_1 > V_2$, $V_1 = V_2$, or $V_1 < V_2$, respectively. (See Page II-14).

There are 12 separate sets of BRANCH hubs on the Program Control Plugboard: BR 1 through BR 12. Each set has four hubs: an IN hub that is patched from a pulse source (as STEP OUT), and three out hubs (+, 0, and -) that are patched to other pulse-in hubs (as STEP IN, a Selector COMMON, etc.) to continue the program.

When the IN hub is pulsed, the setting of Branch Storage is examined. If Branch Storage is set to +, the + hub emits; if set to 0, the 0 hub emits; and if set to -, the - hub emits. Note: Branch storage will be found set to 0 unless the Program Step whose STEP OUT probes Branch Storage set Branch Storage to + or -.

(2) Channel Search Probe Sub-Step

When either of the following General Storage operations has been initiated

Channel Search Equal
Channel Search Unequal

a Channel Search Probe Sub-Step can be initiated to test whether the channel search has been completed or not. This Sub-Step permits the plugboard program to continue immediately if the channel search operation is not completed; and, if the channel search is completed, allows a "decision" based on the result of the search to be made regarding the next operation in the program.

There are four (identical) sets of hubs on the Program Control Plugboard that are associated with the Channel Search Probe Sub-Step.

Any set can be used. Each set has five hubs: a CS PROBE hub that is patched from a pulse source, and four out hubs (ACTIVE, +, -, and 0) each of which is patched to the particular hub that is to be pulsed if that Channel Search Probe out hub emits.

When the CS PROBE hub is pulsed, the status of the previously-initiated channel search operation is tested: if the General Storage System is ACTIVE, the ACTIVE hub emits indicating that the channel search operation is still in progress; if the General Storage System is not ACTIVE, the setting of Channel Search Storage is examined: if Channel Search Storage is set to +, the + hub emits; if set to - the - hub emits; if set to 0, the 0 hub emits.

Note: If the Channel Search Probe Sub-Step is initiated but no channel search operation has been previously initiated, the computer hangs up. If the Channel Search Probe Sub-Step is initiated but no channel search operation was initiated since the last time Channel Search Storage was examined, the same hub emits that emitted during the last Channel Search Probe. Channel Search Storage is "cleared" (left with no setting) by a MASTER CLEAR and each time a Channel Search operation is initiated.

(3) Channel Search Probe & Wait Sub-Step

The Channel Search Probe & Wait Sub-Step can also be used, subsequent to the initiation of a Channel Search Equal or Unequal operation, to test for the result of the channel search. When this Sub-Step is used, however, Program Control will wait if the channel search operation is not completed. This Sub-Step is, therefore, used only when the result of the channel search must be known to continue the program.

There are four (identical) sets of hubs associated with the Channel Search Probe & Wait Sub-Step. Any set can be used. Each set has four hubs: a CS PROBE & WAIT hub that is patched from a pulse source, and three out hubs (+, -, 0) each of which is patched to the particular hub that is to be pulsed if that Channel Search Probe & Wait hub emits.

When the CS PROBE & WAIT hub is pulsed, the status of the previously initiated channel search operation is tested: if the General Storage System is ACTIVE, the computer waits until General Storage becomes INACTIVE (indicating completion of the channel search); Channel Search Storage is then examined; if Channel Search Storage is set to +, the + hub emits; if set to -, the - hub emits; if set to 0, the 0 hub emits. (See note above under Channel Search Probe)

(4) Selector or Alternate Switch "Probe" Sub-Steps

If a pulse source is patched to the COMMON hub of a Selector or Alternate Switch, the next operation will be conditionally defined by the setting of the Selector or Alternate Switch when that pulse source emits. By appropriately patching the Selector's or Alternate Switch's SELECT and NON-SELECT hubs, the desired program variance can be achieved.

(5) Function Sequence Sub-Steps

Function Sequence Sub-Steps are employed to route the program to different Program Steps when the same condition (represented by a pulse) occurs more than once in a program, but is to be used differently each time.

Each Function Sequence is a group of circuits which "pairs" the common event with a related event, at one of the times the common event occurs, in such a way that the combination of these two events produces a different result than the common event occuring alone or with any other event. Function Sequence is similar to Function Delay (described below) in that two events must occur to produce a third. However, in Function Sequence, the events producing each particular result must occur in the proper order or sequence to produce that result.

There are four (identical) sets (1, 2, 3, 4) of FUNCTION SEQUENCE hubs on the Program Control Plugboard. Each set contains the following hubs:

SET	An input hub to which a STEP OUT or similar pulse-source hub may be patched. A pulse delivered to this hub will be "remembered" in the Function Sequence circuitry.
PROBE	Also an input hub to which a pulse-source must be patched. A pulse sent to this hub will cause the OUT hub to emit <u>if a pulse was previously received by the SET hub</u>. A pulse sent to the PROBE hub is not "remembered" by the Function Sequence circuitry.
OUT	This hub will emit a pulse when the PROBE hub is pulsed if the SET hub was previously pulsed. If the SET hub and PROBE hub are not both pulsed, or are not pulsed in sequence, the OUT hub does not emit.

Note: If the OUT of Function Sequence (among other things) is to continue the program, no other Sub-Step pulse source should be employed for this purpose. In this connection it is imperative to provide that the SET hub of the Function Sequence be pulsed <u>before</u> the PROBE hub is pulsed, or no OUT pulse will be forthcoming; the computer will, therefore, hang up.

(6) CDR PULSE Probe Sub-Step

As noted in Paragraph IIBg above, one of the CD Translator circuits, CDR PULSE IN, operates only with pulses and terminates on the Program Control Plugboard in the CDR PULSE IN and CDR PULSE OUT (0-9) hubs. When any of the four (identical) CDR PULSE IN hubs is pulsed (See Figure II-8, Page II-71) and one of the 40 usable characters listed on Page II-61 has been stored in CDR, a pulse is emitted from one of the CDR PULSE OUT (0-9) hubs. The hub which emits depends on the character held in CDR at the time of the probe. As in the case of the CD Translator's CDR GROUP circuits, the Translator's CDR PULSE IN circuit interprets only the excess-three bits of the character held in CDR.

To cause a particular CDR PULSE OUT hub, as CDR PULSE OUT hub "3", to emit place (any) one of the four characters (3, C, L, or T) with the same excess-three bits as the digit 3 in CDR. When the CDR PULSE IN hub is pulsed, the CDR PULSE OUT hub "3", and only that hub, will emit. Ten routes, each of which can be specified by any of the four characters held in CDR, can thus be assigned a probing pulse.

(7) Function Delay Sub-Step

A Function Delay Sub-Step is carried out to delay the start of one operation until two other operations have been completed.

There are four Function Delay circuits (A-D) available. Each Function Delay has two input hubs on the plugboard and one out hub. The input hubs are labelled 1 IN and 2 IN; the out hub is labelled OUT. The OUT hub emits only after a pulse has been received by each IN hub. (The IN hubs need not be pulsed simultaneously or in any particular order; i.e., either IN pulse can be "remembered".)

If it were desirable not to start the next operation, for example, until a Selector affecting which operation is to be executed next has been pulled in, the following patching procedure can be followed:

> a condition indicating that the Selector has been pulled in, as DELAYED OUT of a Program Select, is patched to the 1 IN hub;
>
> the STEP OUT of the current Program Step is patched to the 2 IN hub; and
>
> the (Function Delay) OUT hub is patched to probe the Selector.

The next operation will not begin (i.e., the OUT hub of the Function Delay will not emit and probe the Selector) until <u>both</u> IN hubs are pulsed.

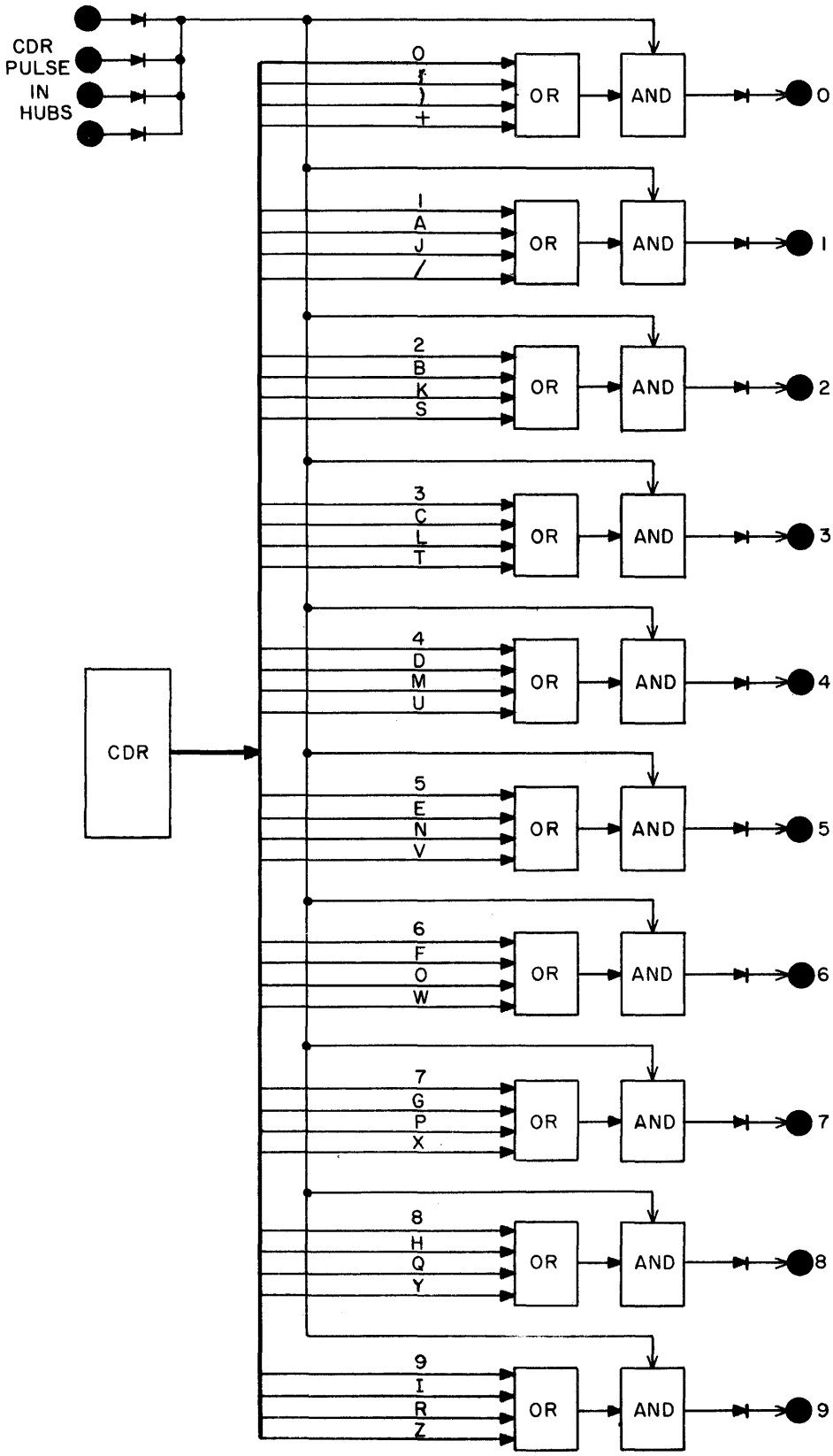


Figure II-8. CD Translator's CDR PULSE IN Circuit

(8) Next Instruction Sub-Step

When a NEXT INSTRUCTION hub (NI) is pulsed, plugboard operations terminate, and Program Control transfers to the internally-stored program for the next instruction. The next instruction executed is an Instruction Word. The NI Sub-Step is thus the means of transferring Program Control from the plugboard to the internally-stored program. Twelve (identical) NI hubs are provided.

j. Central Computer Sub-Steps

Each of the four pairs of hubs associated with the following Sub-Steps are bussed. The pulse source initiating the Sub-Step is patched to one hub of the pair and the other hub of the pair is patched to continue the program, if this is required.

(1) Condition Compare

When the Condition Compare Sub-Step is initiated the next Compare process is modified as follows: Space codes and Zeros are allowed their actual weighted values in Univac code and are compared on that basis. (Normally, the following sets of characters from corresponding positions in V_1 and V_2 are simply ignored.

V_1	0	Δ	0	Δ
V_2	Δ	0	0	Δ
	a	b	c	d

When a Condition Compare Sub-Step is carried out, each of these sets will be compared. Only Sets "c" and "d" will give an equal comparison for the character position. Set "a" will detect $V_1 > V_2$ for the character position; and Set "b"will detect $V_1 < V_2$ for the character position.

(2) Clear BTB To Ignores

When the CLEAR BTB TO IGNORES hub is pulsed, an Ignore Code is placed in each of the 120 character positions of the Block Transfer Buffer, BTB.

Note: The action of Condition Compare and Clear BTB to Ignores Sub-Steps is available to the internal stored program via the Special Character Out (Q-Y) and Breakpoint 1, 2, or 3 Sub-Instructions, provided the hubs these Sub-Instructions cause to emit are patched to a CONDITION COMPARE or CLEAR BTB to IGNORES hub.

k. General Storage Sub-Step Patching

When any of the hubs listed below is pulsed, the correspondingly-named General Storage operation is initiated. These hubs also occur in bussed pairs (four sets). The pulse source is patched to one hub of the pair, and the other hub of the pair is patched to continue the program.

CLEAR GENERAL STORAGE BUFFER TO IGNORES
READ UNIT RECORD
WRITE UNIT RECORD
WRITE UNIT RECORD & CHECK
CHANNEL SEARCH EQUAL
CHANNEL SEARCH UNEQUAL

Each General Storage Sub-Step is identical to the correspondingly-named Sub-Instruction described in Paragraph IIAlc (2) above.

l. I/O Control Sub-Step Patching

The following hubs are involved when I/O Control Sub-Steps are carried out:

Demand Test In (0-9) Sub-Step:

DEMAND TEST IN (0-9)
READY (0-9)
NOT READY (0-9)

Demand In (0-9) Sub-Step:

DEMAND IN (0-9)
DEMAND OUT (0-9)
SPECIAL OUT (0-9)

Related hubs:

(LS) I/O-COMPUTER CONTROL LINE hubs (a-l)
(TEST) HIGH SPEED I/O-COMPUTER CONTROL LINE STORAGE (W, X, Y, Z)
COMPUTER -I/O CONTROL LINES (A-J)
TRACK SWITCH (0-9)

(1) Demand Test In (0-9) Sub-Step

The Demand Test In Sub-Step is used to test the status (ready or not ready) of a particular I/O Unit. It is analogous to the Channel Search Probe Sub-Step used in connection with General Storage in that it permits central computer operations to continue without interruption if an I/O Unit is not ready. The Demand Test In Sub-Step does not affect the demand status (discussed below) of the I/O Unit tested or that of any other I/O Unit.

The Demand Test In Sub-Step is defined by patching a pulse source to the particular DEMAND TEST IN (0-9) hub that corresponds to the I/O Unit whose status is to be tested; and by patching the corresponding READY (0-9) and NOT READY (0-9) hubs to continue the program. When the DEMAND TEST IN (0-9)

hub is pulsed, the status of the associated I/O Unit (0-9) is examined. If the I/O Unit is ready, its associated READY (0-9) hub emits. If the I/O Unit is not ready, its associated NOT READY (0-9) hub emits. Both replies are instantaneous, and the next operation is thus determined immediately on the basis of which one of these two hubs emits.

As explained in Section V, an I/O Unit is ready if it has power, and

is not engaged in a previously initiated operation, or

has no abnormal or error condition (as broken tape, card jam, etc) which would prevent it from functioning properly.

An I/O Unit is not ready if the above conditions are not all present.

The Demand Test In Sub-Step should always be programmed prior to a Demand In Sub-Step unless the latter Sub-Step must be executed at a particular point in a program in order for the program to continue. It is also convenient to program the Demand Test In Sub-Step as a check, prior to Track Switching, to insure that the I/O Unit whose associated I/O Tracks are to be switched is not in the process of transferring data in or out of the computer. This would be a way of Track Switching safely if it is convenient not to use the DEMAND or SPECIAL OUT signals discussed below to track switch at the usual time. In this use of Demand Test In, the READY hub can be bussed to track switch the I/O tracks associated with the I/O Unit tested, and to continue the program. The Demand Test In cannot be used as the only means of determining whether an I/O Unit can receive an I/O Instruction via the COMPUTER-I/O CONTROL LINE hubs (A-J). An I/O Unit must be "on demand" (in addition to being ready) to receive an I/O Instruction (see Demand In Sub-Step below).

(2) Demand In (0-9) Sub-Step

The Demand In Sub-Step is usually used to place a particular I/O Unit "on demand" so that control information can be exchanged between the computer and that I/O Unit. It is also used, however, to place an I/O Unit "on demand" so that a subsequent* storage reference can be made to the I/O Track associated with the I/O Unit via the Plugboard Addressing System.

The Demand In Sub-Step is defined by patching a pulse source to the particular DEMAND IN (0-9) hub that corresponds to the I/O Unit to be placed "on demand"; and by patching the DEMAND OUT and SPECIAL OUT hubs associated with that DEMAND IN (0-9) hub to define the next operation(s).

When one of the DEMAND IN (0-9) hubs is pulsed, the following events occur:

1. High Speed I/O-Computer Control Line Storage (a special-purpose memory described below) is cleared;

* (The I/O Unit must be "on demand" at the time the I/O WORD or FIELD hub in the Plugboard Addressing System is enabled.)

2. every I/O Unit is taken "off demand", and the I/O Unit (0-9) whose associated DEMAND IN (0-9) hub is pulsed is then placed "on demand";

3. a test is made to see if any error condition exists in the I/O Unit placed "on demand". (If an error has occurred, an error signal instead of a DEMAND OUT or SPECIAL OUT discussed below is produced; the Demand In Sub-Step is not completed; and the computer hangs up.)

4. The DEMAND relay in the I/O Unit pulls in. When this relay energizes:

   It allows B+ from the computer to be applied to those (LS) I/O-computer control lines in the I/O Unit that have been programmed on the I/O Unit to energize when the I/O Unit is placed "on demand". The (LS) I/O-Computer control lines so energized then supply B+ to their correspondingly-named (LS) I/O-COMPUTER CONTROL LINE hubs (a-l) on the Program Control Plugboard.

   The coil circuits of all Selectors in the computer using this I/O Unit's Demand Ground control are grounded; and

   The DEMAND (0-9) light associated with this I/O Unit lights on the Console.

   (The DEMAND relay remains pulled in as long as the I/O Unit is "on demand".)

5. The I/O Unit placed "on demand" is conditioned so that should an I/O Instruction be forthcoming from the computer program, that I/O Unit (and only <u>that</u> I/O Unit) will accept the I/O Instruction.

   Note: I/O Instructions are generally sent to an I/O Unit via plugboard-defined programs by pulsing the COMPUTER-I/O CONTROL LINE hubs (A-J) <u>after the I/O Unit has been placed "on demand" and has signified it is ready</u>. When these hubs are pulsed, the Computer-I/O control lines (A-J) signals are applied to every I/O Unit in the system. However, only the I/O Unit "on demand" can receive or accept these signals.

6. The I/O Unit "on demand" is also conditioned so that its (HS) I/O-Computer control lines (W, X, Y, and Z) if energized in the I/O Unit at the time the I/O Unit goes ready will set High Speed I/O-Computer Control Line Storage at that time. Only the I/O Unit "on demand" can cause High Speed I/O-Computer Control Line Storage to be set although this memory is commonly shared by all I/O Units.

If the I/O Unit is not ready when demanded, because it is still engaged in a previously-initiated operation, no further action occurs until the I/O Unit is ready. (If Demand In is programmed after a ready condition is detected by Demand Test In, there is no wait at this point.)

When the I/O Unit "on demand" goes ready:

The READY relay energizes and lights the READY 0-9 light associated with this I/O Unit on the Console.

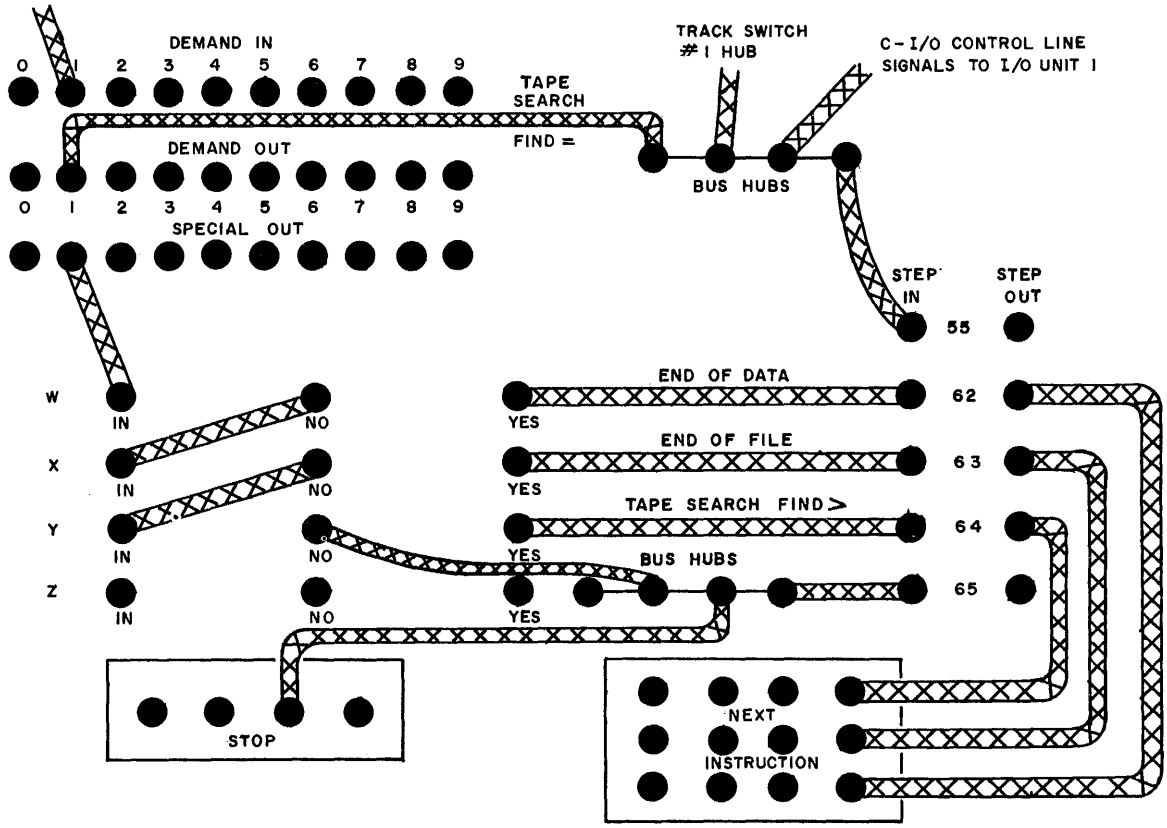
Any (HS) I/O-Computer control lines (W, X, Y, and/or Z) that may be energized* in the I/O Unit at the time the I/O Unit "on demand" goes ready, produce an input to (i.e., set) High Speed I/O-Computer Control Line Storage. The latter is simply a 4-bit memory location which "remembers" the signals received over the (HS) I/O-Computer control lines (W, X, Y, and/or Z).

If any one, or any combination, of the (HS) I/O-Computer control lines is energized when the I/O-Unit "on demand" goes ready, a SPECIAL OUT signal is produced by the I/O Unit. This signal is sent to the SPECIAL OUT (0-9) hub associated with the I/O Unit. Among other things SPECIAL OUT patching is always used to test which condition (or combination of conditions) W, X, Y, or Z was set up in High Speed I/O-Computer Control Line Storage. One method of patching SPECIAL OUT so that the inputs to High Speed I/O-Computer Control Line Storage can be tested is illustrated and explained in Figure II-9, Page II-77.

If no (HS) I/O-Computer control lines are energized when the I/O Unit "on demand" goes ready, a DEMAND OUT signal is produced by the I/O Unit. This signal is sent to the DEMAND OUT (0-9) hub associated with the I/O Unit. Figure II-9, Page II-77 also illustrates one way the DEMAND OUT signal can be used by DEMAND OUT patching.

(Figure II-9 is only a specific use of the DEMAND OUT and SPECIAL OUT signals. Either DEMAND OUT or SPECIAL OUT can be patched to (among other things) Track Switch or send an I/O Instruction to an I/O Unit. Only SPECIAL OUT is used, however, to test High Speed I/O-Computer Control Line Storage.)

* These lines are "programmed to be energized" in the I/O Unit in the same basic manner as the (LS) I/O-Computer control lines (a-l).



PROGRAM STEP	PATCHING					
	STEP IN	PROC	V_1 ADR	V_1 SH	R ADR	STEP OUT
62	when W (End of Data) is found	AT	129	0	PAK	NI (Start at 143)
63	when X (End of File) is found	AT	129	3R	PAK	NI (Start at 167)
64	when Y (Channel Search Find) is found	AT	129	6R	PAK	NI (Start at 150)

The I/O Unit operation assumed in the above patching is Tape Search ≥ . The program treats Tape Search Find = as the expected result (DEMAND OUT) and the UFC Magnetic Tape Unit involved is programmed (by patchcord wiring on its control panel) to notify the computer (via SPECIAL OUT and the setting of IIS I/O-C CLS) of End of Data by W, End of File by X, Tape Search Find > by Y. (For simplicity Z is not used in this example, nor are the other uses of W, X, Y, and Z that can be involved for other I/O Units shown.)

Program Step #55 is the beginning of the (main) sub-routine which processes data from Tape Search Find =. Program Steps 62-64 are each an AT instruction which takes a code word (ΔΔ150167143+) stored at memory location 129, shifts that word as required, loads PAK with the appropriate starting address of the sub-routine which is to be initiated when W, X, and Y are each detected, and then pulses an NI hub.

Figure II-9. An Example of DEMAND and SPECIAL OUT Patching (UFC Magnetic Tape Unit assumed for I/O Unit #1)

(3) Plugboard References to I/O Tracks via I/O WORD and FIELD hubs

While an I/O Unit is "on demand" the following occur if an I/O Track Address is specified by V_1 ADDRESS, V_2 ADDRESS or R ADDRESS patching:

when the I/O WORD (0-9) or FIELD (A-V) hub is enabled, the highest-order stage of SAR is set to 0 and the lowest-order stage of SAR is set to the number or letter corresponding to the hub enabled;

the demand status of each I/O Unit is tested; and

the I/O Unit "on demand" then sends a signal to SAR which sets the middle digit position of SAR to the same number as that assigned to the I/O Unit.

If a Storage reference to an I/O Track is to be defined via the Plugboard Addressing System, therefore, the I/O Unit associated with the I/O Track Address desired must first be placed "on demand" (and be "on demand" when the I/O WORD 0-9 or FIELD A-V hub is enabled).

(4) COMPUTER-I/O CONTROL LINE hubs (A-J)

The Computer-I/O control lines (A-J) of each I/O Unit are all connected to the COMPUTER-I/O CONTROL LINE hubs (A-J). When an I/O Unit is placed "on demand" that and only that I/O Unit's Computer-I/O control lines will accept and receive signals from those hubs. The combination of signals received defines an I/O Instruction for the I/O Unit "on demand".

The COMPUTER-I/O CONTROL LINE hubs (A-J) should never be pulsed unless the particular I/O Unit that is to receive the I/O Instruction is "on demand" and ready. If they are pulsed when the desired I/O Unit is not "on demand" that I/O Unit cannot receive the I/O Instruction. If they are pulsed when a I/O Unit "on demand" is not ready, the I/O Instruction received is discarded. For this reason the DEMAND OUT and SPECIAL OUT signals are usually used to send an I/O Unit an I/O Instruction via the COMPUTER-I/O CONTROL LINE hubs (A-J). When DEMAND OUT or SPECIAL OUT are produced by an I/O Unit that I/O Unit must be "on demand" and ready.

(5) TRACK SWITCH hubs (0-9)

Each I/O Track is actually a pair of tracks: at any given time, one track of the pair is used by the computer and the other track of the pair is used by the I/O Unit assigned to that I/O Track Address. In this way the computer and I/O Unit can time share data transmissions to and from the same I/O Track Address. When the TRACK SWITCH hub (0-9) associated with an I/O Unit is pulsed, the Track Switching circuitry in that I/O Unit's Demand Station is operated: the track connected to the computer is made available to the I/O Unit and what was formerly the I/O's Unit's track becomes the computer's track.

An I/O Unit need not be "on demand" or ready for track switching to occur. Track Switch should not be programmed, however, unless the I/O Unit associated with the I/O Tracks switched is ready. If Track Switching is programmed when an I/O Unit is not ready, the I/O Track may (depending on why the I/O Unit is not ready) be switched while the I/O Unit is engaged in data transmission to and from its I/O track. The data on both I/O Tracks will then be garbled.

m. Pulse Sources that Emit under Control of Internally-Stored Program

BREAKPOINT 1 hub
BREAKPOINT 2 hub
BREAKPOINT 3 hub
SPECIAL CHARACTER OUT (Q-Y) hubs

The Sub-Instructions that cause these hubs to emit are discussed in Paragraphs IIA1(3).

n. ERROR hub Patching

For the following reasons the UFC Model 1 will produce a signal on the Program Control Plugboard, indicating an error condition in the system:

Parity Error (Central Computer)	
Divide Overflow	Arithmetic Errors
Add-Subtract Overflow	
Normalize Overflow	
Arithmetic Check Error	
General Storage Program Error	

Parity Error: each character transmitted from one register or memory location to another, whether this be a programmed storage reference or an automatic reference carried out as part of the execution of an instruction, is checked by various parity-checking circuits. If any check detects an even number of (binary) "1's" in any character transmitted: computer operation stops, the PARITY ERROR hubs on the Program Control Plugboard emit, and the specific parity-checking circuit which detected the error is indicated.

Arithmetic Errors:

Divide Overflow — If improper V_1 and/or V_2 shifts are programmed in an instruction which performs the Divide process, the number of quotient digits that would be formed if the division were performed would exceed 11 digits and sign. A quotient greater than 11 digits and sign would exceed (or "overflow"), the capacity of Register D. Hence, when improper V_1 and/or V_2 shifts are detected in a Divide process, computer operation stops, the DIVIDE OVERFLOW hubs on the Program Control Plugboard emit, and the "overflow" error condition is indicated.

Add/Subtract Overflow — If the number of digits in the result of an Add or Subtract exceeds 11 (i.e., a "carry" past the highest-order digit position of Registers C and D occurs), Computer operation stops, the ADD/SUBTRACT OVERFLOW hubs on the Program Control Plugboard emit, and the "overflow" error is indicated. Note in this case that the "overflow" sum (difference) is actually formed.

Normalize Overflow — If during a Left Normalize Process a V_1 operand equal to zero* is detected, computer operation stops, the NORMALIZE OVERFLOW hub emits, and the "overflow" error is indicated.

Arithmetic Check Error — Checks can be programmed for the following arithmetic operations:

* See RULES FOR LEFT NORMALIZE INSTRUCTIONS, Paragraph IID

Add
Subtract
Multiply, Store Lower
Multiply, Store Upper
Divide, Store Quotient
Divide, Store Remainder

If the result of one of the above processes does not prove when checked, computer operation stops, the ARITHMETIC CHECK ERROR hubs on the Program Control Plugboard emit, and the check error is indicated.

General Storage Program Errors: The following programming errors are automatically detected by the General Storage System when they occur:

α -Zone: when any character with zone bits other than 00 is sent to any stage of GSAR

Inactive Drum Section: when a drum section is referred to by a General Storage Address and no such section exists in the system.

Odd Angular Address: when the two lower-order stages of GSAR store a (two-digit) odd number.

Unit Record Identifiers All Ignores: If a channel search operation is attempted with a Unit Record Identifier which contains nothing but Ignore Codes.

Since the above errors all cause a (pair of) hub(s) on the Program Control Plugboard to emit, patchcord wiring of the error hubs can be used or not for error recovery or error analysis, depending on what the error is, the nature of the program, and whether recovery is desired or not in each case.

(1) Error hubs left unpatched

If the error hubs that emit are left unpatched, the computer hangs up. In the case of all errors associated with the PARITY, OVERFLOW, and CHECK hubs, computer operation stops immediately. If a GS PROGRAM error occurs, the computer hangs up on OED 3 of the next instruction. Operation cannot be resumed until manual controls are operated.

(2) STEP REPEAT Patching

Step Repeat is a group of circuits which enable Central Computer operation to continue even though certain machine errors have occurred. Four STEP REPEAT hubs are provided. When a STEP REPEAT hub is patched from an ERROR hub, the instruction during which the error occurred is automatically repeated when the ERROR hub emits. the PARITY and CHECK ERROR hubs are generally wired to a STEP REPEAT hub; this will insure automatic recovery from all errors due to temporary or momentary machine failure. If the error persists, computer operation should be stopped by pressing the STOP button.

Note: For STEP REPEAT to be effective in those cases where a parity error occurs storing R, the programmer must not use the same location as the source of either or both operands and as the Destination for the result.

(3) STEP CLEAR

Step Clear is a group of circuits which can be used to continue computer operation even though a programming error has been detected. Step Clear does not allow the (main) computer program to continue, since this, in general, is not practical if an error has been detected; however, it does permit an auxiliary plugboard sequence designed for pinpointing the cause of an error to be initiated. This sequence, for example, might type out the contents of certain registers and other memory locations to enable the programmer to find the instruction in which the error occurred.

The O'FLOW hubs can be patched to one of the STEP CLEAR hubs to remove the instruction currently in Program Control, when those error hubs emit; and the OUT hub of a Step Clear can be patched (directly or via variance hubs) to the STEP IN of the first Program Step in an error-analysis plugboard sequence. When the ERROR hubs emit, the error-analysis sequence is automatically begun. Since programming errors require correction procedures involving the operation of manual controls, the error-analysis plugboard sequence will usually terminate by pulsing a STOP hub. The GS PROGRAM error can also be patched in this manner but the error analysis possible is frequently limited because a relationship may be difficult to establish between the General Storage operation during which the error occurred and the Program Step at which the computer stopped operation.* Four sets of Step Clear hubs (IN and OUT) are provided on the Program Control Plugboard.

In addition to the above error conditions which, when detected, emit a pulse on the Program Control Plugboard, provision is also made for detecting each of the errors listed below. When each of these errors occur, the computer hangs up and the attention of maintenance personnel is required.

Timing Pulse Distributor Error (if generation of system timing pulses from High Speed Drum is faulty)

Parity Errors in General Storage Operations*(if all characters transmitted during data transmissions in General Storage operations are not parity correct)*

Write Unit Record and Check Error (if the Unit Record that should have been recorded was not)*

Unit Record Length Error (if a combination of Unit Record Length and starting Angular Address are programmed which would extend a General Storage operation into a Channel's Search Control Location, or the locating circuitry associated with angular addresses fails)*

* When an error occurs in General Storage operations, the particular error is indicated on the General Storage Control Cabinet, the General Storage operation terminates (or is not begun), and the computer stops operation on OED 3 of the next instruction.

Demand Station (0-9) Errors: (if an error condition, already set up in an I/O Unit, is detected when the I/O Unit is placed "on demand".)*

o. Miscellaneous Hubs

(1) PROGRAM INDICATOR Lights (1-6)

There are six PROGRAM INDICATOR Lights (1-6) on both the Program Control Cabinet #1 and on the Console. Each of these Program INDICATOR Lights is illuminated by applying B + (via patchcord wiring) to the corresponding INDICATOR (1-6) hub on the Program Control Plugboard. Current can be applied to these hubs in a variety of ways; for example, by a combination of Selector Hold (B +) and Alternate Switch patching; or, as is usually the case, by patchcord wiring involving a combination of the Indicator Switch, Selector Hold (B +) and another Selector. In the former case, PROGRAM INDICATOR Light patching can be used as a visual indication of which one of a number of plugboard-defined programs, or discrete sub-programs, is being executed at any particular time. In the latter case, the PROGRAM INDICATOR lights can be used to indicate the sequential progress of a program.

(2) INDICATOR SWITCH

The Indicator Switch is fundamentally a pair of pushbutton switches: one on the Program Control Cabinet #1 and one on the Console. These pushbutton switches are both labeled CLEAR and are connected in series, as shown in Figure II-10 below. Their connective wiring is simply a bus which terminates on The Program Control Plugboard at the INDICATOR SWITCH hubs.

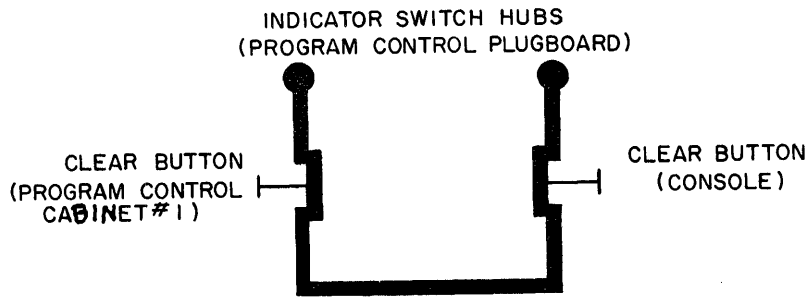


Figure II-10 Indicator Switch

The Indicator Switch is normally closed as shown in the above diagram. If either CLEAR button is depressed, the Indicator Switch is opened as long as the CLEAR button is being pressed, but will return to its closed state when the CLEAR button is released. The CLEAR buttons thus function to temporarily open the Indicator Switch.

* When an error occurs in an I/O Unit, the particular error is indicated on the I/O Unit, the I/O Unit operation stops, the I/O Unit's associated Demand Station (0-9) ERROR light on the Console is lit, and the error condition is remembered in the I/O Unit. When the I/O Unit is demanded, computer operation stops immediately.

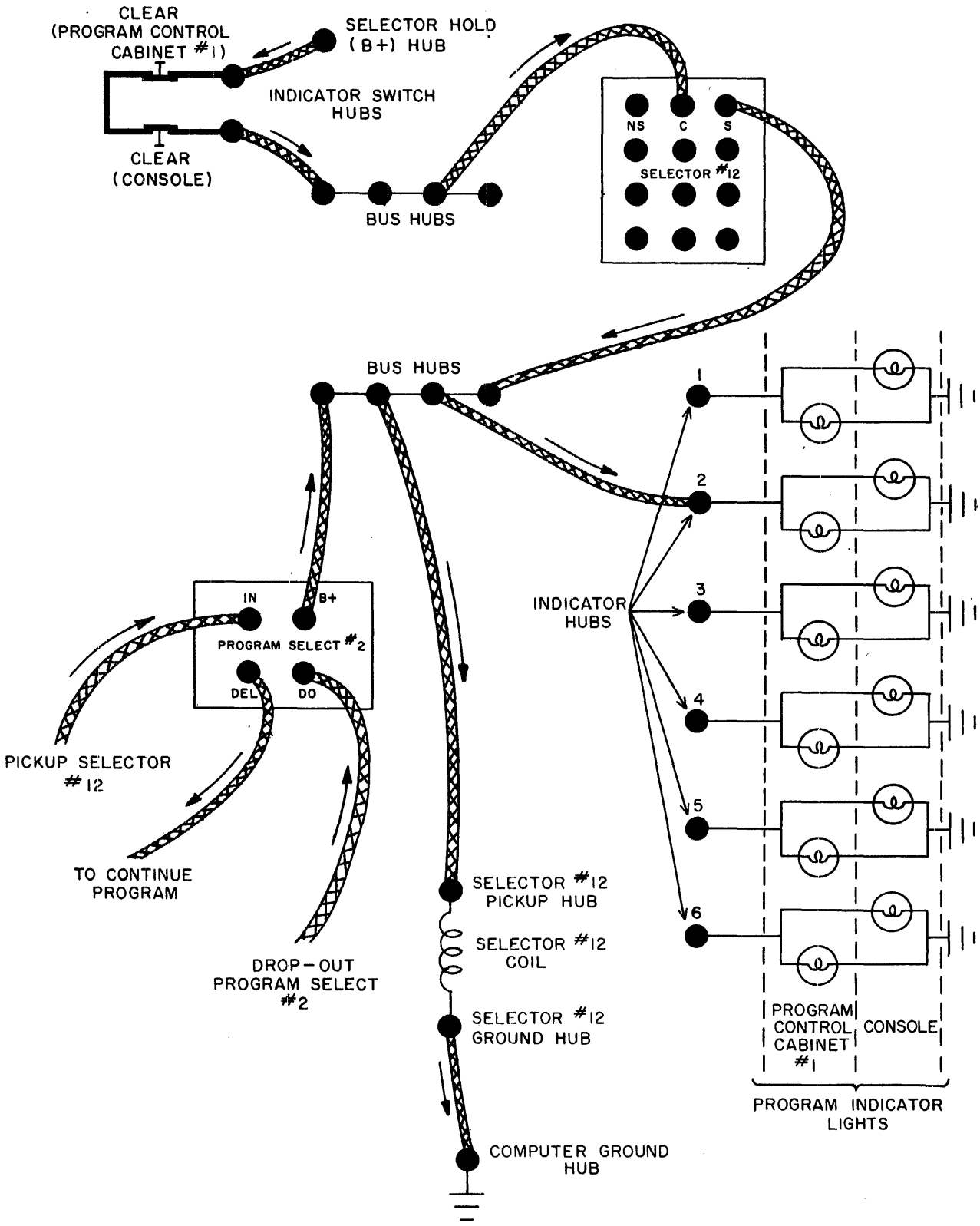


Figure II-11. Use of Indicator Switch in conjunction with PROGRAM INDICATOR Lights.

A typical application of the use of an Indicator Switch in conjunction with the operation of the PROGRAM INDICATOR lights is illustrated in Figure II-11 . In this example, PROGRAM INDICATOR light #2 is a visual display of those portions of the program wherein Selector #12 is in its Select state.

The sequence of events that occur for the example shown in Figure II-11 is:

1. The IN hub of Program Select #2 is pulsed and the OUT hub (B+) emits current. This current is applied to the SELECTOR #12 PICKUP hub and to the INDICATOR #2 hub. Selector #12 is thus set to its Select side, and the PROGRAM INDICATOR light #2 is illuminated to indicate this.

2. With Selector #12 in its Select state, Selector Hold (B+) can then be applied to both the SELECTOR #12 PICKUP hub (and Selector #12 will remain in its Select state even though Program Select #2 is dropped out) and to the INDICATOR #2 hub (keeping PROGRAM INDICATOR #2 lit.)

3. Selector #12 will then remain Select until one of the CLEAR buttons associated with the Indicator Switch is pressed or machine power is turned off; i.e., until Selector Hold (B+) power is removed. When Selector Hold (B+) power is removed, Selector #12 immediately reverts to its Non-Select state, and PROGRAM INDICATOR Light #2 is extinguished. To again illuminate PROGRAM INDICATOR light #2, the IN hub of Program Select #2 must receive another signal. The above sequence of events is then repeated.

(3) BUS hubs.

BUS hubs are used as facility hubs for wiring two or more out hubs not requiring isolation to one or more destinations. Buses can also be used to expand a pulse out, B+ out, or ground. Buses should never be used to expand an enable out. Many sets are provided on the Program Control Plugboard. Some are groups of 3 and 4 hubs; others are groups of 5 and 6 hubs. These groups can be patched together to form larger groups.

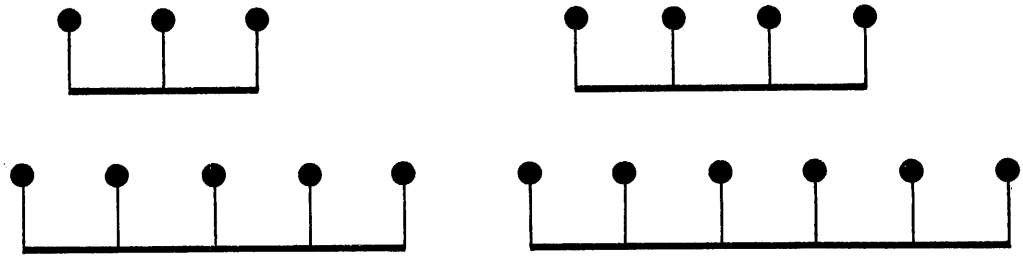


Figure II-12 BUS hubs and buses

(4) UNI-BUS hubs.

The UNI-BUS hubs are a set of five hubs tied to a bus: four IN hubs and an OUT hub.

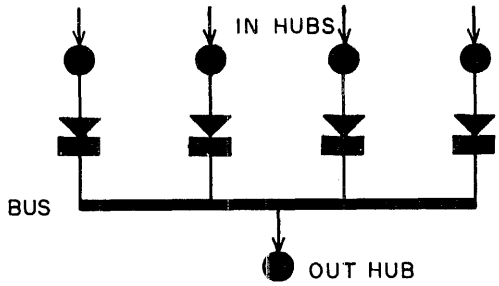


Figure II-13. UNI-BUS hubs

The IN hubs are diode-protected to prevent back circuits. These hubs are used where 2, 3, or 4 pulse sources are to be patched to a common destination and each source also is to produce results peculiar to its own occurrence. Each pulse source is patched to a separate bus. One BUS hub from each bus is patched to a UNI-BUS IN hub. (The other BUS hubs of each bus are patched to achieve the results peculiar to each pulse source.) The UNI-BUS OUT hub is patched to the common destination. Eight sets of UNI-BUS hubs are provided.

(5) OUT Expanders

The OUT EXPANDER hubs (one IN and two OUT hubs connected internally as shown below) are used where a single pulse source is to be patched to several destinations and the pulse source must be amplified (and multiplied) to safely pulse all destinations.

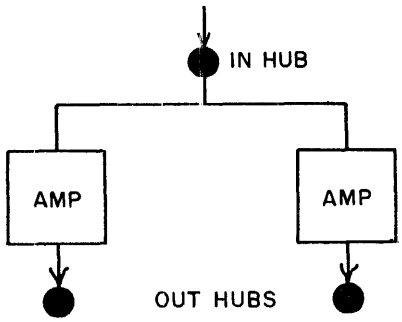


Figure II- 14 OUT EXPANDER hubs

When the IN hub is pulsed, the OUT hubs emit (note that the OUT signals are amplified.) Eight sets of OUT EXPANDER hubs are provided.

(6) START hub

The START hub is a pulse-out hub that emits when the START button is pressed after the MASTER CLEAR button is pressed. It is patched to a STEP IN hub to begin a program from the plugboard. It is patched to a NEXT INSTRUCTION hub (NI) to start internally stored programs. If left unpatched, no program is initiated when the MASTER CLEAR and START buttons are pressed.

(7) STOP hubs

The STOP hub is a pulse-in hub which, when pulsed, will stop computer operation on OED 3. If the pulse source pulsing the STOP hub is also patched to a STEP IN hub, computer operation stops with Program Control "conditioned" to resume operation from the plugboard when the START button is pressed. (The first instruction executed when operation is resumed is the Program Step whose STEP IN was pulsed as noted above.) If the pulse source pulsing the STOP hub is not also patched to a STEP IN hub, computer operation stops with the correct next Instruction Word set up in Program Control. When the START button is pressed, the first instruction executed is that Instruction Word. See Figure II-20, Page II-104.

p. General Rules for Connecting Various Hubs on the Program Control Plugboard by Patchcords

(1) General Information

There are, fundamentally, eight different types of hubs on the Program Control Plugboard. (See Figures II-15 through II-18, Pages II-90 through II-93.)

(a) Pulse In (green): These hubs receive a pulse

(b) Pulse Out (red): These hubs emit a pulse

(c) Enable In (yellow): These hubs receive an enable

(d) Enable Out (blue)*: These hubs emit an enable

(e) B+ In (brown): These hubs receive B+

(f) B+ Out (black): These hubs are B+ sources

(g) Ground Potential (orange): These hubs are only used to ground the coil circuits of Selectors.

(h) Multi-purpose hubs (not color coded). The hubs discussed below do not have any signal requirement as such; i.e., they can be used either for pulses, enables, ground, or B+ . They are not, therefore,

* The enable outs of a Program Step are shown in a combination blue-yellow code to emphasize the facility of "chain wiring" available for use in conjunction with those hubs.

color-coded in Figures II-15 through II-18)

(1) BUS hubs

Plugboard buses have two general functions:

(a) they are the tie point for one or more outs (sources) of the same type, each of which is to produce the same effect.

In this use of buses, the out hubs involved are each patched to a separate BUS hub, and another BUS hub of the set is patched to the destination which is to be commonly defined by the outs.

(b) they are a means of patching a pulse out, B+ out, or ground hub to various destinations.

In this use of buses, the out hub is patched to one BUS hub, and the various destinations to which the out is to be routed are patched from the other BUS hubs. (When an out is patched in this manner to two destinations, it is said to be "Y-wired" to those destinations). Enable outs must not be patched to more than one destination.

Note: 3-hub, 4-hub, 5-hub, and 6-hub buses are provided. Larger groups can be formed by simply connecting a BUS hub from one bus to that of another, etc. by patchcords, or "jumper" plugs.

(2) Selector and Alternate Switch COMMON, SELECT, and NON-SELECT hubs.

For bi-directional routing, the COMMON hub is used as an in hub, and the SELECT and NON-SELECT hubs are used as out hubs, one or the other of which emits, depending on how the Selector is set. For "on-off" control, the SELECT and NON-SELECT hubs can be used as in hubs. In this use only the COMMON is an out. An in at the SELECT hub will cause the COMMON hub to emit only if the SELECTOR is in its Select state. Similarly an in at the NON-SELECT hub will cause the COMMON to emit only if the Selector is in its Non-Select state. Another "on-off" use of Selectors is to use the COMMON as an in and only the SELECT or NON-SELECT as an out.

(2) Fundamental Plugboard Programming Rules

1. Pulse outs (red) are patched only to pulse ins (green). This can be done directly, or via buses, Selectors or Alternate Switches.

2. Enable outs (blue) are patched only to enable ins (yellow). This can be done directly, or via buses, Selectors or Alternate Switches.

3. B+ outs (black) are patched only to B+ ins (brown). This can be done directly, or via buses, Selectors or Alternate Switches.

Note: If a B+ out hub or an enable out hub is patched to a pulse in or a ground potential hub, or if a B+ out hub is patched to an enable in hub, serious harm is done to the computer circuitry.

4. A bus can tie together separately either pulse outs, or enable outs, or B+ outs, or ground supplies. Do not use the same bus for (identical) outs unless each out is to produce the same effect. Keep in mind that, if a bus is used to expand an out, all the results of the out from that bus are produced each time (or as long as, in the case of B+) the out emits. Enable outs must not be expanded by a bus; i.e., do not patch an enable out to two or more destinations. Whenever possible use "chain" wiring rather than buses for enable outs specifying a common destination.

5. For each Selector (1-48) whose Select state is to be used in a program, the corresponding SELECTOR GROUND (1-48) hub must be patched to either a COMPUTER GROUND hub (this unconditionally grounds the corresponding Selector's coil circuit), or to a DEMAND GROUND (0-9) hub (this grounds the Selector's coil circuit only when the associated I/O Unit 0-9 is "on demand"); and the corresponding SELECTOR PICKUP (1-48) hub must be patched (directly or via buses, Selectors, or Alternate Switches) from a B+ out.

6. Alternate Switches are not operated by the program. Once an operator specifies the setting (Select or Non-Select) of an Alternate Switch, that setting is maintained until the operator gives the Alternate Switch an opposite setting.

7. Use of Uni-Buses

When two, three, or four pulse outs are to produce a common result, but each must also produce one or more results peculiar to itself:

patch each pulse out to a separate bus;

patch one hub from each bus to one of the four UNI-BUS IN hubs;

patch the other hubs of each bus to the (pulse) in hubs necessary to get the special results of each pulse out; and

patch the UNI-BUS OUT hub to the pulse in hub that is to be commonly signalled by the pulse outs.

Note: Psuedo-Uni-buses (i.e., "Uni-buses" constructed from a bus by employing all but one BUS hub as in hubs and inserting diodes in the patchcords to these "in" hubs) are not authorized. No guarantee can be given that programs using buses in this way will run; and if incorrect diodes are used, serious harm can be done to the computer circuitry.

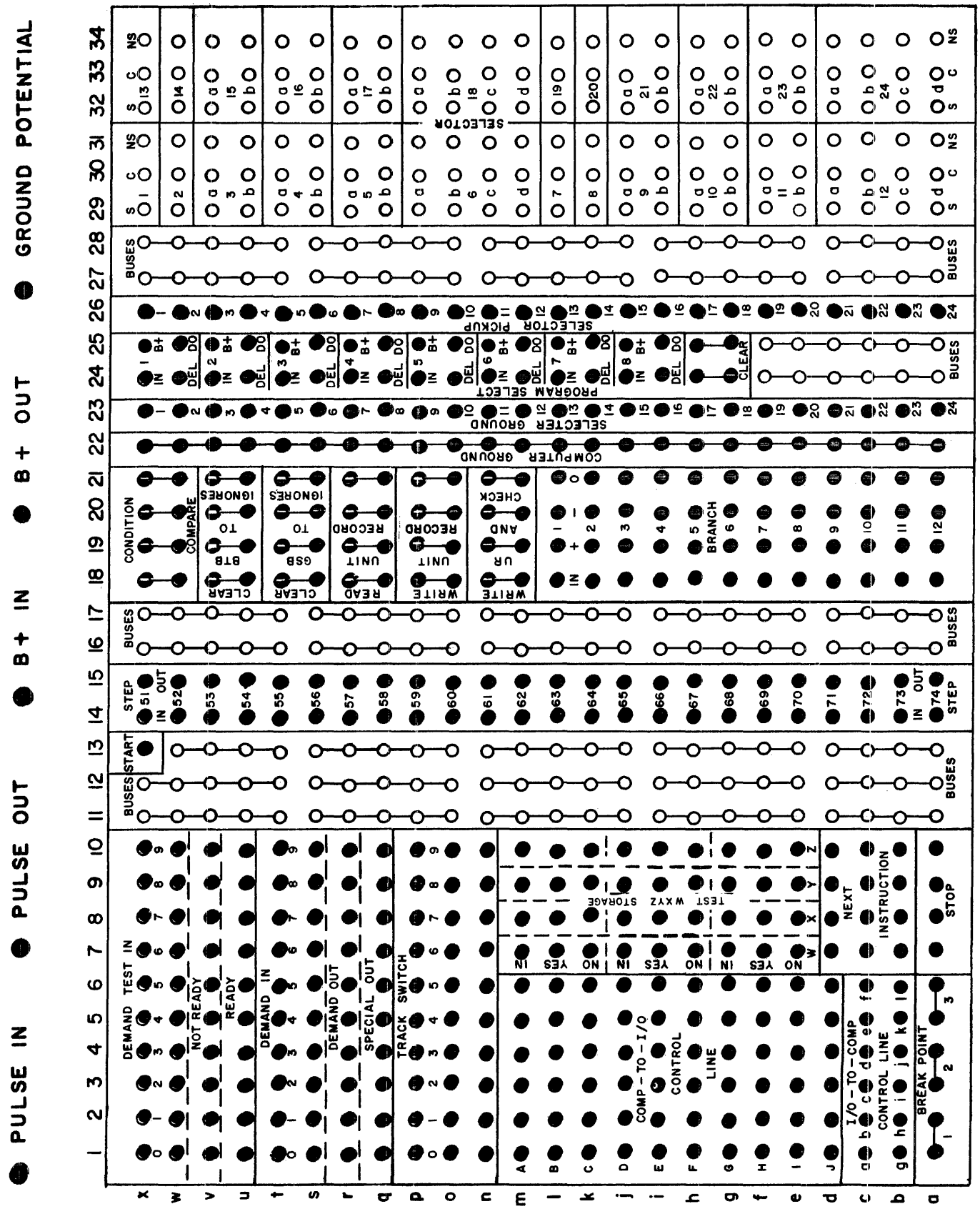


Figure II-15. Upper Left Quadrant of Program Control Plugboard

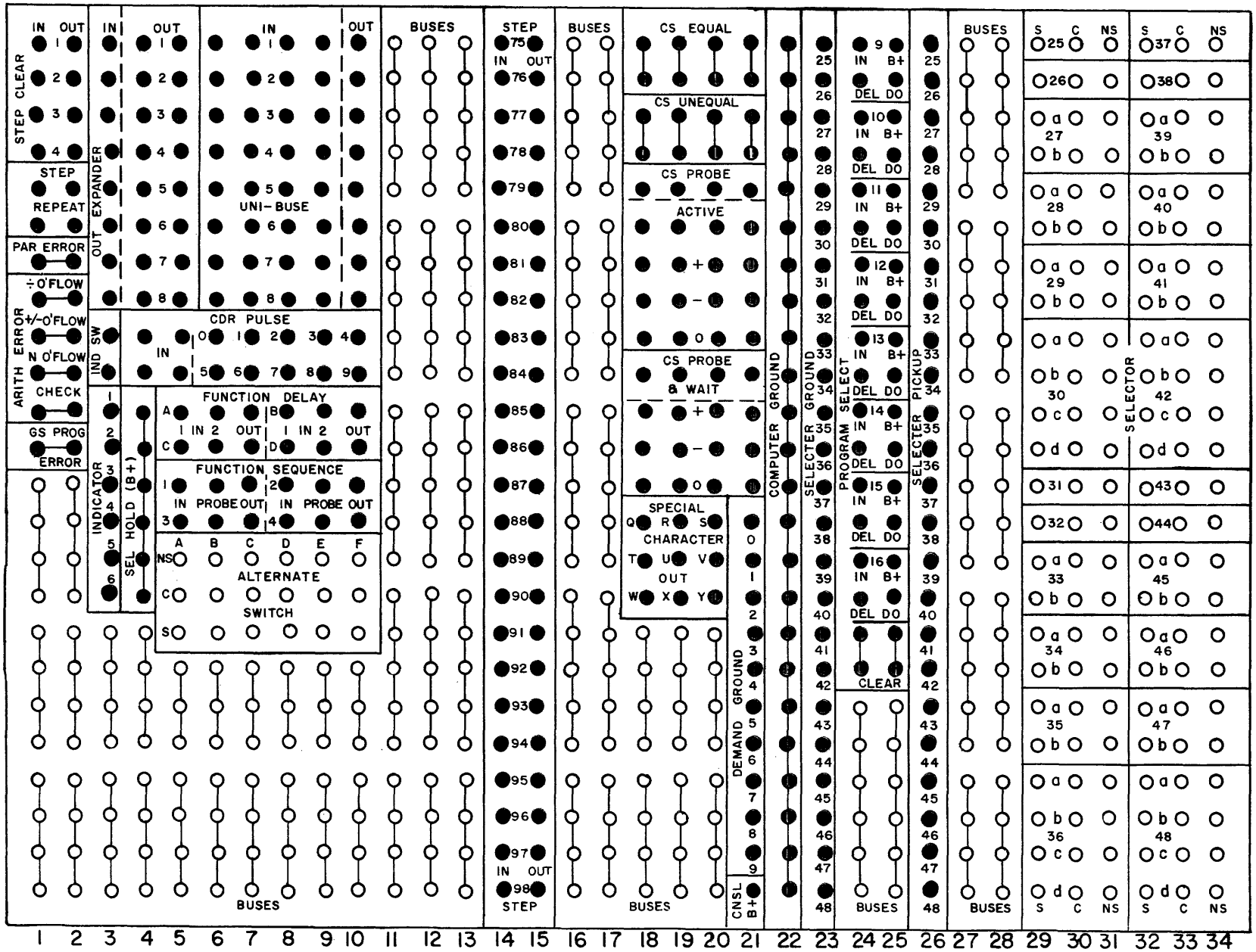


Figure II-16. Lower Left Quadrant of Program Control Plugboard

The combination green-red hubs are actually bused pulse in hubs; either hub can be patched from a pulse source, and the other patched to define the next operation.

The combination purple-black hubs are connected by a switch-controlled bus; either can function as an in and the other as out for B+.

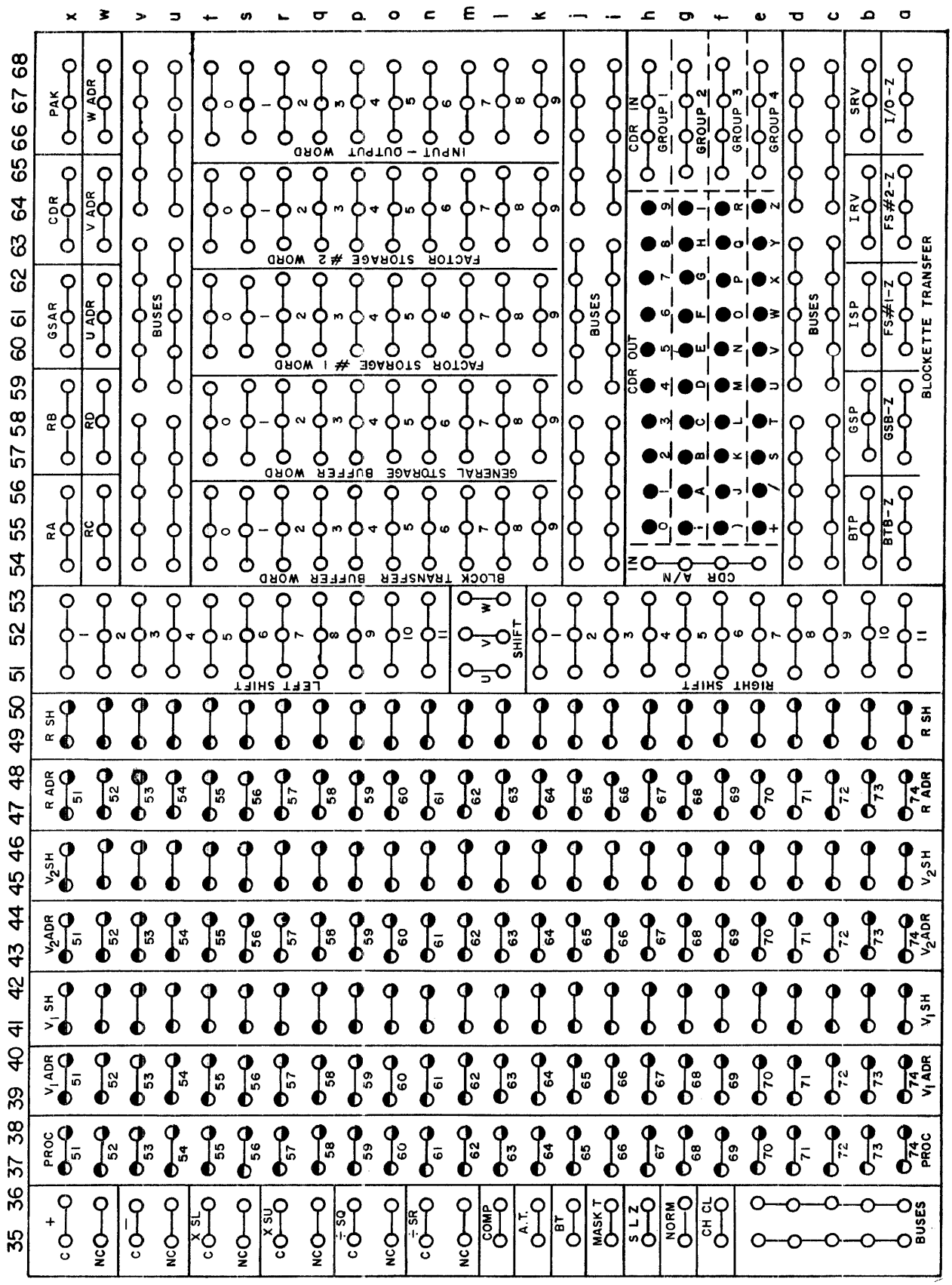


Figure II-17. Upper Right Quadrant of Program Control Plugboard

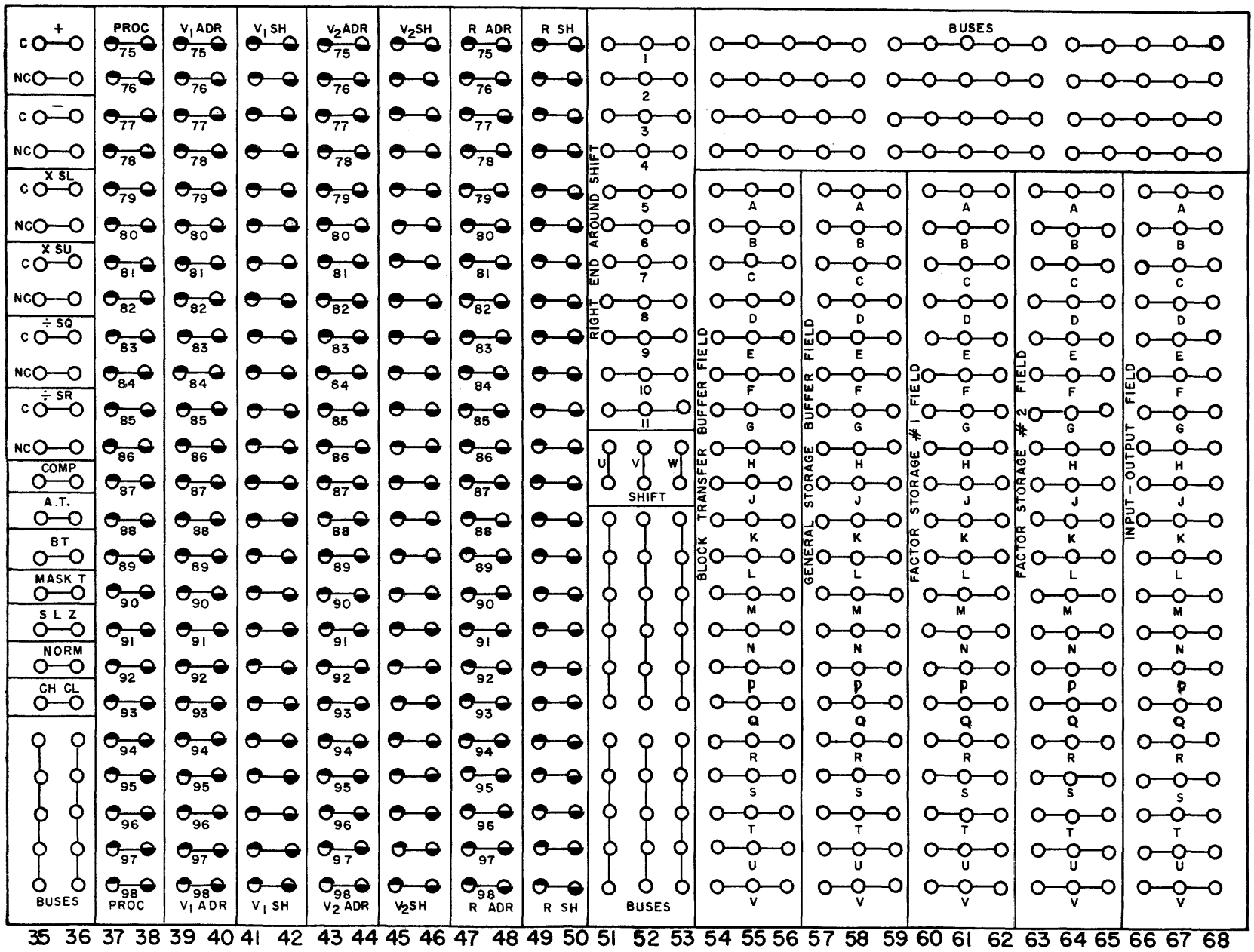


Figure II-18. Lower Right Quadrant of Program Control Plugboard

The combination blue-yellow hubs are actually bused enable out hubs; they are shown in combination color code because these hubs can be "chain wired".

(3) Predicted Relationships for Pulse Out-Pulse In, Enable Out-Enable In, and B+ Out-B+ In Patching

(a) Pulse Out-Pulse In Patching

In Table II-13 Page II-95, each type of pulse out hub is listed alphabetically on the left with the number of "drive units" its associated pulse source produces. Each type of pulse in hub is listed on the right with the number of "drive units" its associated circuitry requires to produce the effect specified by pulsing that pulse in hub. The pulse in hubs are also arranged alphabetically. Note that several pulse in hubs require 2 or 3 "drive units". The basic rule for pulse out-pulse in patching is that the number of "drive units" used by the pulse in hubs to which a pulse out is patched must be less than, or equal to, the "drive units" produced by the pulse source associated with the pulse out. Out Expanders must be used if the number of "drive units" required to perform certain functions by a pulse out exceeds the "drive unit" production of the pulse out's source. Note that each Out Expander requires two "drive units". Uni-bus INS do not require any "drive units". A Uni-bus is merely an isolating device.

A pulse out can be routed through up to 10 Selector contacts and still produce the "drive units" listed.

Any number of Decision-Making Devices for Pulses that receive a pulse and generate a pulse of themselves can be used in sequence, provided the basic rule of pulse out-pulse in patching is observed.

No more than 10 pulse outs should be patched to the same pulse in.

(b) Enable Out-Enable In Patching

Enable outs can be routed through any number of Selector contacts.

The same enable out must never be patched to two enable ins.

No limit is anticipated on the number of enable outs that can be patched to an enable in.

(c) B+ Out-B+ In Patching

B+ can be used to pull all 48 Selectors if necessary.

B+ can be routed through any number of Selectors or bused in any manner, as required.

Table II-13. "Pulse Drives"

PULSE OUT HUBS	"DRIVE UNITS" EACH PRODUCES
+/− OVERFLOW	10
Branch OUTS (+, −, 0)	5 (each)
BREAKPOINT (1, 2, 3)	10 (each)
CS Probe OUTS (ACTIVE, +, −, 0)	5 (each)
CS Probe & Wait OUTS (+, −, 0)	5 (each)
CHECK ERROR	10
CDR PULSE OUTS (0-9)	5 (each)
DEMAND OUT	5
÷ OVERFLOW	10
Function Delay OUT	10
Function Sequence OUT	10
GS PROGRAM ERROR	10
N OVERFLOW	10
NOT READY	5
Out Expander OUTS	10 (each)
PARITY ERROR	10
Program Select DELAYED OUT	10
READY	5
SPECIAL CHARACTER OUTS (Q-Y)	5 (each)
SPECIAL OUT	5
START	10
Step Clear OUT	10
STEP OUT	5
W, X, Y, Z OUTS (YES, NO)	5 (each)

PULSE IN HUBS	"DRIVE UNITS" EACH REQUIRES
Branch IN	3
CS =	1
CS ≠	1
CS Probe IN	2
CS Probe & Wait IN	1
CLEAR BTB TO IGNORES	1
CLEAR GSB TO IGNORES	1
CDR PULSE IN	1
COMPUTER - I/O CONTROL LINES (A-J)	1 (each)
CONDITION COMPARE	1
DEMAND IN	1
DEMAND TEST IN	1
Function Delay INS (1, 2)	1 (each)
Function Sequence INS (SET, PROBE)	1 (each)
NI	1
Out Expander IN	2
Program Select IN	3
Program Select DROP OUT	1
(Program Selects') CLEAR	2
READ UR	1
Step Clear IN	1
STEP IN	2
STEP REPEAT	1
STOP	3
TRACK SWITCH	1
WRITE UR	1
WRITE UR & CHECK	1
W, X, Y, Z INS	2 (each)

2. Interpretation, Execution, and Sequencing of Program Steps.

Table II-14 Page II-98 outlines the manner in which Program Steps are interpreted, executed, and sequenced. Note the following principal differences between the interpretation, execution, and sequencing of Program Steps and that of Instruction Words:

> When a Program Step's STEP IN hub is pulsed, that Program Step is "acquired"; an enable identifying that Program Step's process is supplied to PCT; and that program Step's location 51-98 is set up in PR. Program Control can thus "remember" which Program Step is being executed and as OED advances can supply the enable outs and STEP OUT signal for that particular Program Step. PAK and IRVn are, therefore, not used to acquire the next Program Step as the current Program Step is executed; and PAK does not sequence plugboard-defined programs. Plugboard-defined programs are sequenced by STEP OUT-STEP IN patching.
>
> Addresses and shifts can be defined either by enabling a hub on the plugboard which will set SAR or SK; or one of the U, V, and W, ADDRESS and SHIFT hubs can be involved, in which case IRVc and SRV send SAR and SK, respectively, the required address and shift. IRVc and SRV can therefore be used, as required, in Program Steps. When used their operation is the same as when used during the execution of Instruction Words.

OED will carry out the same basic cycle (3-11) for a process specified by a Program Step as it does when that process is specified by an Instruction Word. This can be verified by examining the OED cycle for the different processes defined by a Program Step given in Paragraph IID.

TAC and SK perform the same function during the execution of a Program Step as during the execution of an Instruction Word; and the functions of SAR are also the same with the following exceptions:

> SAR, is not used for storing "abc" or "xxx" in I/O Control operations as in the execution of Instruction Words; and
>
> the middle digit for an I/O Track Address must be formed in SAR by placing the I/O Unit associated with the I/O Track to be referred to "on demand", prior to enabling an I/O WORD (0-9) or FIELD (A-V) hub in a Program Step.

PCT operates in much the same fashion also, and controls the times at which the basic hubs of the Program Step emit.

C. COMBINATION INTERNALLY-STORED/PLUGBOARD DEFINED PROGRAMS.

1. Initial Starts:

When the MASTER CLEAR button is pressed, OED is set to 12. When the START button is then pressed, the START hub emits, a wait is generated, OED is set to 13, and a computer program can begin. Whether the program starts internally or on the plugboard depends on how the START hub is patched:

START-NI patching: program begins internally; when the NI hub is pulsed, the wait is removed, and the next Advance OED signal will set OED to 0. (The first instruction executed is the Instruction Word stored at the location specified by the initial setting of PAK.) OED then cycles 0-13 as the first Instruction Word is executed;

START-STEP IN patching: program begins on the plugboard; when the STEP IN hub is pulsed, the wait is removed, and OED is set to 3. (The first instruction executed is the Program Step whose STEP IN hub is pulsed.) OED cycles 3-12 as the Program Step is executed.

If the START hub is not patched, the program does not begin.

2. Start after a Programmed or Manual Stop:

When a programmed or manual stop occurs the computer always sets up the next instruction in Program Control before actually stopping. When the START button is subsequently pushed, operation can be resumed from the point where the computer left off. (If the MASTER CLEAR button is pressed and then the START button is pressed, the START hub emits and operation is resumed in the same manner as described above for an Initial Start.)

3. Transfer from Internally-Stored Program to Plugboard-Defined Program (See Figure II-19, Page II-100)

There are three ways in which an internally-stored program is automatically interrupted and a plugboard-defined sequence or program begun:

Execution of a Transcop Instruction Word;

A combination of the emission of the Breakpoint 1, 2, or 3 hubs, and BREAKPOINT 1, 2, or 3 to STEP IN patching; and

A combination of the detection of an error condition and the following patching: ERROR hub to STEP CLEAR IN, and STEP CLEAR OUT to STEP IN.

Transcop and Breakpoint are usually used to initiate the plugboard portions of a normally-running program. Error hub patching is usually used to initiate an error analysis sub-routine should some error occur in the program.

→● STEP IN When a Program Step's STEP IN hub is pulsed, the Plugboard Step Number (51-98) of that Program Step is set up in PR, the Program Step's process is determined, an enable identifying that process is supplied to PCT, and OED is set to 3.

OED	PRINCIPAL EVENTS	EXPLANATION
3*	V_1 ADDRESS → SAR or U, V, or W of IRVc → SAR	Place address of V_1 in SAR
	V_1 SHIFT → SK or U, V, or W of SRV → SK	Place shift for V_1 in SK
4	Load V_1	Obtain V_1 and place it in RA
5*	V_2 ADDRESS → SAR or U, V, or W of IRVc → SAR	Place address of V_2 in SAR
	Shift V_1	Shift V_1 in RA
6*	Load V_2	Obtain V_2 and place it in RB
	V_2 SHIFT → SK or U, V, or W of SRV → SK	Place shift for V_2 in SK
7**	(Advance PAK, PAK → SAR)	Set up address for next IW in SAR
	Shift V_2	Shift V_2 in RB
8***	(Load IRVn)	Acquire next IW
	R SHIFT → SK or U, V, or W of SRV → SK	Place shift for R in SK
	Initiate Process	Carry out process (and check in Addition and Subtraction)
9*	R ADDRESS → SAR or U, V, or W of IRVc → SAR	Place address for R in SAR
	Shift R	Shift R in register from which R is stored
10	Store R	Store the result
11	Check	Check (Multiplication & Division)
12***	STEP OUT	STEP OUT hub emits

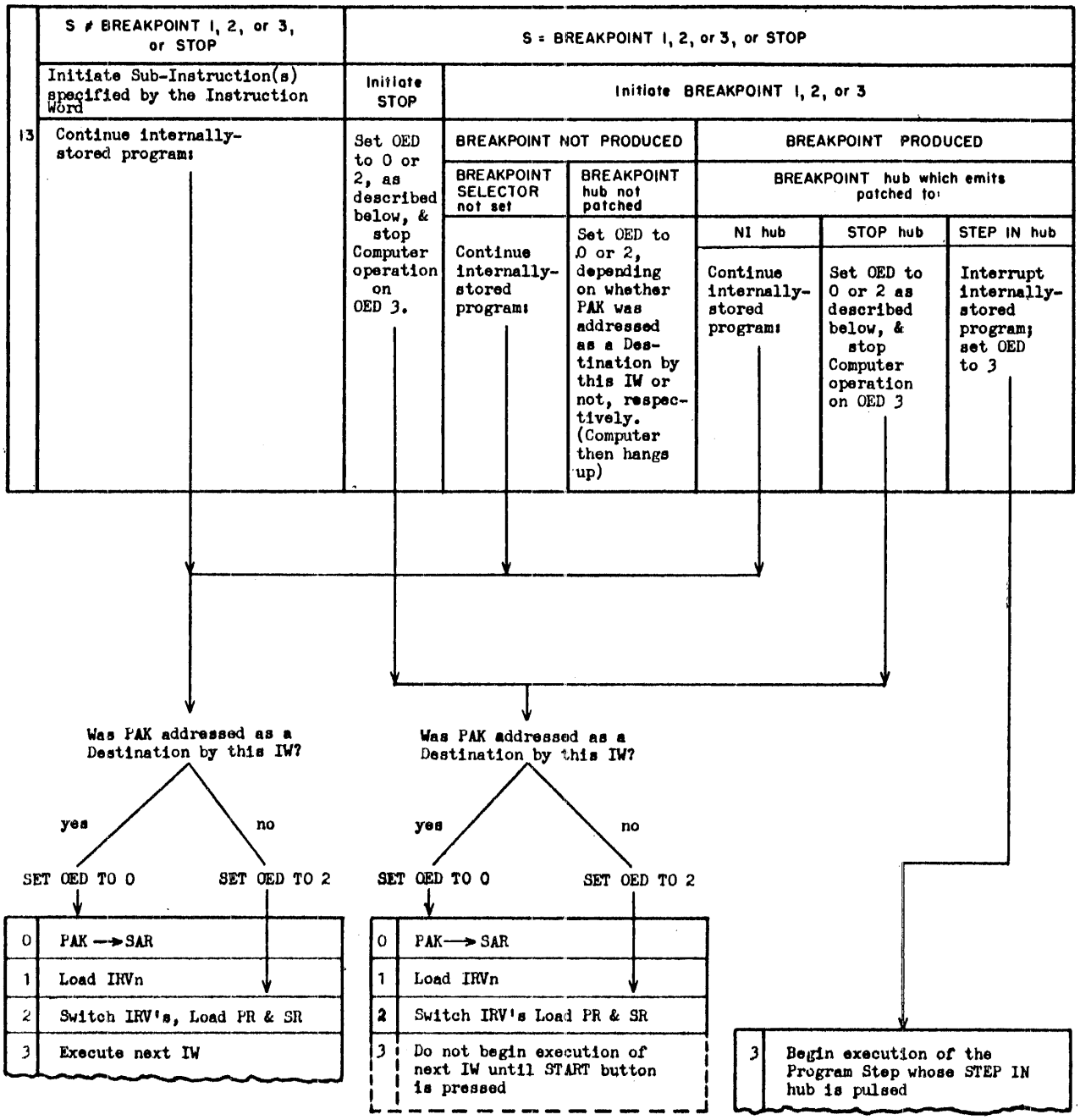
Table II-14. Typical OED Cycle for a Program Step

* If the ADDRESS and SHIFT hubs are patched to U, V, W ADDRESS and U, V, W SHIFT hubs respectively, the section of IRVc/SRV (U, W, or W) corresponding to the U, V, or W hub gets shifted into SAR/SK. If the ADDRESS and SHIFT hubs are patched to any other hubs in the Plugboard Addressing System or to any other shift-defining hubs, those hubs will <u>set</u> SAR/SK when enabled.

** The operations in parenthesis in these OED steps are performed only if this Program Step is the first in a Transcop sequence. See Paragraph IIC.

*** The Advance OED pulse which causes the STEP OUT hub to emit actually sets OED to 13, and generates a wait. If STEP OUT-STEP IN patching is employed, the wait is removed, and OED is set to 3 when the STEP IN hub receives a pulse. If STEP OUT-NI patching is employed, the wait is removed when the NI hub is pulsed, but OED is left at 13. The next Advance OED signal will set OED to 0 or 2, and initiate or resume the internally-stored program. If the STEP OUT hub is left unpatched, the computer hangs up with OED set to 13.

Figure II-19. Manner in Which Instruction Words Terminate and OED is set for Next Instruction



a. Via Transcop Instruction Words. (See also Paragraph IID, Transcop Instruction Word)

When Program Control executes a Transcop Instruction Word, PR is loaded on OED 2 with a TC code, i.e., with a Plugboard Step Number (51-98) and OED is set to 3. When Program Control finds a TC code in the Process Register at OED 3 time, it interrupts the internally-stored program and begins execution of the Program Step at the Plugboard Step numerically equal to TC. A sequence of Program Steps called a Transcop sequence is then executed.

During the first Program Step of every Transcop Sequence, PAK is advanced and the next Instruction Word is obtained as that first Program Step is executed. Subsequent Program Steps do not advance PAK or acquire any further Instruction Words. If the internally stored program is to be resumed after the Transcop sequence is completed, the STEP OUT of the last Program Step in the sequence is patched to one of the NI (NEXT INSTRUCTION) hubs. When that STEP OUT hub emits, OED is set to 13, and a wait is generated. It is not then reset to 3 since no STEP IN is pulsed. When the NI hub is pulsed, the wait is removed, and the Transcop Instruction Word's Sub-Instruction(s) is (or are) initiated. The setting given OED (0 or 2) by the next Advance OED signal depends on whether PAK was addressed as a Destination or not during the Transcop Sequence: (See Figure II-20 , Page II-104)

(1) If PAK was not addressed as a Destination by any Program Step in the Transcop sequence, OED is set to 2 by the next Advance OED signal and Program Control will immediately begin executing the next Instruction Word (This Instruction Word is in IRVn since it was obtained as the first Program Step of the Transcop sequence was executed.)

(2) If PAK was addressed as a Destination in any Program Step in the Transcop sequence, Program Control will detect this on OED 13 and know that it does not have the correct next Instruction Word in IRVn. Accordingly, OED will then be set to 0 by the next Advance OED signal, and Program Control will first acquire the correct next Instruction Word (i.e., acquire the Instruction Word whose location is specified by the contents of PAK at the conclusion of the Transcop sequence) and then execute that Instruction Word.

b. Via Breakpoint

If an Instruction Word is executed that contains a Breakpoint Special Character and the BREAKPOINT SELECTOR is set so as to allow that Breakpoint signal to be produced on the Program Control Plugboard, the next operation in the program depends on how the BREAKPOINT hub that emits is patched. (See Figure II-19 Page II-100).

(1) BREAKPOINT hub that emits is patched to a STEP IN hub:

The execution of the internally-stored program is interrupted on OED 13 of the Instruction Word containing the Breakpoint Special Character. OED is set to 3 when the STEP IN hub is pulsed, and Program Step sequencing begins.

No Program Step of a Breakpoint sequence advances PAK or obtains the next Instruction Word, as in the case of Transcop plugboard sequences

Unless the Instruction Word containing Breakpoint addressed PAK as a Destination, IRVn was loaded with the correct next Instruction Word during the execution of that instruction.

If the internally-stored program is to be subsequently resumed after a Breakpoint plugboard sequence, the STEP OUT hub of the last Program Step is wired to an NI hub. When that STEP OUT hub emits, OED is set to 13, and a wait is generated. It is not then reset to 3, since no STEP IN hub is pulsed. When the NI hub is pulsed, the wait is removed. If the correct next Instruction Word was loaded into IRVn during the execution of the Instruction Word initiating the Breakpoint Sequence, OED will be set to 2 by the next Advance OED signal, and Program Control will take that Instruction Word as the next instruction, unless any Program Step in the Breakpoint plugboard sequence addressed PAK as a Destination.

If PAK was sent data either by the Instruction word initiating Breakpoint or by any Program Step in the Breakpoint sequence, OED will be advanced from 13 to 0 by the first Advance OED signal to occur after the NI hub is pulsed. Program Control will thus first obtain the correct next Instruction Word from the address specified by the contents of PAK at the conclusion of the Breakpoint plugboard sequence before resuming the internally-stored program.

(2) BREAKPOINT hub that emits is patched to a STOP hub.

When Breakpoint is used in this manner, OED advances from 13 to 0 or to 2 depending on whether the Instruction Word initiating the Breakpoint addressed PAK as a Destination or not, respectively. In any event, the next Instruction Word is set up in Program Control and Central Computer operations stop with OED set to 3. Operation can be resumed from that point in the internally-stored program by pressing the START button.

(3) BREAKPOINT hub that emits is patched to an NI hub.

No interruption of the internally-stored program takes place. After the NI hub is pulsed, the next Advance OED signal sets OED to either 0 or 2 depending on whether PAK was addressed as a Destination or not, respectively, by the Instruction Word initiating Breakpoint. The execution of the next Instruction Word then proceeds as if no Breakpoint were initiated.

c. Via ERROR-STEP CLEAR Patching

As noted in Paragraph IIB1n above, a sequence of instructions designed to analyze the cause of an error can be initiated when certain errors occur, provided appropriate ERROR-STEP CLEAR patching is employed. When error analysis sequences are initiated, Program Control always starts these sequences from the plugboard as follows:

When an error condition is detected, the corresponding ERROR hub emits and the STEP CLEAR IN hub is pulsed. This clears Program Control of the current instruction. If the STEP CLEAR OUT is patched to a STEP IN, the Program Step whose STEP IN hub is pulsed, is initiated. The subsequent error-analysis sub-routine can involve merely a sequence of Program Steps or it can employ sub-routines in the internally-stored program. In the latter case, of

course, an NI hub must be pulsed to begin the internally-stored portion of the error-analysis routine.

4. Transfer from Plugboard-Defined Programs to the Internally-Stored Program

Program Control interrupts plugboard-defined programs and begins execution of internally-stored programs when one of the NI (NEXT INSTRUCTION) hubs on the Program Control Plugboard is pulsed. In every case where an NI hub is pulsed, OED is thereafter set to either 0 or 2 and the internally-stored program begun. Transfer to the internally-stored program from each of the following plugboard sequences is illustrated in Figure II- 20, Page II-104:

Initial Start sequence
Transcop sequence
Breakpoint sequence

The manner in which the internally-stored program resumes after Transcop and Breakpoint sequences was discussed above. The manner in which internally-stored programs begin when the program starts on the plugboard is discussed in the next paragraph.

When a program begins on the Program Control Plugboard (i.e., the START hub is patched to a STEP IN hub) the first instruction executed is a Program Step and the plugboard sequence initiated is called an Initial Start sequence. In such programs, a switch to the internally-stored portion of the program is accomplished in the usual way; i.e., by appropriately pulsing an NI hub. If PAK was not referred to as a Destination in any Program Step, the initial setting of PAK is used as the address of the first instruction Word; if PAK was addressed as a Destination by any Program Step, the new setting, rather than the original setting of PAK is used. In short, when the internally-stored program is begun after an Initial Start Plugboard sequence, the contents of PAK at the time the NI hub is pulsed specifies the address of the Instruction Word which is to be executed first in the program.

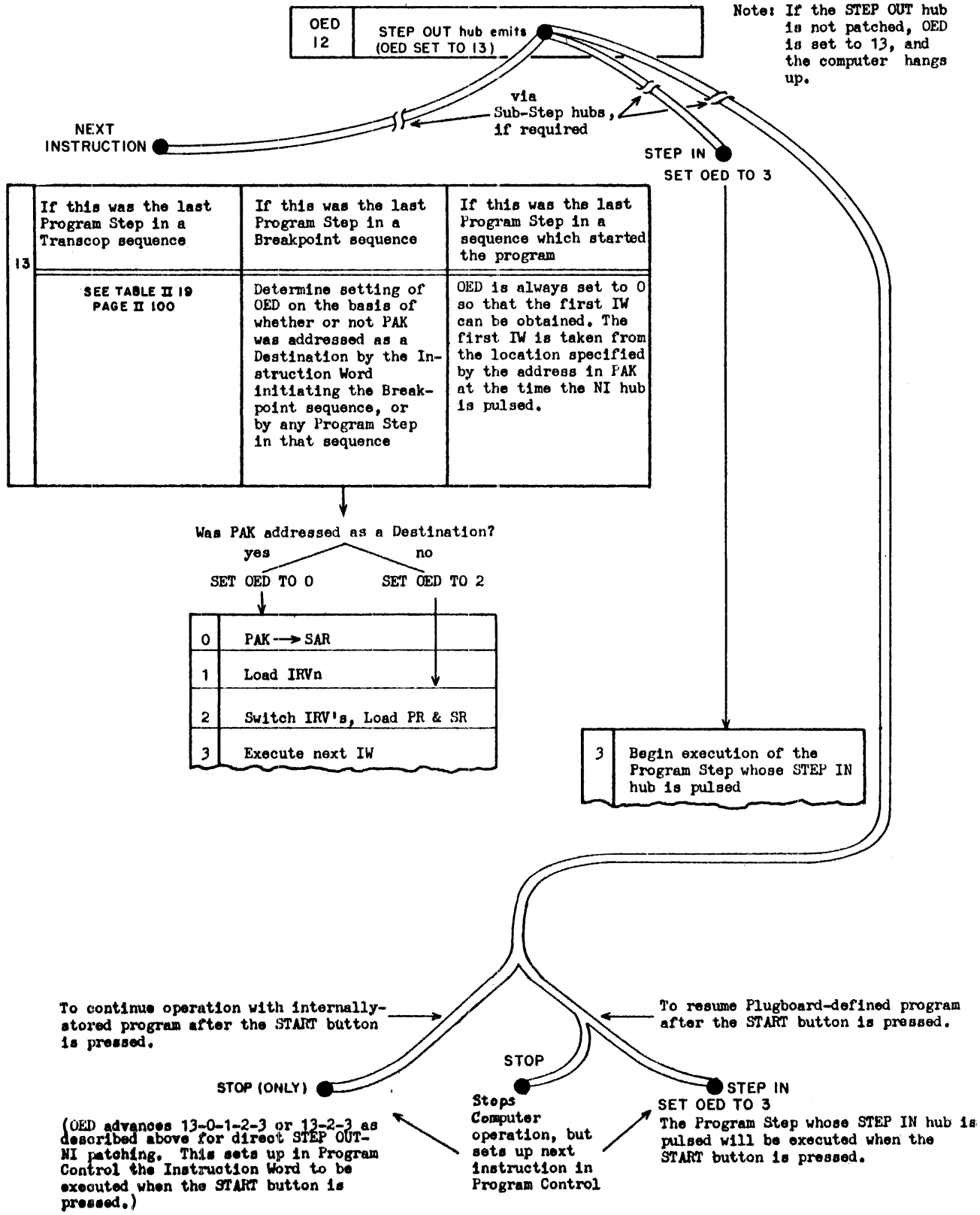
As illustrated in Figure II-20 Program Control can also be transferred to the internally-stored program by direct STEP OUT-STOP patching. In this case, of course, the internally-stored program cannot be resumed (or started) until the START button is pressed.

D. DETAILED ANALYSIS OF EACH COMPUTER INSTRUCTION

The remaining pages in this section describe the basic repetory of instructions in the UFC Model 1 Central Computer.

Instruction Words not duplicated as instructions on the plugboard are presented first. Computer Instructions that can be defined either by an Instruction Word or a Program Step are then discussed. The description of each instruction includes an explanation of the instruction, a listing of the basic rules that must be observed in using the instruction in a program, and an OED cycle that outlines the principal events that are carried out in executing the instruction. The time required for the execution of each instruction is listed in the OED cycle for that instruction.

Figure II-20. Manner in Which Program Steps Terminate and OED is set for Next Instruction



SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Examine Conditional Storage for MINUS:	PR = 15
If Conditional Storage is MINUS: transfer the	three characters that form the U-section of this Instruction Word
to the W-section of the Word Location* specified by the	V-section of this Instruction Word;
then, transfer the	W-section of this Instruction Word
to PAK, and use the (modified) contents of PAK as the address of the next instruction.	
If Conditional Storage is not MINUS: Ignore the U, V, and W-sections of this Instruction Word and take the next Instruction Word from the location specified by the (unaltered) contents of PAK.	
Initiate Sub-Instruction(s) in accordance with	S

*See Jump Instruction Rules.

JUMP ON PLUS
JP
(17)

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Examine Conditional Storage for PLUS:	PR = 17
If Conditional Storage is PLUS: transfer the	three characters that form the U-section of this Instruction Word
to the W-section of the Word Location* specified by the	V-section of this Instruction Word;
then, transfer the	W-section of this Instruction Word
to PAK, and use the (modified) contents of PAK as the address of the next instruction.	
If Conditional Storage is not PLUS: Ignore the U, V, and W sections of this Instruction Word and take the next Instruction Word from the location specified by the (unaltered) contents of PAK.	
Initiate Sub-Instruction(s) in accordance with	S

*See Jump Instruction Rules.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Examine Conditional Storage for ZERO: If Conditional Storage is ZERO: transfer the	PR = 19 three characters that form the U-section of this Instruction Word
to the W-section of the Word Location* specified by the	V-section of this Instruction Word;
then transfer the	W-section of this Instruction Word
to PAK, and use the (modified) contents of PAK as the address of the next Instruction Word.	
If Conditional Storage is not ZERO: Ignore the U, V, and W-sections of this Instruction Word, and take the next Instruction Word from the location specified by the (unaltered) contents of PAK	
Initiate Sub-Instruction(s) in accordance with	S

* See Jump Instruction Rules.

UNCONDITIONAL JUMP
UJ
(41)

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Modify the contents of PAK and cause a jump to occur as follows:	PR = 41
Transfer the	three characters that form the U-section of this Instruction Word
to the W-section of the Word Location* specified by the	V-section of this Instruction Word;
then, transfer the	three characters that form the W-section of this Instruction Word
to PAK, and take the next Instruction Word from the location specified by the modified contents of PAK.	
Initiate Sub-Instruction(s) in accordance with	S

*See Jump Instruction Rules.

RULES FOR JUMP INSTRUCTION WORDS

1. Permissible Destinations for U:

   a). V = any Word Location in GSB or BTB

TIMING for U→V (milliseconds)		
Minimum	Average	Maximum
.252	.483	.714

   b). V = any Word Location on I/O, FS, or IS Tracks

TIMING for U→V (milliseconds)		
Minimum	Average	Maximum
.252	2.751	5.250

   c). V = Register C : U→V = .042 milliseconds

   Notes: If any but the above locations are specified by V, the computer hangs up.

   V = RC is used when it is desirable to jump but it is not necessary to condition the program to return to any particular point.

2. The Conditional Jump Instruction Words (JN, JP, JZ) merely test the status of Conditional Storage at the time the IW is executed; i.e., their normal use requires that Conditional Storage be previously set by a Set Conditional Storage Sub-Instruction.

   A MASTER CLEAR sets Conditional Storage to ZERO. Each Set Conditional Storage Sub-Instruction will set Conditional Storage to the current setting of Branch Storage. If no Set Conditional Storage Sub-Instruction is executed prior to a Conditional Jump, therefore, the ZERO setting of Conditional Storage (from MASTER CLEAR) will be found.

3. If V is the address of the location from which the Jump Instruction is obtained, the Jump Instruction modifies itself in storage but not in IRVc as it is being executed.

OED CYCLE FOR CONDITIONAL JUMPS
(Instruction Words)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK ⟶ SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's	.210	.441	.672
	From OP of IRVc: 15, 17, or 19 → PR value of S → SR			
3	EXAMINE CONDITIONAL STORAGE: if PR = 15, test for Negative if PR = 17, test for Plus if PR = 19, test for Zero Condition Tested is Found: JUMP			
	U of IRVc → RC	.588	.819	1.050
4		.042		
5	Inhibit PAK Advance on OED 7	.210	.441	.672
	W of IRVc → PAK			
6		.042		
7	PAK → SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	V of IRVc → SAR	.210	.441	.672
10	Store (U) the lower-order three characters in RC at the location specified by V	See "Permissible Destinations", under Rules for Jump Instruction Words.		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s)	.042		
	Clear SRV			
	Set OED (See Page II-100)			

*Omitted if this IW is already in IRVn.

Condition Tested is not found: NO JUMP

3	Set OED to 7	.042		
7	Advance Pak PAK→SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9		.042		
10		.042		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II-100)	.042		

OED CYCLE FOR UNCONDITIONAL JUMP
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 41→PR value of S→SR	.210	.441	.672
3	U of IRVc→RC	.588	.819	1.050
4		.042		
5	Inhibit PAK Advance on OED 7 W of IRVc→PAK	.210	.441	.672
6		.042		
7	PAK→SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	V of IRVc→SAR	.210	.441	.672
10	Store (U) the lower-order three characters in RC at location specified by V	See "Permissible Destinations". under Rules for Jump Instruction Words.		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Test to see if a previously initiated Channel Search operation is completed or not:	PR = 27
If the previously initiated search is not completed, take the next Instruction Word from the location specified by the	U-section of this Instruction Word
If the previously initiated search operation is completed, examine Channel Search Storage:	
if Channel Search Storage is set to MINUS, take the next instruction from the location specified by the	V-section of this Instruction Word
if Channel Search Storage is set to ZERO, take the next instruction from the location specified by the	W-section of this Instruction Word
if Channel Search Storage is set to PLUS, take the next instruction from the location specified by the contents of PAK, and ignore the U, V, W-sections of this Instruction Word.	
Initiate Sub-Instruction(s) in accordance with	S

This IW merely tests the status of Channel Search Storage; i.e., it assumes that a Channel Search operation has been previously initiated. If no Channel Search operation has been previously initiated, Channel Search Storage will be in a "cleared" (no setting) condition and the computer will hang up. If no Channel Search operation has been initiated since the last Channel Search Probe, the setting of Channel Search Storage at the time of the last probe will be found again. Each time a Channel Search operation is initiated Channel Search Storage is "cleared" (i.e., its setting is removed). a MASTER CLEAR also "clears" Channel Search Storage.

OED CYCLE FOR CHANNEL SEARCH PROBE
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 27→PR value of S→SR	.210	.441	.672
3	PROBE CHANNEL SEARCH STORAGE: Channel Search Storage is not + : JUMP If ACTIVE, send U to PAK on OED 5 If — , send V to PAK on OED 5 If 0 , send W to PAK on OED 5	.042		
4		.042		
5	Inhibit PAK Advance on OED 7 U, V, or W of IRVc→PAK	.210	.441	.672
6		.042		
7	PAK→SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	Set OED to 12	.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II-100)	.042		

*Omitted if this IW is already in IRVn.

Channel Search Storage is + : NO JUMP

3	Set OED to 7	.042		
7	Advance PAK	.210	.441	.672
	PAK→SAR			
8	Load IRVn	See Page II-229		
9	Set OED to 12	.042		
12		.042		
13	Initiate Sub-Instruction(s)	.042		
	Clear SRV			
	Set OED (See Page II-100)			

LOAD SHIFT
LS
(32)

SEQUENCES OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Transfer this Instruction Word to the Shift Revolver, as follows:	PR = 32
place the	U-section of this Instruction Word
in the U-section of SRV;	
place the	V-section of this Instruction Word
in the V-section of SRV;	
place the	W-section of this Instruction Word
in the W-section of SRV; and	
place the	OP-section of this Instruction Word
in the lower - order character positions of SRV.	
Initiate Sub-Instruction(s) in accordance with	S

SHIFT REVOLVER RULES

1. Shift Words can be loaded into SRV either by executing the Load Shift (32S) Instruction Word or by addressing SRV as a Destination in an instruction (i. e. by making the W-Address of an Instruction Word 998, or by patching the R ADDRESS hub of a Program Step to the SRV hub).

2. If only Instruction Words are executed in a program, SRV is "cleared" i.e. the Shift Word it contains is unavailable after each Instruction Word is executed, unless that Instruction Word was a Load Shift instruction, or SRV was addressed as a Destination in that Instruction Word. If SRV is loaded by one Instruction Word, the Shift Word it holds is available only to the next Instruction Word.

3. If only Program Steps are executed in a program, SRV is not cleared after each instruction. Any Shift Word placed in SRV (by addressing SRV as a Destination) is available to succeeding Program Steps until that Shift Word is replaced by another.

4. In programs in which both Instruction Words and Program Steps are executed, two general rules can be stated:

   If the first instruction after an instruction in which SRV is loaded is an Instruction Word, the Shift Word in SRV is available only to that Instruction Word.

   If the first instruction after an instruction in which SRV is loaded is a Program Step, the Shift Word in SRV is available to that and all subsequent Program Steps until either an NI Sub-Step is performed or a new Shift Word is loaded into SRV.

   The following cases are examples of the application of these rules:

   (a) Program Control begins with Instruction Words and is transferred to the plugboard by a Transcop Instruction Word:

      (1) If SRV was loaded during the Instruction Word which immediately preceded the Transcop Instruction Word, the Shift Word held in SRV is available to any Program Step in the Transcop plugboard sequence. The Shift Word in SRV can be changed by any Program Step as in 3. above. When the NI hub on the plugboard is pulsed and Program Control reverts to the internally-defined program, the contents of SRV are not available unless the last Program Step in the sequence loaded SRV.

      (2) If SRV was not loaded during the Instruction Word which immediately preceded the Transcop Instruction Word, no Shift Word is available to the first Program Step in the Transcop plugboard sequence. Any Program Step in the sequence can load SRV, however,

and place a Shift Word in SRV for use by subsequent Program Steps. At the conclusion of the Transcop plugboard sequence, i. e. when the NI hub is pulsed, the contents of SRV are not available unless the last Program Step in the sequence loaded SRV.

(b) Program Control begins with Instruction Words and is transferred to the plugboard by Breakpoint:

(1) If SRV was loaded during the Instruction Word whose Special Character is Breakpoint, the Shift Word in SRV at the beginning of the Breakpoint plugboard sequence is available to any Program Step in that sequence. Any Program Step can change the Shift Word in SRV. When the NI hub is pulsed, however, the contents of SRV are not available unless the last Program Step in the sequence loaded SRV.

(2) If SRV was not loaded by the Instruction Word initiating the Breakpoint sequence, no Shift Word is available to the first Program Step in the Breakpoint sequence. Any Program Step in the sequence can load SRV, however, and place a Shift Word in SRV for use by subsequent Program Steps. At the conclusion of the Breakpoint sequence, i. e., when the NI hub is pulsed, the contents of SRV are not available unless the last Program Step in the sequence loaded SRV.

(c) Program Control begins with Program Steps and is transferred to the internally-stored program by an NI Sub-Step:

(1) No Shift Word is available until SRV is loaded by a Program Step. Once SRV is loaded, the Shift Word it contains is available to subsequent Program Steps as long as (a) no other Shift Word is loaded into SRV, and (b) Program Control continues to obtain instructions from the plugboard. When the NI hub is pulsed, the contents of SRV are not available to the next instruction (Instruction Word) unless the last Program Step in the Initial Start sequence loaded SRV.

OED CYCLE FOR LOAD SHIFT
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 32→PR value of S→SR	.210	.441	.672
3	Set OED to 7	.042		
7	Advance PAK PAK→SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	Force Set SAR to 998	.042		
10	IRVc→SRV	.588	.819	1.050
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn

LOAD GSAR
LA
(31)

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Load GSAR with 7 characters from this Instruction Word as follows: place the 2 lowest-order digits of the → U-section of this Instruction Word └ in GSAR's two-highest order digit positions; place the 3 digits of the → V-section of this Instruction Word └ in GSAR's next-three-higher-order digit positions; and place the 2 highest-order digits of the → W-section of this Instruction Word └ in GSAR's two-lowest-order digit positions.	PR = 31
Initiate Sub-Instruction(s) in accordance with	S

See Pages VI-4 through VI-12 for the rules regarding coding General Storage Addresses

OED CYCLE FOR LOAD GSAR
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 31→PR value of S→SR	.210	.441	.672
3	Set OED to 7	.042		
7	Advance PAK PAK→SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	Force Set SAR to 995	.042		
10	IRVc (7 characters)→GSAR	.378	.609	.840
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn

SUBSTITUTE U
SU
(24)

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Place, in Register A, the contents of the location specified by the	U-section of this Instruction Word
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver
Place, in Register B, the contents of the location specified by the	V-section of this Instruction Word
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver
Right end around shift the contents of Registers A and B; send a modified copy of the contents of Register A to Registers C and D as follows: Replace the contents of the U-section of Register A by the contents of the U-section of Register B, when forming the modified copy of the contents of Register A in Registers C & D.	PR = 24
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver
Store the final contents of Register D at the location specified by the	W-section of this Instruction Word
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Place, in Register A, the contents of the location specified by the	U-section of this Instruction Word
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver
Place, in Register B, the contents of the location specified by the	V-section of this Instruction Word
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver
Right end around shift the contents of Registers A and B; send a modified copy of the contents of Register A to Registers C and D as follows: Replace the contents of the V-section of Register A by the contents of the V-section of Register B, when forming the modified copy of the contents of Register A in Registers C & D.	PR = 25
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver
Store the final contents of Register D at the location specified by the	W-section of this Instruction Word
Initiate Sub-Instruction(s) or Sub-step(s) in accordance with	S

SUBSTITUTE W
SW
(26)

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Place, in Register A, the contents of the location specified by the	U-section of this Instruction Word
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver
Place, in Register B, the contents of the location specified by the	V-section of this Instruction Word
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver
Right end around shift the contents of Registers A and B; send a modified copy of the contents of Register A to Registers C and D as follows: Replace the contents of the W-section of Register A by the contents of the W-section of Register B, when forming the modified copy of the contents of Register A in Registers C & D.	PR = 26
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver
Store the final contents of Register D at the location specified by the	W-section of this Instruction Word
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S

RULES FOR SUBSTITUTE INSTRUCTION WORDS

1. Permissible Sources for V_1 and V_2:

   Any Word Location in GSB, BTB, I/O, IS, or FS Tracks
   Any Field Location in GSB, BTB, I/O, IS, or FS Tracks
   Notes: if a Field greater than 12 characters is referred to only the lower-order 12 characters of the Field are acquired;

   if a Field less than 12 characters is referred to, the Field is loaded into the lower-character positions of the arithmetic register(s) involved (RA, & RB) and the higher-order character positions of the arithmetic register are padded with Space codes.
   RA, RB, RC, RD, or IRVc

   GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as Sources, a plus (Δ) sign is automatically generated and sent as the lowest-order character of V_1 or V_2. The contents of GSAR, PAK or CDR are then loaded into the arithmetic register(s) involved (RA, RB) and the required number of Space codes are padded into the higher-order character positions.
   The entire contents of a buffer or track (Z-address) must not be specified.

2. Permissible Destinations for R:

   Any Word Location in GSB, BTB, I/O, FS, or IS Tracks
   Any Field Location in GSB, BTB, I/O, FS, or IS Tracks
   Notes: if a Field greater than 12 characters is specified, the contents of the Source of the result (RD) is loaded into lower order character positions of the Field Location and Space codes are padded into the higher-order character positions;

   if a Field less than 12 characters is specified, only the lower-order characters in the Source of the result are sent to the Field Location.

   The entire contents of a buffer or track (Z-address) must not be specified.
   RA, RB, RC, RD, IRVn, or SRV

   GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as a Destination, the sign in the arithmetic register that is the Source of the result and a number of characters equal to the capacity of GSAR, PAK or CDR are sent to these locations. As the highest-order character transmitted is shifted into these locations, the sign is shifted off the right end.

3. The only modification of data involved when the Substitute U, V, and W Instruction Words are executed is the replacement of the U, V, or W-section of V_1 in the result by the corresponding U, V, or W-section of V_2.

4. Shifts, any type, can be programmed for V_1, V_2, and R.

OED CYCLE FOR SUBSTITUTE U, V, and W
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's	.210	.441	.672
	From OP of IRVc: 24, 25, or 26→PR value of S →SR			
3	U of IRVc →SAR	.210	.441	.672
	U of SRV→SK			
4	Load V_1 in RA	See Page II-229		
**5	V of IRVc →SAR	.210	.441	.672
	Shift V_1 in RA	.042 (n+1)		
6	Load V_2 into RB	See Page II-229		
	V of SRV→SK			
**7	Advance PAK	.210	.441	.672
	PAK→SAR			
	Shift V_2 in RB	.042 (n+1)		
8	Load IRVn	See Page II-229		
	W of SRV→SK			
	Carry Out Substitution	.63		
**9	W of IRVc →SAR	.210	.441	.672
	Shift R in RD	.042 (n+1)		
10	Store R from RD	See Page II-230		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s)	.042		
	Conditional SRV Clear			
	Set OED (See Page II-100)			

* Omitted if this IW is already in IRVn

**Times listed are simultaneous. Larger of the two determines the time required for this OED Step. ("n" is the number of places V_1, V_2, or R is shifted.)

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Determine whether I/O Unit "b" is READY or NOT READY for subsequent use, as follows:	PR = 34
Examine the	middle digit (b) of the U-section of this Instruction Word
and send the I/O Unit specified by this digit a DEMAND TEST IN signal.	
If I/O Unit "b" is READY, take the next Instruction Word from the location specified by the	V-section (xxx) of this Instruction Word.
If I/O Unit "b" is NOT READY, take the next Instruction Word from the location specified by the	W-section (yyy) of this Instruction Word.
Initiate Sub-Instruction(s) in accordance with	S

RULES FOR TEST DEMAND IN INSTRUCTION WORD

1. The Demand Test In Instruction Word does not affect the demand status of the I/O Unit tested or that of any other I/O Unit.

2. The Demand Test In Instruction Word is a jump instruction.

3. The middle character of U, "b", can be any of the 44 characters in Univac code that have the same excess-three bits as the characters 0-9, and ': i.e., to specify I/O Unit 2, "b" can be 2, B, K, or S, since each of these characters have the excess three bits as the digit 2.

4. This Instruction Word is generally employed prior to the Demand In Instruction Word unless the latter must be executed at a particular point for the program to continue.

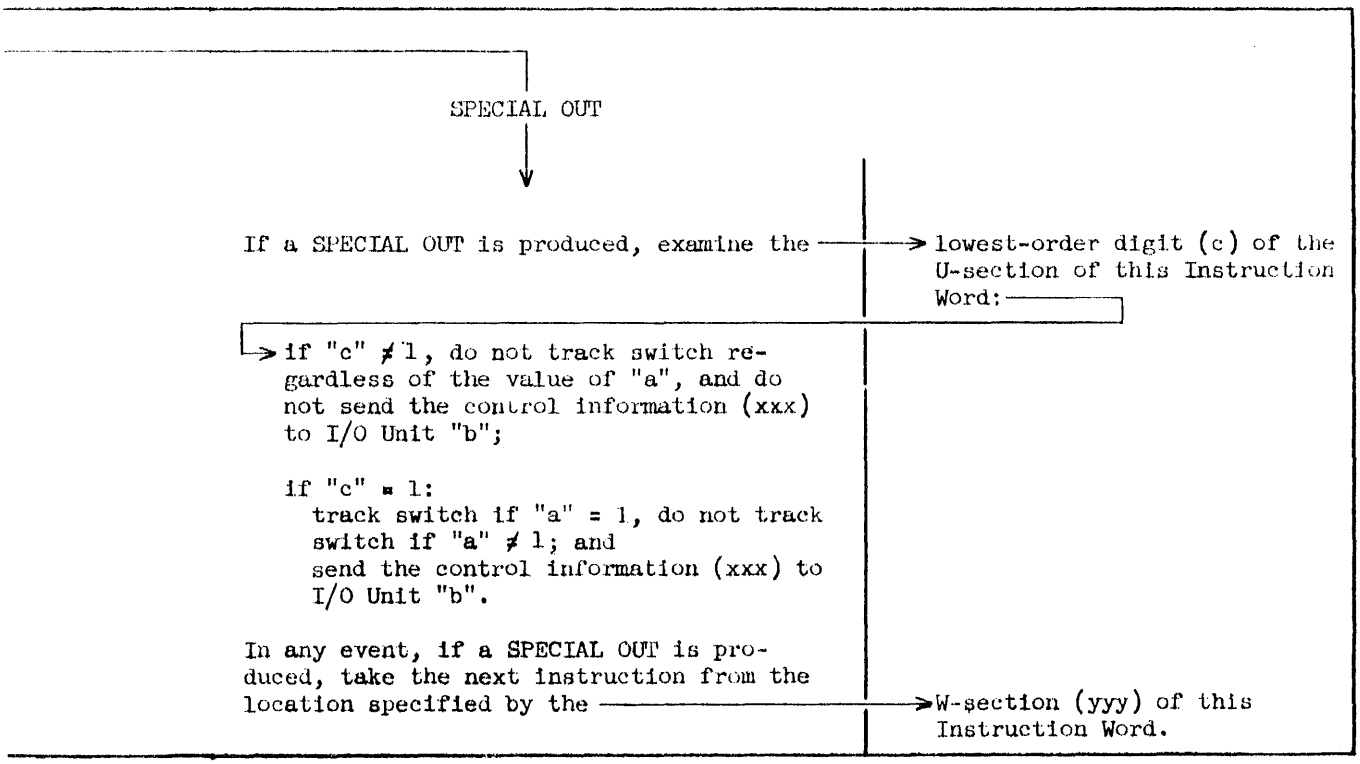
OED CYCLE FOR TEST DEMAND IN

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 34→PR value of S→SR	.210	.441	.672
3	U of IRVc→SAR	.210	.441	.672
4	Probe SAR to determine "b" and send I/O Unit "b" a DEMAND TEST IN signal If I/O Unit "b" is READY, send V to PAK on OED 5. If I/O Unit "b" is NOT READY, send W to PAK on OED 5.	.084		
5	Inhibit PAK Advance on OED 7 V, or W of IRVc→PAK	.210	.441	.672
6		.042		
7	PAK→SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	Set OED to 12	.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn.

DEMAND IN
DE
(45)

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Place I/O Unit "b" on demand, conditionally track switch, and exchange control information, as follows:	PR = 45
Examine the →	middle digit (b) of the U-section of this Instruction Word →
→ and send I/O Unit "b" a DEMAND IN signal; i.e., place I/O Unit "b" on demand:	
The following control lines of I/O Unit "b" are connected to the computer:	
C-I/O lines (A-J) (HS) I/O-C lines (W-Z) (LS) I/O-C lines (a-l). These lines (a-l) begin to energize immediately if I/O Unit "b" has produced the condition that causes them to energize.	
When I/O Unit "b" becomes READY, it produces a DEMAND OUT or a SPECIAL OUT signal. If it produces a SPECIAL OUT it also sends one or more signals over the (HS) I/O-C control lines (W,X,Y,Z) to High Speed I/O-Computer Control Line Storage	
DEMAND OUT	
If a DEMAND OUT "b" is produced, track switch if the →	left-most digit (a) of the U-section of this Instruction Word →
is = 1 ; do not track switch if it is ≠ 1 . Whether track switching occurs or not, send I/O Unit "b" the control information specified by the →	V-section (xxx) of this Instruction Word, →
(via the C-I/O) control lines A-J), and take the next Instruction Word from the location specified by the contents of PAK.	
Initiate Sub-Instruction(s) in accordance with →	S


SPECIAL OUT
If a SPECIAL OUT is produced, examine the
lowest-order digit (c) of the U-section of this Instruction Word:
if "c" ≠ 1, do not track switch regardless of the value of "a", and do not send the control information (xxx) to I/O Unit "b";
if "c" = 1:
track switch if "a" = 1, do not track switch if "a" ≠ 1; and
send the control information (xxx) to I/O Unit "b".
In any event, if a SPECIAL OUT is produced, take the next instruction from the location specified by the
W-section (yyy) of this Instruction Word.

RULES FOR DEMAND IN INSTRUCTION WORD

1. Permissible values of U = abc

<table>
<tr><th colspan="3">U</th></tr>
<tr><th>a</th><th colspan="2">b</th><th>c</th></tr>
<tr><td rowspan="2">"a" = 1 is detected in SAR if the highest-order character in U is

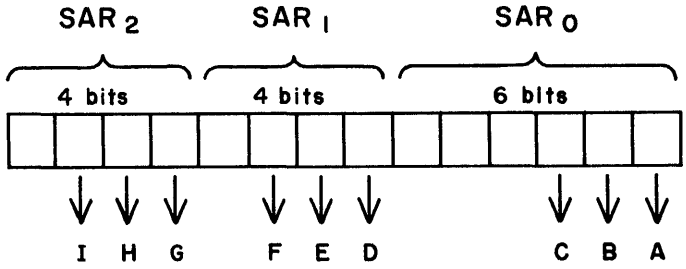
1
A
J
/
i
r
t, or
Σ</td><td>I/O Unit placed "on demand"</td><td>When Middle character of U is</td><td rowspan="2">"c" = 1 is detected in SAR if the lowest-order character of U is 1 or i.</td></tr>
<tr><td>0</td><td>0(digit), ;,), or +</td></tr>
<tr><td rowspan="10">"a" ≠ 1 is detected in SAR if the highest-order character in U is any other computer character</td><td>1</td><td>1, A, J, or /</td><td rowspan="10">"c" ≠ 1 is detected in SAR if the lowest-order character of U is any other computer character.</td></tr>
<tr><td>2</td><td>2, B, K, or S</td></tr>
<tr><td>3</td><td>3, C, L, or T</td></tr>
<tr><td>4</td><td>4, D, M, or U</td></tr>
<tr><td>5</td><td>5, E, N, or V</td></tr>
<tr><td>6</td><td>6, F, O(letter) or W</td></tr>
<tr><td>7</td><td>7, G, P, or X</td></tr>
<tr><td>8</td><td>8, H, Q, or Y</td></tr>
<tr><td>9</td><td>9, I, R, or Z</td></tr>
<tr><td>unit currently "on demand"</td><td>', #, $, %</td></tr>
</table>

Refer to III-4, Page III-16 for the manner in which SAR is interpreted

2. Permissible values of V = xxx (I/O Instructions)

Of the ten Computer-I/O, control lines (A-J) nine (A-I) can be pulsed via the Demand In Instruction Word to define an I/O Instruction for the I/O Unit "on demand". The particular control lines pulsed (hence the I/O Instruction sent) depends on what is stored in SAR at OED 10 time; i.e., on the coding of V (xxx) in the Demand In Instruction Word.

To determine how V can be coded to have a particular (one or set of) Computer-I/O control line(s) energized, the following must be noted regarding the examination of SAR in the Demand In Instruction Word at OED 10 time:



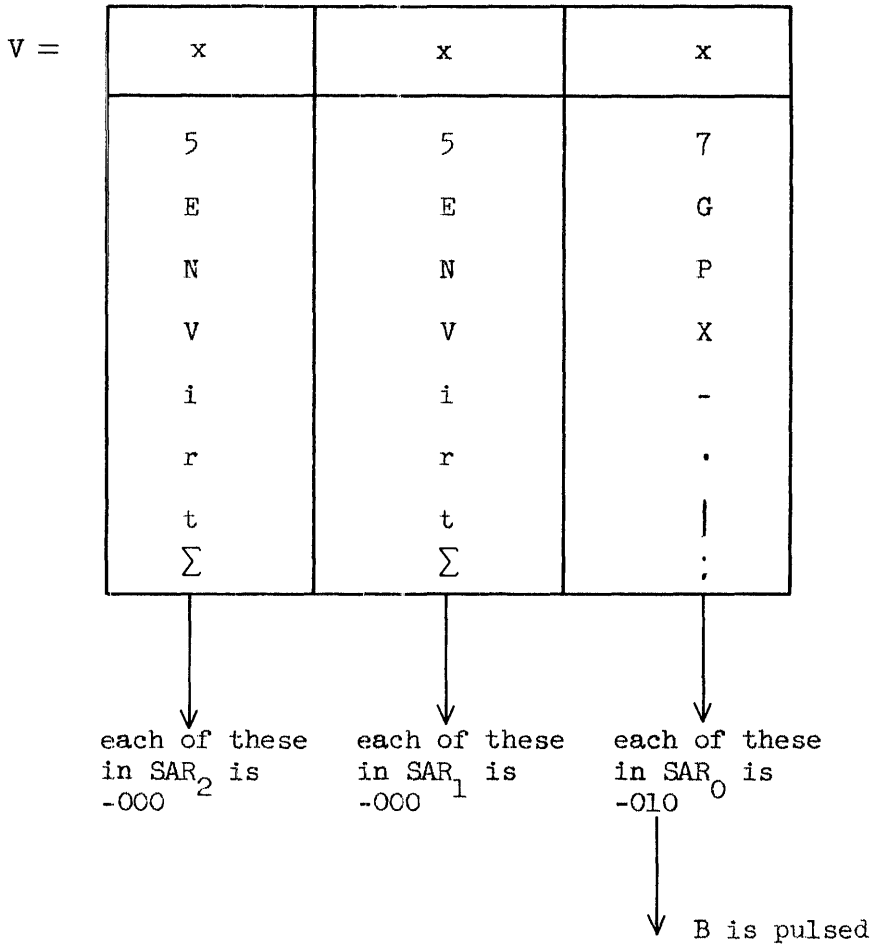
Only the lower-order three bits in each stage of SAR are examined to determine which C-I/O control lines are to be pulsed. For each of these bit positions that contain a "1" a Computer-I/O control line is pulsed. The control line associated with each bit position (and the control line pulsed when each bit position contains a "1") is indicated above. If for example the lowest-order bit position in SAR contains a (binary) "1", Computer-I/O control line A is pulsed; if the two lowest-order bit positions of SAR contain a "1", Computer-I/O control lines A and B are pulsed. If the lowest-order bit position of the middle stage of SAR contains a (binary) "1", Computer-I/O control line D is pulsed, etc. All that is required for the proper coding of V to cause one or more Computer-I/O control lines to be pulsed is that characters be used in V which contain binary "1's" in the same bit positions as those bit positions in SAR that cause the appropriate Computer-I/O control lines to be pulsed. A wide range of characters can be employed to do this. For example, to code V so that only Computer-I/O control line A is energized (when appropriate) in a Demand In Instruction Word, any of the following combinations could be used.

V =

x	x	x
5 E N V i r t Σ	5 E N V i r t Σ	6 F O W Δ , " β
↓	↓	↓
each of these in SAR_2 is -000	each of these in SAR_1 is -000	each of these in SAR_0 is -001

↓ A is pulsed

To code V so that only Computer-I/O control line B is energized (when appropriate) in a Demand In Instruction Word, any of the following combinations could be used



Although many possibilities exist for the manner in which V can be coded, in most programs only the digits 0-9 need be used. The table below shows the 3-digit values of V that can be normally used to cause various combinations of Computer-I/O control lines A-D to be pulsed. This table can be expanded to include the other control lines as required.

RULES FOR DEMAND IN INSTRUCTION WORD (Continued)

If V =	These Computer-I/O control lines are pulsed
555	NONE
556	A
557	B
558	AB
559	C
552	AC
554	ABC
553	BC
565	D
566	AD
568	ABD
564	ABCD
562	ACD
567	BD
563	BCD
569	CD
etc.	

Each I/O Unit's Computer-I/O control lines (A-J) are assigned a meaning (or are determined to have no meaning at all) as part of the system program. In some I/O Units, as the UFC Magnetic Tape Units, the combinations of signals sent over the Computer-I/O control lines (A-J) each cause an operation to be performed; and the combinations for each operation are fixed; i.e., each combination (as ABC) when it occurs always produces the same effect. In other I/O Units, as the punched card equipment, the Computer-I/O control lines (A-J) are assigned a particular meaning by patchcord wiring on the punched card equipment's plugboard. I/O Instructions can thus be made to vary from program to program, if required. Further, I/O Instructions do not all specify an I/O operation to be performed. In certain I/O Units, they may merely condition the I/O Unit for subsequent operations. To properly use Demand In Instruction

Words for the control of each I/O Unit, the definition of each Computer-I/O control line in each I/O Unit must be known or provided for and V must be coded in the appropriate Demand In Instruction Words to send the correct I/O Instructions as required.

3. W-section Coding:

The W-section is an address that usually specifies the location where the first Test Incoming Control Instruction Word for this I/O Unit has been stored; this permits the program to test High Speed I/O-Computer Control Line Storage immediately upon receipt of a SPECIAL OUT.

It is important to note the following about the Demand In Instruction Word:

it clears High Speed Control Line Storage at the time the DEMAND IN signal is sent to the I/O Unit;

it takes all I/O Units "off demand" and places I/O Unit "b" "on demand".

I/O unit "b" remains "on demand" until another I/O unit is placed "on demand" (all I/O Units go "off demand" when this is done and only the one specified is then placed "on demand").

Note: The (LS) I/O-Computer control lines (a-l) that are programmed in each I/O Unit to be energized when that I/O Unit goes "on demand" will cause their correspondingly-named (LS) I/O-COMPUTER CONTROL LINE hubs (a-l) to emit B+ as long as the I/O Unit is "on demand". These lines are used to pull Selectors on the Program Control Plugboard. If programmed for use in connection with the internally-stored program, a Transcop or Breakpoint sequence must also be programmed so that the (program variance) effect of these lines can be determined.

OED CYCLE FOR DEMAND IN
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 45→PR value of S→SR	.210	.441	.672
3	U of IRVc→SAR	.210	.441	.672
4	Probe SAR to determine "b" and send DEMAND IN Signal to I/O Unit "b" Clear HS I/O-C Control Line Storage Take all I/O Units off-demand Place I/O Unit "b" on demand **- When I/O Unit "b" is READY: DEMAND OUT Probe SAR to determine "a" if "a" = 1, track switch if "a" ≠ 1, do not track switch Set OED to 7	.084		
7	Advance PAK, PAK→SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	V of IRVc→SAR	.210	.441	.672
10	Probe SAR and send C-I/O signal(s)	.042		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn.
** Delay if I/O Unit demanded is not ready.

SPECIAL OUT

Probe SAR to determine "c"

"c" = 1

4	Probe SAR to determine "a" if "a" = 1, track switch if "a" ≠ 1, do not track switch	.084		
5	Inhibit PAK Advance on OED 7 W of IRVc → PAK	.210	.441	.672
6		.042		
7	PAK → SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	V of IRVc → SAR	.210	.441	.672
10	Probe SAR and send C-I/O signal(s)	.042		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II-100)	.042		

"c" ≠ 1

4	Do not track switch (ignore "a")	.084		
5	Inhibit PAK Advance on OED 7 W of IRVc → PAK	.210	.441	.672
6		.042		
7	PAK → SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9		.042		
10		.042		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II-100)	.042		

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Test High Speed I/O - Computer Control Line Storage for W, X,Y, or Z, as follows: Examine the ———	PR = 39 lowest order character (c) of the U-section of this Instruction Word:
If "c" = W, test High Speed I/O - Computer Control Line Storage for the W - condition; If "c" = X, test for the X - condition; If "c" = Y, test for the Y - condition; and If "c" = Z, test for the Z - condition	
If the particular W,X,Y, or Z - condition tested is found, examine the ———	middle digit (b) and highest order digit (a) of the U-section of this Instruction Word
for track switching: "b" specifies the I/O Unit whose associated I/O Tracks are to be conditionally switched: if "a" = 1, track switch if "a" ≠ 1, do not track switch In any event, take the next Instruction Word from the location specified by the ———	W- section (yyy) of this Instruction Word.
If the particular, W,X,Y, or Z condition tested is not found, do not track switch regardless of the value of "a"; and take the next Instruction Word from the location specified by the contents of PAK.	
Initiate the Sub-Instruction(s) in accordance with ———	S

RULES FOR TEST INCOMING CONTROL INSTRUCTION WORD

1. Permissible values of U = abc

U				
a	b		c	
"a" = 1 is detected in SAR if the highest-order character in U is: 1 A J / i r t, or Σ	I/O Tracks Ob- that are conditionally switched*	When middle character of U is:	To Test the following conditions	The lowest-order character of U must be:
	00-	0(digit), ;,), or +	W	W
	01-	1, A, J, or /	X	X
	02-	2, B, K, or S		
	03-	3, C, L, or T	Y	Y
	04-	4, D, M, or U	Z	Z
	05-	5, E, N, or V		
"a" ≠ 1 is detected in SAR if the highest-order character in U is any other computer character	06-	6, F, 0(letter) or W		
	07-	7, G, P, or X		
	08-	8, H, Q, or Y		
	09-	9, I, R, or Z		
	unit currently "on demand"	', #, $, %		

* See Rule 2 below.

2. The I/O Tracks that are conditionally switched are usually those associated with the I/O Unit that set High Speed I/O-Computer Control Line Storage and generated the SPECIAL OUT which jumped the program to the sequence of Test Incoming Control Instruction Words necessary to test for W, X, Y, and Z. However, any I/O Unit's associated I/O Tracks can be switched by appropriately coding "b".

3. Since track switching can be involved in both the Demand In and Test Incoming Control Instruction Words and since the latter are generally programmed to immediately follow a Demand In when a SPECIAL OUT is produced, care must be taken in the coding of "a" in both types of Instruction Words (and the coding of "c" in the Demand In) so that two (nullifying) track switching operations are not inadvertently specified.

RULES FOR TEST INCOMING CONTROL INSTRUCTION WORD (Continued)

4. The Test Incoming Control Instruction Word is completely executed in the central computer, and does not affect or test the demand status of any I/O Unit.

5. The SPECIAL OUT which makes it necessary to use the Test Incoming Control Instruction Word gives no clue as to which condition W, X, Y, or Z was set up in High Speed I/O-Computer Control Line Storage. W, X, Y, or Z, or any combination thereof could have been set up, depending on what has been programmed for, and what has happened in, the I/O Unit producing the SPECIAL OUT.

6. Only one condition: W, X, Y, or Z can be tested per Test Incoming Control Instruction Word. The number of Test Incoming Control Instruction Words that must be programmed for each I/O Unit depends on the number or conditions (W, X, Y, Z) and/or combinations of conditions (as WX, WXZ, etc.) that could be set up by the I/O Unit involved. The number of sets of Test Incoming Control Instruction Words that must be programmed depends on what differentiation must be made for the W, X, Y, and Z conditions from each I/O Unit.

OED CYCLE FOR TEST INCOMING CONTROL
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK → SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc; 39 → PR value of S → SR	.210	.441	.672
3	U of IRVc → SAR	.210	.441	.672
4	Probe SAR to determine whether "c" = W, X, Y, or Z; then test High Speed C-I/O Control Line Storage for W, X, Y, or Z as specified by "c": Condition tested, found: JUMP Probe SAR to determine "a" and "b" if "a" = 1, track switch I/O Tracks 0"b" if "a" ≠ 1, do not track switch	.042		
5	Inhibit PAK Advance on OED 7 W of IRVc → PAK	.210	.441	.672
6		.042		
7	PAK → SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	Set OED to 12	.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn

Condition tested, not found: NO JUMP

4	Set OED to 7	.042		
7	Advance PAK	.210	.441	.672
	PAK→SAR			
8	Load IRVn	See Page II-229		
9	Set OED to 12	.042		
12		.042		
13	Initiate Sub-Instruction(s)	.042		
	Clear SRV			
	Set OED (See Page II-100)			

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES IN THE INSTRUCTION WORD
Interrupt the execution of Instruction Words as follows: retain this Instruction Word in IRVc; transfer Program Control to the Plugboard Step numerically equal to	TC = (51-98); and
execute the Program Step patched there.	

A sequence of one or more Program Steps is thus initiated. The U, V, and W - sections of this Instruction Word are available to each Program Step in the Transcop sequence.

If the plugboard - defined program pulses the NEXT INSTRUCTION hub, the Program Control resumes the internally-stored program, and then initiates the Sub-Instruction(s) specified by S in this Instruction Word.

RULES FOR TRANSCOP INSTRUCTION WORD

1. TC = 50 or 99 the computer will hang up.

2. The U, V, and W-sections of a Transcop Instruction Word are available to Program Steps in Transcop sequences by patching the V_1 ADDRESS, V_2 ADDRESS and R ADDRESS hubs to a U, V, or W ADDRESS hub. When so used, provision must be made in coding the U, V, and W-sections of Transcop Instruction Words so that they are permissible addresses for the processes specified by the Program Steps which use them.

OED CYCLE FOR TRANSCOP
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 51-98→PR value of S→SR	.210	.441	.672

Program Control now switches to the plugboard to continue the program. The effecto is the same as if the STEP IN hub at the Plugboard Step (51-98) numerically equal to the contents of the Process Register were pulsed; OED is set to 3; the Program Step patched at that location is executed; and a Transcop sequence of instructions is initiated.

If the STEP OUT hub of Program Step in the Transcop sequence is patched to an NI hub: the Transcop sequence terminates; OED is set to 13; and the internal program resumes completing the execution of the Transcop Instruction Word:

13	Initiate Sub-Instrucion(s) specified by TC Instruction Word Conditional SRV Clear Set OED (See Page II-100)	.042

* Omitted if this IW is already in IRVn

ADD
AD
(14)
S=B,C,D,E,F,G,H, or I

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Add the contents of Registers A and B algebraically forming the sum (separately) in both Register C and D.	PR = 14	PROCESS to * NC patching
Determine that checking is to be suppressed in this instruction.	S must have one of the following values: B,C,D,E,F,G,H, or I	
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Store the final contents of Register D at the location specified by	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S* or	STEP OUT patching

*If S = E, the only Sub-Instruction (Suppress Check) specified by this instruction has already been completed at this time. If S has any of the other values: B,C,D,F,G,H, or I the other Sub-Instruction(s) these values indicate are initiated at this time.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Add the contents of Registers A and B algebraically, forming the sum (separately) in both Registers C and D	PR = 14	PROC to + C patching
Determine that checking is not to be suppressed in this instruction; and check the addition. (Use the sum in Register C for the check operation.)	S must not have one of the following values; B, C,D,E,F,G,H, or I	
Shift the contents of Register D in accordance with	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Store the final contents of Register D at the location specified by	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

RULES FOR ADDITION

1. Permissible Sources for V_1 and V_2:

   Any Word Location in GSB, BTB, I/O, IS, or FS Tracks
   Any Field Location in GSB, BTB, I/O, IS, or FS Tracks
   Notes: if a Field greater than 12 characters is referred to only the lower-order 12 characters of the Field are acquired;

   if a Field less than 12 characters is referred to, the Field is loaded into the lower-character positions of the arithmetic register(s) involved (RA, & RB) and the higher-order character positions of the arithmetic register are padded with Space codes.

   RA, RB, RC, RD, or IRVc

   GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as Sources, a plus (Δ) sign is automatically generated and sent as the lowest-order character of V_1 or V_2. The contents of GSAR, PAK or CDR are then loaded into the arithmetic register(s) involved (RA, RB) and the required number of Space codes are padded into the higher-order character positions.

   The entire contents of a buffer or track (Z-address) must not be specified.

2. Permissible Destinations for R:

   Any Word Location in GSB, BTB, I/O, FS, or IS Tracks
   Any Field Location in GSB, BTB, I/O, FS, or IS Tracks
   Notes: if a Field greater than 12 characters is specified, the contents of the Source of the result (RD is loaded into lower-order character positions of the Field Location and Space codes are padded into the higher-order character positions;

   if a Field less than 12 characters is specified, only the lower-order characters in the Source of the result are sent to the Field Location.

   RA, RB, RC, RD, IRVn, or SRV

   GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as a Destination, the sign in the arithmetic register that is the Source of the result and a number of characters equal to the capacity of GSAR, PAK or CDR are sent to these locations. As the highest-order character transmitted is shifted into these locations, the sign is shifted off the right end.

   The entire contents of a buffer or track (Z-address) must not be specified.

3. All three quantities (V_1, V_2, and R) manipulated by Add or Add & Check can be shifted right, left, or right end around. Appropriate shifts must be programmed so that

a. Correspondingly-significant characters in V_1 and V_2 are (algebraically) added; and

b. The (algebraic) sum of V_1 and V_2 does not exceed 11 characters and sign.

Note: If the (algebraic) sum of V_1 and V_2 is greater than 11 characters and sign in an Add or Add & Check, a +/- OVERFLOW error occurs:

the Add or Add & Check instruction terminates; and

the +/- OVERFLOW hub on the Program Control Plugboard emits.

4. Alpha and Numeric Characters in V_1 and V_2,

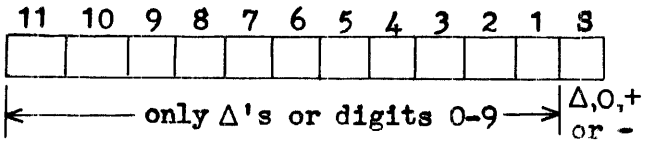
In Add and Add & Check processes, the arithmetic section recognizes the 6 bits in Univac code of each character in RA and RB, and forms 6-bit characters in the result (in RC and RD). It accordingly can differentiate alpha and numeric characters.

	V_1		V_2	
	Numeric	Alpha	Numeric	Alpha
Sign Position	Δ, 0, + these are all plus - only this is negative	All other characters are alpha, except i. An Ignore code is not regarded as alpha or numeric, but as a special operator for which separate rules apply	No alpha/numeric differentiation is made in the V_2 sign position. However the following sign interpretation is made: If V_2 sign is -, V_2 is negative. If V_2 sign is any other character but an Ignore code, V_2 is positive If V_2 sign is an Ignore code, V_2 is neither positive nor negative and special rules apply	
Character Positions 1-11	Δ and the digits 0-9 are numeric	all other characters are alpha except i. Special rules apply to i.	Δ and the digits 0-9 are numeric	All other characters are alpha except i. Special rules apply to i.

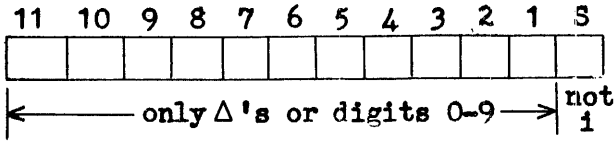
5. Numeric Add Rules:

The computer applies the rules listed below when V_1 and V_2 have the following composition:

Register A (V_1)



Register B (V_2)



Result	V_1 and V_2 have the same Sign			V_1 and V_2 have opposite Signs								
				$\lvert V_1 \rvert > \lvert V_2 \rvert$			$\lvert V_2 \rvert > \lvert V_1 \rvert$			$\lvert V_1 \rvert = \lvert V_2 \rvert$		
SIGN OF THE RESULT	V_1 Sign=	0,Δ,+	-	V_1 Sign=	0, Δ,+	-	V_1 Sign=	0, Δ,+	-	V_1 Sign =	0, Δ,+	-
	V_2 Sign=	any char. but i or -	-	V_2 Sign=	-	any char. but i or -	V_2 Sign=	-	any char. but i or -	V_2 Sign =	-	any char. but i or -
	R Sign=	Δ	-	R Sign=	Δ	-	R Sign=	-	Δ	R Sign=	Δ	-
MAGNITUDE OF THE RESULT	$\lvert V_1 \rvert + \lvert V_2 \rvert$			$\lvert V_1 \rvert - \lvert V_2 \rvert$			$\lvert V_2 \rvert - \lvert V_1 \rvert$			ZERO		

As corresponding characters in V_1 and V_2 are added (subtracted), any carry (borrow) from the character position to the right is added to (subtracted from) the sum (difference) of the V_1 and V_2 characters, and any carry (borrow) resulting from the sum (difference) is applied to the next highest-order character position.

Numeric Add and Add & Check are thus algebraic additions. Note that the Sign of the Result is either Δ (meaning plus) or - (meaning negative). In this connection it is important to also note the following special-handling of Δ's that occur in Character Positions 1-11 of V_1 and V_2:

Character Positions 1-11

If the V_1 Character =	Δ	0	Δ	Δ	1-9
and the V_2 Character =	Δ	Δ	0	1-9	Δ
then the Result Character =	0	0	0	1-9	1-9

6. "Alpha Add" Rules:

(a)

	WHEN V_1 CHARACTER IN CHARACTER POSITIONS 11-1 IS		
	NUMERIC	ALPHA	AN IGNORE CODE
AND THE CORRESPONDING V_2 CHARACTER IS NUMERIC	The result character formed in RC and RD is numeric; it is the sum (difference) of the absolute values of the V_1 and V_2 characters and the carry (or minus the borrow) from the next-lowest-order character position . Any carry (or borrow) resulting from the sum (or difference) of these two characters is applied to the next-highest-order character position	The V_1 character is sent to RC and RD as the result character. Any carry (or borrow) from the next-lowest-order character position is ignored; no carry (or borrow) is applied to the next-highest-order character position.	The V_2 character is sent to RC and RD as the result character. Any carry (or borrow) from the next-lowest-order character position is ignored; no carry (or borrow) is applied to the next-highest-order character position.
AND THE CORRESPONDING V_2 CHARACTER IS ALPHA	The V_1 character is sent to RC and RD as the result character. Any carry (or borrow) from the next-lowest-order character position is ignored; no carry (or borrow) is applied to the next-highest-order character position.	The V_1 character is sent to RC and RD as the result character. Any carry (or borrow) from the next-lowest-order character position is ignored; no carry (or borrow) is applied to the next-highest-order character position.	The V_2 character is sent to RC and RD as the result character. Any carry (or borrow) from the next-lowest-order character position is ignored; no carry (or borrow) is applied to the next-highest-order character position.
AND THE CORRESPONDING V_2 CHARACTER IS AN IGNORE CODE	The V_1 character is sent to RC and RD as the result character. Any carry (or borrow) from the next-lowest-order character position is ignored; no carry (or borrow) is applied to the next-highest-order character position.	The V_1 character is sent to RC and RD as the result character. Any carry (or borrow) from the next-lowest-order character position is ignored; no carry (or borrow) is applied to the next-highest-order character position.	An Ignore code is sent to RC and RD as the result character. Any carry (or borrow) from the next-lowest-order character position is ignored; no carry (or borrow) is applied to the next-highest-order character position.

RULES FOR ADDITION (Continued)

(b)

	IF THE SIGN OF V_1 IS		
	NUMERIC (Δ,0,+ OR −)	ALPHA (all other characters except i)	AN IGNORE CODE (i)
AND THE SIGN OF V_2 IS PLUS (any character except - or i)	The Sign of the Result that is formed in RC and RD is determined in the usual algebraic manner. See Numeric Add Rules "Sign of R", above.	The V_1 Sign (alpha) is sent to RC and RD as the Sign of the Result	The V_2 Sign is sent to RC and RD as the Sign of the Result. If the V_2 Sign is Δ however, a 0 is sent to RC and RD as the Sign of the Result
AND THE SIGN OF V_2 IS MINUS (-)			
AND THE SIGN OF V_2 IS AN IGNORE CODE (i)	The V_1 Sign is sent to RC and RD as the Sign of the Result. If the V_1 Sign is Δ, however, a 0 is sent to RC and RD as the Sign of the Result		An Ignore code is sent to RC and RD as the Sign of the Result

Alpha Sign Add Rules

a. When the V_1 Sign is alpha or i, the V_2 Sign is plus (any character except -), and the process is Add or Add & Check, the arithmetic section will regard V_1 and V_2 as having like Signs and will simply add each character in V_2 to the corresponding character in V_1. (The Alpha Add Rules are applied as required in Character Positions 1-11.)

b. When the V_1 Sign is alpha or i, the V_2 Sign is minus (-), and the process is Add or Add & Check, the arithmetic section will regard V_1 and V_2 as having unlike Signs and will simply subtract each character in V_2 from the corresponding character in V_1. (The Alpha Add Rules are applied, as required, in Character Positions 1-11.) When V_1 and V_2 have unlike signs, the machine always assumes that V_2 is smaller than V_1 and attempts a "trial" subtraction of V_2 from V_1. If $V_1 > V_2$ as assumed, the "trial" subtraction is the actual subtraction. If $V_2 > V_1$, an "end borrow" is generated. This tells the arithmetic section to reverse V_1 and V_2 in RA and RB and perform the subtraction again to obtain the correct result. In Alpha Sign Add no relative magnitude of V_1 and V_2 is checked. Hence, when the V_1 Sign is alpha or i and the V_2 sign is -, only the "trial" subtraction is performed.

c. When the V_1 Sign is 0,Δ, or +, the V_2 Sign is an Ignore code, and the process is Add or Add & Check, the arithmetic section will regard V_1 and V_2 as having like Signs and will simply add each character in V_2 to the corresponding character in V_1. (The Alpha Add Rules are applied, as required, in Character Positions 1-11.)

d. When the V_1 Sign is -, the V_2 Sign is an Ignore code, and the process is Add or Add & Check, the arithmetic section will regard V_1 and V_2 as having unlike Signs and will simply subtract each character in V_2 from the corresponding character in V_1 (only the "trial" subtraction noted in b above is performed; the Alpha Add Rules are applied, as required, in Character Positions 1-11.)

7. If an Add and Check is programmed and the computer does an "alpha add", or "alpha sign add", checking is not performed.

8. If the character 11 positions of each operand are both numeric and if when an add is performed their sum is > 9, computer operation ceases and a pulse is emitted from the add/subtract overflow hub on the Program Control Plugboard. Checking is not performed.

9. Setting of Branch Storage by an Add or Add & Check Process

a. Numeric Add

V_1 Sign is Δ , 0, +, 0-
V_2 Sign is not an Ignore code
V_1 and V_2 are numeric

	Sign of R	Setting of Branch Storage
(R ≠ 0)	Δ	+
(R ≠ 0)	-	-
(R = 0)	Δ or -	0

b. Alpha Add

V_1 Sign is Δ , 0, +, or -
V_2 Sign is not an Ignore Code
V_1 and/or V_2 (apart from sign character) are numeric

Sign of R	Setting of Branch Storage
Δ	+
-	-

Note: The conditions R=0 and R≠0 are not detected when the computer performs an Alpha Add or Subtract. Branch Storage is therefore always given either a + or - setting.

c. Alpha Sign Add

V_1 Sign is not Δ , 0, +, or -
and / or
V_2 Sign is an Ignore Code

Branch Storage has a 0 setting after all Alpha Sign Add operations, regardless of the sign of the result.

OED CYCLE FOR ADD AND ADD & CHECK
(Instruction Words)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK → SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 14 → PR value of S → SR	.210	.441	.672
3	U of IRVc → SAR U of SRV → SK	.210	.441	.672
4	Load V_1 into RA	See Page II-229		
**5	V of IRVc → SAR	.210	.441	.672
	Shift V_1 in RA	.042 (n+1)		
6	Load V_2 into RB V of SRV → SK	See Page II-229		
**7	Advance PAK PAK → SAR	.210	.441	.672
	Shift V_2 in RB	.042 (n+1)		
**8	Load IRVn W of SRV → SK	See Page II-229		
	Add V_2 to V_1 algebraically and form the sum in RC and RD Check the result, using the sum in RC, if S ≠ B, C, D, E, F, G, H, or I (If S = B, C, D, E, F, G, H, or I do not check)	V_1, V_2 Numeric, Same Sign Check: 1.89 No Check: .71 V_1, V_2 Numeric, Opposite Signs, $V_1 > V_2$ Check: 1.89 No Check: .71 V_1, V_2 Numeric, Opposite Signs, $V_1 < V_2$ Check: 2.47 No Check: 1.30 Alpha Add: .71 Alpha Sign Add: .67		
**9	W of IRVc → SAR	.210	.441	.672
	Shift R in RD	.042 (n+1)		
10	Store R from RD	See Page II-230		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Conditional SRV Clear Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn.
** Times listed are simultaneous. Larger of the determines the time required for this OED Step. The Add process time to be used in OED 8 depends on the operands used (and, if these are numeric, on whether checking is suppressed or not.)

OED CYCLE FOR ADD and ADD & CHECK
(Program Steps)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)
3	V_1 ADDRESS → SAR V_1 SHIFT → SK	.042
4	Load V_1 into RA	See Page II-229
*5	V_2 ADDRESS → SAR	.042
	Shift V_1 in RA	.042 (n+1)
6	Load V_2 into RB V_2 SHIFT → SK	See Page II-229
7	(Advance PAK, PAK → SAR)** Shift V_2 in RB	.042 (n+1)
*8	(Load IRVn)** R SHIFT → SK Add V_2 to V_1 algebraically and form the sum in RC and RD Check the result, using the sum in RC if the PROCESS hub is patched to + C; do not check if PROCESS hub is patched to + NC.	V_1,V_2 Numeric, Same Sign Check: 1.89 No Check: .71 V_1,V_2 Numeric, Opposite Sign, $V_1 > V_2$ Check: 1.89 No Check: .71 V_1,V_2 Numeric, Opposite Sign, $V_1 < V_2$ Check: 2.47 No Check: 1.30 Alpha Add: .71 Alpha Sign Add: .67
*9	R ADDRESS → SAR	.042
	Shift R in RD	.042 (n+1)
10	Store R from RD	See Page II-230
11		.042
12	STEP OUT Set OED (See Page II-104)	.042

Note: The above notation (as V_1 ADDRESS → SAR and V_1 SHIFT → SK), as well as the timing listed for setting up addresses in SAR and shifts in SK, does not apply unless SAR and SK are set to an address or shift, respectively, via the plugboard. If data is shifted into these registers, as when the U, V, or W ADDRESS or SHIFT hubs are patched from a Program Step's ADDRESS or SHIFT hubs, the following notation and timing apply:

For Addresses	For Shifts	Timing (milliseconds)		
		Minimum	Average	Maximum
U, V, or W of IRVc → SAR	U, V, or W of SRV → SK	.210	.441	.672

* Times listed are simultaneous. Use shifting time as the time for this OED Step if n (number of places shifted) ≠ 0.

** These operations are performed only if this Program Step is the first in a Transcop sequence. For timing see OED 7-8 of OED cycle for corresponding Instruction Word. Apply * when these operations occur. Use time listed for this OED Step when these operations do not occur.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Subtract (algebraically) the contents of Register B from Register A, forming the difference (separately) in both Registers C & D.	PR = 22	PROCESS to - NC patching
Determine that checking is to be suppressed in this instruction	S must have one of the following values: B,C,D,E,F,G,H, or I	
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Store the final contents of Register D at the location specified by	W, or	R ADDRESS patching
Initiate Sub-Instrucion(s) or Sub-Step(s) in accordance with	S*, or	STEP OUT patching

*If S = E, the only Sub-Instruction (Suppress Check) specified by this instruction has already been completed at this time. If S has any of the other values: B,C,D,F,G,H, or I, the other Sub-Instruction(s) which these values indicate are initiated at this time.

SUBTRACT & CHECK
SB
(22)
S≠B,C,D,E,F,G,H, or I

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Subtract (algebraically) the contents of Register B from Register A, forming the difference (separately) in both Registers C and D	PR = 22	PROCESS to - C patching
Determine that checking is not to be suppressed in this instruction; and check the subtraction. (Use the difference formed in Register C for the check operation.)	S must not have one of the following values: B, C,D,E,F,G,H, or I	
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Store the final contents of Register D at the location specified by	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

RULES FOR SUBTRACTION

1. Permissible Sources for V_1 and V_2:

   Any Word Location in GSB, BTB, I/O, IS, or FS Tracks
   Any Field Location in GSB, BTB, I/O, IS, or FS Tracks
   Notes: if a Field greater than 12 characters is referred to only the lower-order 12 characters of the Field are acquired;

   if a Field less than 12 characters is referred to, the Field is loaded into the lower-character positions of the arithmetic register(s) involved (RA, & RB) and the higher-order character positions of the arithmetic register are padded with Space codes.

   RA, RB, RC, RD, or IRVc

   GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as Sources a plus (Δ) sign is automatically generated and sent as the lowest-order character of V_1 or V_2. The contents of GSAR, PAK or CDR are then loaded into the arithmetic register(s) involved (RA, RB) and the required number of Space codes are padded into the higher-order character positions.

   The entire contents of a buffer or track (Z-address) must not be specified.

2. Permissible Destinations for R:

   Any Word Location in GSB, BTB, I/O, FS, or IS Tracks
   Any Field Location in GSB, BTB, I/O, FS, or IS Tracks
   Notes: if a Field greater than 12 characters is specified, the contents of the Source of the result (RD) is loaded into lower-order character positions of the Field Location and Space codes are padded into the higher-order character positions;

   if a Field less than 12 characters is specified, only the lower-order characters in the Source of the result are sent to the Field Location.

   RA, RB, RC, RD, IRVn, or SRV

   GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as a Destination, the sign in the arithmetic register that is the Source of the result and a number of characters equal to the capacity of GSAR, PAK or CDR are sent to these locations. As the highest-order character transmitted is shifted into these locations, the sign is shifted off the right end.

   The entire contents of a buffer or track (Z-address) must not be specified.

3. All three quantities (V_1, V_2, and R) manipulated by Subtract or Subtract and Check can be shifted right, left, or right end around.

Appropriate shifts must be programmed so that

a. Correspondingly-significant characters in V_1 and V_2 are (algebraically) subtracted; and

b. The (algebraic) difference of V_1 and V_2 does not exceed 11 characters and sign.

Note: If the (algebraic) difference of V_1 and V_2 is greater than 11 characters and sign in a Subtract or Subtract & Check, a +/- OVERFLOW error occurs:

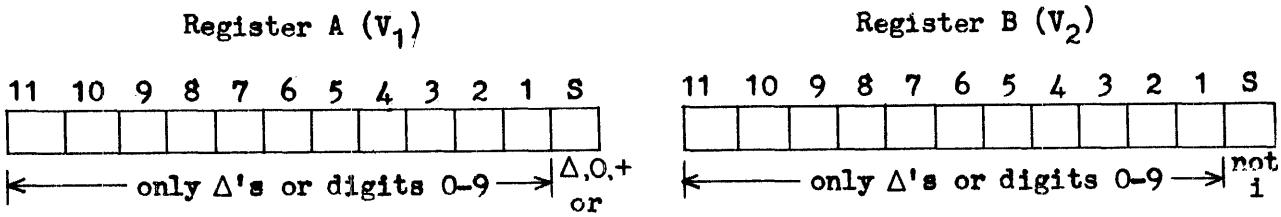
the Subtract or Subtract & Check instruction terminates; and the +/- OVERFLOW hub on the Program Control Plugboard emits.

4. Alpha and Numeric Characters in V_1 and V_2

In Subtract and Subtract & Check processes, the arithmetic section recognizes the 6 bits in Univac code of each character in RA and RB, and forms 6-bit characters in the result (in RC and RD). It accordingly can differentiate alpha and numeric characters. The alpha/numeric assignments are the same as those for Add and Add & Check.

5. Numeric Subtract Rules:

The computer applies the rules listed below when V_1 and V_2 have the following composition:



Result	V_1 and V_2 have opposite Sign		
SIGN OF THE RESULT	V_1 Sign =	Δ,0,+	-
	V_2 Sign =	-	any character but i or -
	R Sign =	Δ	-
MAGNITUDE OF THE RESULT	$\lvert V_1 \rvert + \lvert V_2 \rvert$		

RULES FOR SUBTRACTION (Continued)

Result	V_1 and V_2 have the same sign								
	$\|V_1\| > \|V_2\|$			$\|V_2\| > \|V_1\|$			$\|V_1\| = \|V_2\|$		
SIGN OF THE RESULT	V_1 Sign =	Δ, 0, +	-	V_1 Sign =	Δ, 0, +	-	V_1 Sign =	Δ, 0, +	-
	V_2 Sign =	any character but i or -	-	V_2 Sign =	any character but i or -	-	V_2 Sign =	any character but i or -	-
	R Sign =	Δ	-	R Sign =	-	Δ	R Sign =	Δ	-
MAGNITUDE OF THE RESULT	$\|V_1\| - \|V_2\|$			$\|V_2\| - \|V_1\|$			Zero		

As corresponding characters in V_1 and V_2 are subtracted (added), any borrow (carry) from the character position to the right is subtracted from (added to) the difference (sum) of the V_1 and V_2 characters, and any borrow (carry) resulting from the difference (sum) is applied to the next-higher-order character position.

Numeric Subtract and Subtract & Check are thus algebraic subtractions. Note that the Sign of the Result is either Δ (meaning plus) or - (meaning negative). In this connection it is important to also note the following special-handling of Δ's that occur in Character Positions 1-11 of V_1 and V_2:

Character Positions 1-11

If the V_1 character =	Δ	0	Δ	Δ	1-9
and the V_2 character =	Δ	Δ	0	1-9	Δ
then the Result character =	0	0	0	1-9	1-9

6. "Alpha Subtract" Rules:

(a)	WHEN V_1 CHARACTER IN CHARACTER POSITIONS 11-1 IS		
	NUMERIC	ALPHA	AN IGNORE CODE
AND THE CORRESPONDING V_2 CHARACTER IS NUMERIC	The result character formed in RC and RD is numeric; it is the difference (sum) of the absolute values of the V_1 and V_2 characters minus the borrow (or plus the carry) from the next-lowest-order character position. Any borrow (or carry) resulting from the difference (or sum) of these two characters is applied to the next highest-order character position.	The V_1 character is sent to RC and RD as the result character. Any borrow (or carry) from the next-lowest-order character position is ignored; no borrow (or carry) is applied to the next-highest-order character position.	The V_2 character is sent to RC and RD as the result character. Any borrow (or carry) from the next-lowest order character position is ignored; no borrow (or carry) is applied to the next-highest-ored character position.
AND THE CORRESPONDING V_2 CHARACTER IS ALPHA	The V_1 character is sent to RC and RD as the result character. Any borrow (or carry) from the next-lowest-order character position is ignored; no borrow (or carry) is applied to the next-highest-order character position.	The V_1 character is sent to RC and RD as the result character. Any borrow (or carry) from the next-lowest-order character position is ignored; no borrow (or carry) is applied to the next-highest-order character position.	The V_2 character is sent to RC and RD as the result character. Any borrow (or carry) from the next-lowest-order character position is ignored; no borrow (or carry) is applied to the next-highest-order character position.
AND THE CORRESPONDING V_2 CHARACTER IS AN IGNORE CODE	The V_1 character is sent to RC and RD as the result character. Any borrow (or carry) from the next-lowest-order character position is ignored; no borrow (or carry) is applied to the next-highest-order character position.	The V_1 character is sent to RC and RD as the result character. Any borrow (or carry) from the next-lowest-order character position is ignored; no borrow (or carry) is applied to the next-highest-order character position.	An Ignore Code is sent to RC and RD as the result character. Any borrow (or carry) from the next-lowest-order character position is ignored; no borrow (or carry) is applied to the next-highest-order character position.

RULES FOR SUBTRACTION (Continued)

<table>
<tr><td>(b)</td><td colspan="3">IF THE SIGN OF V_1 IS</td></tr>
<tr><td></td><td>NUMERIC</td><td>ALPHA (all other characters except i)</td><td>AN IGNORE CODE (i)</td></tr>
<tr><td>AND THE SIGN OF V_2 IS PLUS (any character except - or i)</td><td rowspan="2">The Sign of the Result that is formed in RC and RD is determined in the usual algebraic manner. See Numeric Subtract Rules "Sign of R", above.</td><td rowspan="3">The V_1 Sign (alpha) is sent to RC and RD as the Sign of the Result.</td><td rowspan="2">The V_2 Sign is sent to RC and RD as the Sign of the Result If the V_2 Sign is Δ, however, a 0 is sent to RC and RD as the Sign of the Result.</td></tr>
<tr><td>AND THE SIGN OF V_2 IS MINUS (-)</td></tr>
<tr><td>AND THE SIGN OF V_2 IS AN IGNORE CODE (i)</td><td>The V_1 Sign is sent to RC and RD as the Sign of the Result. If the V_1 Sign is Δ, however, a 0 is sent to RC and RD as the Sign of the Result.</td><td>An Ignore Code is sent to RC and RD as the Sign of the Result.</td></tr>
</table>

Alpha Sign Subtract Rules

a. When the V_1 Sign is alpha or i, the V_2 Sign is plus (any character except -), and the process is Subtract or Subtract & Check, the arithmetic section will change the V_2 Sign to minus (-), and then regard V_1 and V_2 as having unlike Signs: each character in V_2 will be subtracted from the corresponding character in V_1. (Only the "trial" subtraction referred to in Alpha Sign Add, Rule b takes place; Alpha Subtract Rules are applied, as required in Character Positions 1-11.)

b. When the V_1 Sign is alpha or i, the V_2 Sign is -, and the process is Subtract or Subtract & Check, the arithmetic section will change the V_2 Sign to plus, and then regard V_1 and V_2 as having like Signs: each character in V_2 will be added to the corresponding character in V_1 (Alpha Subtract Rules are applied, as required, in Character Positions 1-11.)

c. When the V_1 Sign is 0, Δ, or +, the V_2 Sign is an Ignore code, and the process is Subtract or Subtract & Check, the arithmetic section will regard the Ignore code as a plus, change it to a minus and then (because V_1 and V_2 have unlike signs) subtract V_2 from V_1 as in a. above.

d. When the V_1 Sign is -, the V_2 Sign is an Ignore code, and the process is Subtract or Subtract & Check, the arithmetic section will regard the Ignore code as a plus, change it to a minus and then (because V_1 and V_2 have like signs) add V_2 to V_1 as in b. above.

7. If a Subtract & Check is programmed and the computer does an "alpha subtract" or "alpha sign subtract", checking is not performed.

8. Setting of Branch Storage by a Subtract or Subtract & Check Process.

a. Numeric Subtract

V_1 Sign is Δ , 0, +, or -
V_2 Sign is not an Ignore code
V_1 and V_2 are numeric

	Sign of R	Setting of Branch Storage
(R ≠ 0)	Δ	+
(R ≠ 0)	-	-
(R = 0)	Δ or -	0

b. Alpha Subtract

V_1 Sign is Δ , 0, +, or -
V_2 Sign is not an Ignore Code
V_1 and/or V_2 (apart from sign character) are numeric

Sign of R	Setting of Branch Storage
Δ	+
-	-

Note: The conditions R=0 and R≠0 are not detected when the computer performs an Alpha Add or Subtract. Branch Storage is therefore always given either a + or - setting.

c. Alpha Sign Subtract

V_1 Sign is not △, 0, +, or -
V_2 Sign is an Ignore Code

Branch Storage has a 0 setting after all Alpha Sign Subtract operations, regardless of the sign of the result.

OED CYCLE FOR SUBTRACT AND SUBTRACT & CHECK
(Instruction Words)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK → SAR	.210	.441	.672
*1	Load IRVn		See Page II-229	
2	Switch IRV's From OP of IRVc: 22 → PR value of S → SR	.210	.441	.672
3	U of IRVc → SAR U of SRV → SK	.210	.441	.672
4	Load V_1 into RA		See Page II-229	
**5	V of IRVc → SAR	.210	.441	.672
	Shift V_1 in RA		.042 (n+1)	
6	Load V_2 into RB V of SRV → SK		See Table II-	
**7	Advance PAK, PAK → SAR	.210	.441	.672
	Shift V_2 in RB		.042 (n+1)	
**8	Load IRVn W or SRV → SK		See Page II-229	
	Subtract V_2 from V_1 algebraically and form the difference in both RC and RD Check the result, using the difference in RC, if S ≠ B, C, D, E, F, G, or I, (If S = B, C, D, E, F, G, H, or I do not check)	V_1 and V_2 Numeric, Same Sign, $V_1 > V_2$ Check: 1.89 No Check: .71 V_1 and V_2 Numeric, Same Sign, $V_2 > V_1$ Check: 2.47 No Check: 1.30 V_1 and V_2 Numeric, Opposite Signs Check: 1.89 No Check: .71 Alpha Subtract: .71 Alpha Sign Subtract: .67		
**9	W of IRVc → SAR	.210	.441	.672
	Shift R in RD		.042 (n+1)	
10	Store R from RD		See Page II-230	
11			.042	
12			.042	
13	Initiate Sub-Instruction(s) Conditional SRV Clear Set OED (See Page II-100)		.042	

* Omitted if this IW is already in IRVn.
** Times listed are simultaneous. Larger of the two determines the time required for this OED Step. The Subtract process time to be used in OED 8 depends on the operands used (and, if these are numeric, on whether checking is suppressed or not.

OED CYCLE FOR SUBTRACT AND SUBTRACT & CHECK
(Program Steps)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)
3	V_1 ADDRESS → SAR	.042
	V_1 SHIFT → SK	
4	Load V_1 into RA	See Page II-229
*5	V_2 ADDRESS → SAR	.042
	Shift V_1 in RA	.042 (n+1)
6	Load V_2 into RB	See Page II-229
	V_2 SHIFT → SK	
7	(Advance PAK, PAK → SAR)**	
	Shift V_2 in RB	.042 (n+1)
*8	(Load IRVn)**	V_1, V_2 Numeric, Same Sign, $V_1 > V_2$
	R SHIFT → SK	Check: 1.89
	Subtract V_2 from V_1 algebraically and form the difference in both RC and RD	No Check: .71
	Check the result, using the difference in RC, if S ≠ B,C,D,E,F,G,H, or I. (If S = B,C,D,E,F,G,H, or I do not check)	V_1, V_2 Numeric, Same Sign, $V_1 < V_2$
		Check: 2.47
		No Check: 1.30
		V_1, V_2 Numeric, Opposite Signs
		Check: 1.89
		No Check: .71
		Alpha Subtract: .71
		Alpha Sign Subtract: .67
*9	R ADDRESS → SAR	.042
	Shift R in RD	.042 (n+1)
10	Store R from RD	See Page II-230
11		.042
12	STEP OUT	.042
	Set OED (See Page II-104)	

Note: The above notation (as V_1 ADDRESS → SAR and V_1 SHIFT → SK), as well as the timing listed for setting up addresses in SAR and shifts in SK, does not apply unless SAR and SK are set to an address or shift, respectively, via the plugboard. If data is shifted into these registers, as when the U, V, or W ADDRESS or SHIFT hubs are patched from a Program Step's ADDRESS or SHIFT hubs, the following notation and timing apply:

For Addresses	For Shifts	Timing (milliseconds)		
		Minimum	Average	Maximum
U, V, or W of IRVc → SAR	U, V, or W of SRV → SK	.210	.441	.672

*Times listed are simultaneous. Use shifting time as the time for this OED Step if n (number of places shifted) ≠ 0.

**These operations are performed only if this Program Step is the first in a Transcop sequence. For timing see OED 7-8 of OED cycle for corresponding Instruction Word. Apply * when these operations occur. Use time listed for this OED Step when these operations do not occur.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Multiply the contents of Register A by the contents of Register B; determine the sign of the product in the usual algebraic manner; form a 22-character product as follows: form the sign of the product and the 11-lower-order digits of the program in Register D; form the sign of the product and the 11 higher-order digits of the product in Register C.	PR = 43	PROCESS to XSL NC patching
Determine that checking is to be suppressed in this instruction	S must have one of the following values: B,C,D,E,F,G,H, or I	
Shift the contents of RD and RC, or RD only, in accordance with the	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Store the final contents of Register D at the location specified by	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S*, or	STEP OUT patching

* If S = E, the only Sub-Instruction (Suppress Check) specified by this instruction has already been completed at this time. If S has any of the other values: B,C,D,F,G,H, or I, the other Sub-Instruction(s) which these values indicate are initiated at this time.

(43)
S≠B,C,D,E,F,G,H, or I

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Multiply the contents of Register A by the contents of Register B; determine the sign of the product in the usual algebraic manner; form a 22-character product as follows: form the sign of the product and the 11 lower-order digits of the product in Register D; form the sign of the product and the 11 higher-order digits of the product in Register C	PR = 43	PROCESS to XSL C patching
Determine that checking is not to be suppressed in this instruction	S must not have one of the following values: B, C,D,E,F,G,H, or I	
Store the contents of Register D at the location specified by	W, or	R ADDRESS patching
Check the multiplication		
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Multiply the contents of Register A by the contents of Register B; determine the sign of the product in the usual algebraic manner; form a 22-character product as follows: form the sign of the product and the 11 lower-order digits of the product in Register D; form the sign of the product and the 11 higher-order digits of the product in Register C.	PR = 44	PROC to XSU NC patching
Determine that checking is to be suppressed in this instruction.	S must have one of the following values: B,C,D,E,F,G,H, or I	
Shift the contents of RC and RD, or RC only, in accordance with the	contents of the W-section of the shift Revolver, or	R SHIFT patching
Store the final contents of Register C at the location specified by	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S*, or	STEP OUT patching

*If S = E, the only Sub-Instruction (Suppress Check) specified by this instruction has already been completed at this time. If S has any of the other values: B,C,D,F,G,H, or I, the other Sub-Instruction(s) which these values indicate are initiated at this time.

MULTIPLY, STORE UPPER & CHECK
MU
(44)
S≠B,C,D,E,F,G,H, or I

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Multiply the contents of Register A by the contents of Register B; determine the sign of the product in the usual algebraic manner; form a 22-character product as follows: form the sign of the product and the 11 lower-order digits of the product in Register D. form the sign of the product and the 11 higher-order digits of the product in Register C.	PR = 44	PROCESS to XSU-C patching
Determine that checking is not to be suppressed in this instruction.	S must not have one of the following values:B, C,D,E,F,G,H, or I	
Store the contents of Register C at the location specified by	W, or	R ADDRESS patching
Check the multiplication		
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

RULES FOR MULTIPLICATION

1. Permissible Sources for V_1 and V_2:

Any Word Location in GSB, BTB, I/O, IS, or FS Tracks
Any Field Location in GSB, BTB, I/O, IS, or FS Tracks
Notes: if a Field greater than 12 characters is referred to only the lower-order 12 characters of the Field are acquired;

if a Field less than 12 characters is referred to, the Field is loaded into the lower-character positions of the arithmetic register(s) involved (RA, RB) and the higher-order character positions of the arithmetic register are padded with Space codes.

RA, RB, RC, RD, or IRVc

GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as Sources, a plus (Δ) sign is automatically generated and sent as the lowest-order character of V_1 or V_2. The contents of GSAR, PAK or CDR are then loaded into the arithmetic register(s) involved (RA, RB) RB) and the required number of Space codes are padded into the higher-order character positions.

The entire contents of a buffer or track (Z-address) must not be specified.

2. Permissible Destinations for R:

Any Word Location in GSB, BTB, I/O, FS, or IS Tracks
Any Field Location in GSB, BTB, I/O, FS, or IS Tracks
Notes: if a Field greater than 12 characters is specified, the contents of the Source of the result (RD or RC) is loaded into the lower-order character positions of the Field Location and Space codes are padded into the higher-order character positions;

if a Field less than 12 characters is specified, only the lower-order characters in the Source of the result are sent to the Field Location.

RA, RB, RC, RD, IRVn, or SRV: If RA or RB are used as a Destination for R, and a check is programmed an ARITHMETIC CHECK ERROR will occur.

GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as a Destination, the sign in the arithmetic register that is the Source of the result and a number of characters equal to the capacity of GSAR, PAK or CDR are sent to these locations. As the highest-order character transmitted is shifted into these locations, the sign is shifted off the right end.

The entire contents of a buffer or track (Z-address) must not be specified.
Notes: In ML, the result is stored only from RD.
In MU, the result is stored only from RC.

If RC or RD is used as a Destination and it is also the Source of the result, in any Multiply instruction for which checking has been programmed, no result is stored.* This is due to the fact that checking takes place subsequent to the storage of the result, and the contents of RC and RD are destroyed during the check. (Both RC and RD contain - 0 after a Multiply check.) In ML if the result is stored in RC, or if in MU the result is stored in RD, an ARITHMETIC CHECK error, described in Rule 3, occurs ; and no result is permanently stored.

3. In Multiply instructions any type of shift for V_1 and V_2 can be programmed and these shifts are performed in the same manner as in any other instruction. If a check is not programmed, R can be programmed for shifting. When right or left shifts are programmed for R, RC and RD shift as one (22-character) register. When a right end around shift is programmed for R, only the register from which R is stored is shifted. If a check is also programmed, an ARITHMETIC CHECK ERROR occurs: computer operation stops immediately, the ARITHMETIC CHECK ERROR hubs emit, and the "check" error is indicated.

4. In all Multiply instructions, the arithmetic section recognizes only the excess-three bits of the characters in V, and V_2, and forms only digits 0-9 in the Result. If alpha characters with the same excess-three bits as the digits 0-9, therefore, are used in V_1 and V_2, they are treated as if they were their digit equivalent in excess-three code. Alpha characters other than those with the same excess-three bits as the digits 0-9 will cause an invalid (meaningless) multiplication to occur; such characters should never be used.

5. All characters in the Sign Position of V_1 and V_2, except -, are plus signs; only - is negative. The Sign of the Result is determined in the usual algebraic manner as follows:

Sign of V_1 =	any character but -	any character but -	-	-
Sign of V_2 =	any character but -	-	any character but -	-
Sign of Result =	Δ	-	-	Δ

6. The magnitude of the Result is simply the product of the absolute values of V_1 and V_2. Since RC and RD act as one (22-character) register in Multiply instructions, no "overflow" can result. The computer treats V_1 and V_2 as integers and forms an integer for the Result. Any decimal connotation for machine numbers is strictly a programming matter. When decimal numbers are multiplied, the number of decimal places in the Result

* That is, permanently stored

is, of course, the sum of the decimal places in V_1 and V_2. If it is desirable to have a particular number of decimal places in the Result, appropriate shifts for V_1, V_2, and/or R must be programmed as required.

7. The Sign of the Result and the 11 lower-order characters of the Result (product) are formed in RD; the Sign of the Result and the 11 higher-order characters of the Result (product) are formed in RC. To illustrate this, a few (types of) products are listed on the next page.

8. The Value V_1 in RA is the multiplicand; the value V_2 in RB is the multiplier. The sum of the values of the digits in the multiplier determines how many times the basic cycle in the Multiply algorithm is carried out. See .588 $(m_{11}+m_{10}+\cdots+m_1)$ in OED cycle for Multiply processes. To speed up programs, that quantity the sum of the values of whose digits is the least should be programmed for V_2, wherever possible.

9. The multiplicand and multiplier are left undisturbed in RA and RB, respectively, at the conclusion of all Multiply instructions. As noted above, if checking is programmed RC and RD = -0 after the check. If no check is programmed RC and RD retain the same product they stored at OED 10 time.

10. Setting of Branch Storage by Multiply Instructions

Sign of R		Setting of Branch Storage
(R≠0)	Δ	+
(R≠0)	-	-
(R=0)	Δ or -	0*

*Branch Storage is set to 0 at OED 7 time. If R=0 is detected, no + or - setting is given Branch Storage.

	Register C 11 10 9 8 7 6 5 4 3 2 1 S	Register D 11 10 9 8 7 6 5 4 3 2 1 S
$V_1 \neq 0$; $V_2 = 0$ (V_1 and V_2 have unlike Signs) R = -0	Δ Δ Δ Δ Δ Δ Δ Δ Δ Δ 0 -	0 0 0 0 0 0 0 0 0 0 0 -
$V_1 \neq 0$; $V_2 = 0$ (V_1 and V_2 have like Signs) R = +0	Δ Δ Δ Δ Δ Δ Δ Δ Δ Δ 0 Δ	0 0 0 0 0 0 0 0 0 0 0 Δ
$V_1 = 0$; $V_2 = 185$ (V_1 and V_2 have unlike Signs) R = -0	Δ Δ Δ Δ Δ Δ Δ 0 0 0 -	0 0 0 0 0 0 0 0 0 0 0 -
$V_1 = 0$; $V_2 = 87654321$ (V_1 and V_2 have like Signs) R = +0	Δ Δ Δ 0 0 0 0 0 0 0 0 Δ	0 0 0 0 0 0 0 0 0 0 0 Δ
$V_1 = 1234567+$; $V_2 = 123-$ R ≠ 0, and number of digits in product < 11	Δ Δ Δ Δ Δ Δ Δ 0 0 0 -	0 0 1 5 1 8 5 1 7 4 1 -
$V_1 = 7777777+$; $V_2 = 6666+$ R ≠ 0, and number of digits in product = 11	Δ Δ Δ Δ Δ Δ Δ 0 0 0 0 Δ	5 1 8 4 6 6 6 1 4 8 2 Δ
$V_1 = 1234567891+$; $V_2 = 123456-$ R ≠ 0, and number of digits in product is > 11	Δ Δ Δ Δ Δ 0 0 1 5 2 4 -	1 4 8 1 3 5 5 1 2 9 6 -
$V_1 = 9999999990+$; $V_2 = 9999999990+$ R ≠ 0, and a 22-digit product is formed	9 9 9 9 9 9 9 9 9 8 0 Δ	0 0 0 0 0 0 0 0 1 0 0 Δ

If u and v represent the number of digits in V_1 and V_2, respectively, after any shifts that may have been programmed for V_1 and V_2 have been carried out, then u+v or u+v-1 is the number of product digits.

When the product is less than 22 digits, the number of Δ's in the higher-order digit positions of RC is 11-v. (Zero is treated as a one-digit number if V_1 or V_2.)

OED CYCLE FOR MULTIPLY STORE UPPER, and MULTIPLY STORE LOWER
(Instruction Words)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 43→PR (xSL) 44→PR(xSU) value of S→SR	.210	.441	.672
3	U of IRVc→SAR U of SRV→SK	.210	.441	.672
4	Load V_1 into RA	See Page II-229		
**5	V of IRVc→SAR	.210	.441	.672
	Shift V_1 in RA	.042 (n+1)		
6	Load V_2 into RB V of SRV→SK	See Page II-229		
**7	Advance PAK PAK→SAR	.210	.441	.672
	Shift V_2	.042 (n+1)		
**8	Load IRVn W of SRV→SK	See Page II-229		
	Multiply V_1 by V_2 algebraically; form the sign of the product and the 11 lower-order digits of the product in RD; form the sign of the product and 11-higher-order digits of the product in RC	$1.60 + .588\ (m_{11} + m_{10} + \ldots m_{11})$ where $(m_{11} + m_{10} + \ldots + m_1)$ is the sum of the value of the digits in the multiplier (V_2)		
**9	W of IRVc→SAR	.210	.441	.672
	Shift R in RD (xSL) or in RC (xSU). Shift for R is not permitted if checking is programmed; right end around occurs in register from which R is stored.	.042 (n+1)		
10	Store R from RD (xSL) or from RC (xSU)	See Page II-230		
11	Check the multiplication, unless S = B,C,D,E,F,G,H, or I.	Check $1.00 + (m_{11} + m_{10} + \ldots m_1)$ where $(m_{11} + m_{10} + \ldots + m_1)$ is the value of the digits in the multiplier (V_2) No Check .042		
12		.042		
13	Initiate Sub-Instruction(s) Conditional SRV Clear Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn.

** Times listed are simultaneous. Larger of the two determines the time required for this OED Step.

OED CYCLE FOR MULTIPLY STORE UPPER and MULTIPLY STORE LOWER
(Program Steps)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)
3	V_1 ADDRESS → SAR V_1 SHIFT → SK	.042
4	Load V_1 into RA	See Page II-229
*5	V_1 ADDRESS → SAR	.042
	Shift V_1 in RA	.042 (n+1)
6	Load V_2 into RB V_2 SHIFT → SK	See Page II-229
7	(Advance PAK, PAK → SAR)** Shift V_2 in RB	.042 (n+1)
*8	(Load IRVn)** R SHIFT → SK Multiply V_1 by V_2; form the sign of the product and the 11 lower-order digits of the product in RD; form the sign of the product and the 11 highest-order digits of the product in RC.	1.60 + .588 ($m_{11} + m_{10} + \ldots + m_1$) where ($m_{11} + m_{10} + \ldots + m_1$) is the sum of the value of the digits in the multiplier (V_2)
*9	R ADDRESS → SAR	.042
	Shift R in RD (xSL) or in RC (xSU) (Programmed shift for R not permissible if checking is part of this IW; right end around shift takes place only in register from which result is stored.	.042 (n+1)
10	Store R from RD (xSL) or from RC (xSU)	See Page II-230
11	Check the multiplication if the PROCESS hub is patched to either xSL C or xSU C; do not check the multiplication if the PROCESS hub is patched to either xSL NC or xSU NC.	1.00 + ($m_{11} + m_{10} + \ldots + m_1$) Check where ($m_{11} + m_{10} + \ldots + m_1$) is the sum of the value of the digits in the multiplier (V_2) No Check .042
12	STEP OUT Set OED (See Page II-104)	.042

The above notation (as V_1 ADDRESS → SAR and V_1 SHIFT → SK), as well as timing listed for setting up addresses in SAR and shifts in SK, does not apply unless SAR and SK are set to an address or shift, respectively, via the plugboard. If data is shifted into these registers, as when the U, V, or W ADDRESS or SHIFT hubs are patched from a Program Step's ADDRESS or SHIFT hubs, the following notation and timing apply:

For Addresses	For Shifts	Timing (milliseconds)		
		Min.	Av.	Max.
U, V, or W of IRVc → SAR	U, V, or W of SRV → SK	.210	.441	.672

* Times listed are simultaneous. Use shifting time for this OED Step if n (number of places shifted) ≠ 0.

** These operations are performed only if this Program Step is the first in a Transcop sequence. For timing see OED 7-8 of OED cycle for corresponding Instruction Word. Apply * when these operations occur. Use time listed for this OED Step when these operations do not occur.

DIVIDE, STORE QUOTIENT
DQ
(48)
S=B,C,D,E,F,G,H, or I

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Registers A and C, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Registers A* and C in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Divide the contents of Register A by the contents of Register B; determine the sign of the quotient in the usual algebraic manner; and form a quotient and remainder as follows: form the quotient in Register D (up to 11 characters and the correct sign); form the remainder in Register C (sign of remainder same as sign of dividend)	PR = 48	PROCESS to ± SQ NC patching
Determine that checking is to be suppressed in this instruction.	S must have one of the following values: B,C,D,E,F,G,H, or I	
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Store the final contents of Register D at the location specified by	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S**, or	STEP OUT patching

* Register A is not actually shifted. See RULES FOR DIVISION
**If S = E, the only Sub-Instruction (Suppress Check) specified by this instruction has already been completed at this time. If S has any of the other values: B,C,D,F,G,H, or I the other Sub-Instructions which these values indicate are initiated at this time.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Registers A and C, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Registers A* and C in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Divide the contents of Register A by the contents of Register B; determine the sign of the quotient in the usual algebraic manner; and form a quotient and remainder as follows: form the quotient in Register D (up to 11 characters and the correct sign); form the remainder in Register C (sign of remainder same as sign of dividend)	PR = 48	PROCESS to ♦ SQ C patching
Determine that checking is not to be suppressed in this instruction.	S must not have one of the following values: B, C,D,E,F,G,H, or I	
Store the final contents of Register D at the location specified by	W, or	R ADDRESS patching
Check the Division		
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

* Register A is not actually shifted. See RULES FOR DIVISION

DIVIDE, STORE REMAINDER
DR
(49)
S=B,C,D,E,F,G,H, or I

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A and C, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Registers A* and C in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Divide the contents of Register A by the contents of Register B; determine the sign of the quotient in the usual algebraic manner; and form a quotient and remainder as follows: form the quotient in Register D (up to 11 characters and the correct sign); form the remainder in Register C (sign of remainder same as sign of dividend.)	PR = 49	PROCESS to ÷ SR NC patching
Determine that checking is to be suppressed in this instruction.	S must have one of the following values: B,C,D,E,F,G,H, or I	
Shift the contents of Register C in accordance with the	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Store the final contents of Register C at the location by	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S**, or	STEP OUT patching

* Register A is not actually shifted. See RULES FOR DIVISION

**If S = E, the only Sub-Instruction (Suppress Check) specified by this instruction has already been completed at this time. If S has any of the other values: B,C,D,F,G,H, or I, the other Sub-Instructions which these values indicate are initiated at this time.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A and C, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Registers A* and C in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Divide the contents of Register A by the contents of Register B; determine the sign of the quotient in the usual algebraic manner; and form a quotient and remainder as follows: form the quotient in Register D (up to 11 characters and the correct sign); form the remainder in Register C (sign of remainder same as sign of dividend)	PR = 49	PROC to ÷ SR C patching
Determine that checking is not to be suppressed in this instruction.	S must not have one of the following values: B,C,D, E,F,G,H, or I	
Store the final contents of Register C at the location specified by the	W, or	R ADDRESS patching
Check the division		
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

* Register A is not actually shifted. See RULES FOR DIVISION

RULES FOR DIVISION

1. Permissible Sources for V_1 and V_2: (V_1 is placed in both RA and RC. See Rule 3.)

Any Word Location in GSB, BTB, I/O, IS, or FS Tracks
Any Field Location in GSB, BTB, I/O, IS, or FS Tracks
Notes: if a Field greater than 12 characters is referred to only the lower-order 12 characters of the Field are acquired;

if a Field less than 12 characters is referred to, the Field is loaded into the lower-character positions of the arithmetic register(s) involved (RA, RC, RB) and the higher-order character positions of the arithmetic register are padded with Space codes.

RA, RB, RC, RD, or IRVc

GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as Sources, a plus (Δ) sign is automatically generated and sent as the lowest-order character of V_1 or V_2. The contents of GSAR, PAK or CDR are then loaded into the arithmetic register(s) involved (RA, RC, RB) and the required number of Space codes are padded into the higher-order character positions.

The entire contents of a buffer or track (Z-address) must not be specified.

2. Permissible Destinations for R:

Any Word Location in GSB, BTB, I/O, FS, or IS Tracks
Any Field Location in GSB, BTB, I/O, FS, or IS Tracks
Notes: if a Field greater than 12 characters is specified, the contents of the Source of the result (RD or RC) is loaded into the lower-order character positions of the Field Location and Space codes are padded into the higher-order character positions;

if a Field less than 12 characters is specified, only the lower-order characters in the Source of the result are sent to the Field Location.

RA, RB, RC, RD, IRVn, or SRV: If RA or RB are used as a Destination for R, and a check is programmed an ARITHMETIC CHECK ERROR will occur.

GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as a destination, the sign in the arithmetic register that is the Source of the result and a number of characters equal to the capacity of GSAR, PAK or CDR are sent to these locations. As the highest-order character transmitted is shifted into these locations, the sign is shifted off the right end.

The entire contents of a buffer or track (Z-address) must not be specified.

Notes: In DQ, the result (quotient) is stored only from RD.
In DR, the result (remainder) is stored only from RC.

If RC or RD is used as a Destination and it is also the Source of the result in any Divide instruction for which checking has been programmed, no result is stored.* This is due to the fact that checking takes place subsequent to the storage of the result, and the contents of RC and RD are destroyed during the check. In DQ if the result is stored in RC, or if in DR the result is stored in RD, an ARITHMETIC CHECK ERROR, described below in Rule 3, R Shifts, occurs. (In these cases, also, no result is permanently stored.)

3. All V_1 shifts in a Divide Instruction, regardless of whether they are programmed as Left, Right, or Right End around are treated as left shifts by the computer. A shift, any type, can be programmed for V_2 or R. However, the following factors usually determine what shifts are programmed:

V_1 and V_2 Shifts

(a) Number of digits in the Quotient

If u = number of digits in V_1, and
v = number of digits in V_2 (after shifting, if any V_2 shifts are programmed), and
n = number of programmed shifts for V_1;

then Q = u-v+n+1 is the number of digits (sign excluded) in the quotient.

If Q > 11 is detected by the arithmetic section, the division is not performed and a DIVIDE OVERFLOW error occurs: computer operation stops immediately, the ÷ OVERFLOW hubs on the Program Control Plugboard emit, and the "overflow" condition is indicated.

If Q = (u-v+n+1) < 1 is detected, the machine cannot divide V_1 by V_2. This situation is described as "Divide Underflow". It does not cause an error; however, no division is performed. To insure that a correct "result" is obtained in this situation, V_1 is always placed in RA and RC in a Divide instruction; and the programmed shifting for V_1 is carried out in RC at V_1 Shift time. When Q < 1 is detected at process time, then, the correct remainder (i.e., the shifted dividend, or the dividend if no shifting is programmed) is already in RC. RD is sent Zeros (and the zero "quotient" is plus or minus in accordance with the algebraic sign rules for division).

(b) Number of decimal places desired in Quotient

As in all other operations, the machine treats V_1 and V_2 as integers. Any decimal connotation is strictly a matter of programming. If no V_1 or V_2 shifts are programmed, the number of decimal places in the quotient

* That is, permanently stored.

is the number of decimal places in the dividend (V_1) minus the number of decimal places in the divisor (V_2). This can be expressed in equation form as

$$R_d = V_{1d} - V_{2d}$$

where R_d = the number of decimal places in the quotient
V_{1d} = number of decimal places in V_1
V_{2d} = number of decimal places in V_2

However, if desirable, a particular number of decimal places in the quotient can be programmed provided the basic rule that Q must be ≤ 11 is observed. The procedure to follow in determining the V_1 and/or V_2 shifts necessary to obtain a quotient with a particular number of decimal places is shown on the next page. The number of decimal places in the remainder is always the same as the number of decimal places in the original V_1 (dividend).

R Shifts

If a check is programmed, a shift for R must not be programmed. If it is, an ARITHMETIC CHECK error occurs: computer operation stops immediately, the ARITHMETIC CHECK ERROR hubs emit, and the "check" error is indicated.

4. In all Divide instructions, the arithmetic section recognizes only the excess-three bits of the characters in V_1 and V_2, and forms only digits in the quotient and remainder. If alpha characters with the same excess-three bits as the digits 0-9, therefore, are used in V_1 and V_2 they are treated as if they were their digit equivalent in excess-three code. Alpha characters other than those with the same excess-three bits as the digits 0-9 will cause an invalid (meaningless) division to occur; such characters should never be used. *

5. All characters in the Sign Position of V_1 and V_2, except -, are plus signs; only - is negative. The Sign of the quotient is determined in the usual algebraic manner as follows

Sign of V_1 (dividend)	any character but -	any character but -	-	-
Sign of V_2 (divisor)	any character but -	-	any character but -	-
Sign of Quotient	Δ	-	-	Δ

* only 0, Δ, -, and i are regarded as non-significant in that portion of the divide algorithm wherein the operands are normalized. Among other things this means that ";", ")", and "+" although they have the same excess-three bits as the digit 0 will be significant and stop the normalize operation when they are detected.

When $[R_d-(V_{1d}-V_{2d})]$ is	Program the following shifts			
	V_1		V_2	
	Type of Shift	Number of Places	Type of Shift	Number of Places
(a) positive and < 12	Left*	$\lvert R_d-(V_{1d}-V_{2d})\rvert$	none	none
(b) positive and ≥ 12	Left*	11	Right**	$[R_d-(V_{1d}-V_{2d})]-11$
(c) minus	none	none	Left***	$\lvert R_d-(V_{1d}-V_{2d})\rvert$

* Programmed left shifts of V_1 in the Divide instruction are not actually performed in RA; the effect of a programmed V_1 left shift in the Divide algorithm is to increase the magnitude of the dividend (i.e., the higher-order digits in V_1 are not lost or disregarded, but V_1 is "made larger"). The "u" portion of the "U" section of the shift revolver is ignored when shifting the V_1 operand in a Divide operation. V_1 is always shifted left xx places.

** Note here that significance is lost in V_2 (i.e., the lower-order digits in RB are shifted out and lost).

*** A right shift of R (same number of places) can be used, as an alternate shift, provided checking is not programmed.

Examples

	R_d	V_{1d}	V_{2d}	$[R_d-(V_{1d}-V_{2d})]$ =	Programmed Shifts
(a)	2	2	1	$[2-(2-1)] = 1$	V_1 left shift of 1
(b)	6	1	8	$[6-(1-8)] = 13$	V_1 left shift of 11 V_2 right shift of 2
(c)	3	6	2	$[3-(6-2)] = -1$	V_2 left shift of 1 or R right shift of 1 if check is not programmed.

RULES FOR DIVISION (Continued)

The Sign of the remainder is always the Sign of the dividend (V_1). If the Sign of V_1 is any character but -, the Sign of the remainder is Δ. If the Sign of V_1 is -, the Sign of the remainder is -.

6. Magnitude of Quotient and Remainder

	Normal Division $Q = u-v+n+1 > 1$ but $\leqq 11$	Division by Zero $V_2 = 0$	Divide "Underflow" $Q = u-v+n+1 < 1$
Magnitude of the Quotient	The number of integral times the absolute magnitude of the divisor (V_2) is contained in the absolute value of the dividend (V_1). Note: if $V_1 = 0$, the magnitude of the quotient is 0	Zero	Zero
Magnitude of the Remainder	$\lvert V_1 \rvert - \lvert (Q \times V_2) \rvert$ Note: if $V_1 = 0$, the magnitude of the remainder as well as that of the quotient=0	Zero	Absolute magnitude of the dividend (V_1)

7. At the conclusion of all Divide instructions, the dividend and divisor are destroyed in RA and RB, respectively. If a check is programmed; the contents of RC (remainder) and RD (quotient) are destroyed. If a check is not programmed the contents of RC and RD are the same as they were when the result was stored. (See also Rule 9)

8. Setting of Branch Storage by Divide instructions. (Determined by Quotient)

Sign of R	Setting of Branch Storage
Δ	+
-	-

The conditions Quotient = 0 and Quotient $\neq$ 0 are not detected in connection with the setting of Branch Storage. Branch Storage is always set to + or - after a Divide instruction.

9. If the Quotient is less than 11 digits and Sign, the higher-order digit positions in RD are filled with Zeros.

The composition of RC at the end of a division is determined as follows:

> The remainder digits are located in the lower-order digit positions of RC. The higher order digit positions of RC contain Space Codes. (These are placed in RC when the remainder is shifted right to get it in correct position at the end of the divide algorithm. The number of Space codes is 11-u-n.) The other (intermediate) digit positions of RC are filled with Zeros.

OED CYCLE FOR DIVIDE, STORE QUOTIENT and DIVIDE, STORE REMAINDER
(Instruction Words)

OED	PRINCIPAL EVENTS	TIMING (Milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 48→PR (÷SQ) 49→PR (÷SR) value of S→SR	.210	.441	.672
3	U of IRVc→SAR U of SRV→SK	.210	.441	.672
4	Load V_1 into RA and RC	See Page II-229		
**5	V of IRVc→SAR	.210	.441	.672
	Shift V_1 in RC	.042 (n+1)		
6	Load V_2 into RB V of SRV→SK	See page II-229		
**7	Advance PAK PAK→SAR	.210	.441	.672
	Shift V_2 in RB	.042 (n+1)		
8	Load IRVn W of SRV→SK	See Page II-229		
	Divide V_1 by V_2 algebraically; form the quotient (with proper sign) in RD; form the remainder (with sign of V_1) in RC.	See Page II-194		
**9	W or IRVc→SAR	.210	.441	.672
	Shift R in RD (÷SQ) or in RC (÷SR) (Programmed shift for R not permissible if checking is part of this IW.)	.042 (n+1)		
10	Store R from RD (÷SQ) or from RC (÷SR)	See Page II-230		
11	Check the division unless S = B,C,D,E,F,G,H, or I.	Check: $1.6 + .042\,(11-v-n) + .588\,(q_{u-v+n+1} + q_{u-v+n} + \ldots + q_1)$ No Check: .042		
12		.042		
13	Initiate Sub-Instruction(s) Conditional SRV Clear Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn.

** Times listed are simultaneous. Larger of the two determines the time required for this OED Step.

OED CYCLE FOR DIVIDE, STORE QUOTIENT and DIVIDE, STORE REMAINDER
(Program Steps)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)
3	V_1 ADDRESS → SAR V_1 SHIFT → SK	.042
4	Load V_1 into RA and RC	See Page II-229
*5	V_2 ADDRESS → SAR Shift V_1 in RC	.042 .042 (n+1)
6	Load V_2 into RB V_2 SHIFT → SK	See Page II-229
7	(Advance PAK, PAK → SAR)** Shift V_2 in RB	.042 (n+1)
8	(Load IRVn)** R SHIFT → SK Divide V1 by V2 algebraically; form the quotient (with proper sign) in RD; form the remainder (with sign of V_1) in RC.	See Page II-194
	R ADDRESS → SAR	.042
	Shift R in RD (÷SQ) or in RC (÷SR) (Programmed shift for R not permissible if checking is part of this IW.)	.042 (n+1)
10	Store R from RD (÷SQ) or from RC (÷SR).	See Page II-230
11	Check the division if the PROCESS hub is patched to either ÷ SQ C or ÷ SR C; do not check the division if the PROCESS hub is patched to either ÷ SQ NC or ÷ SR NC	Check $1.6 + .042\,(11-v-n) + .588\,(q_{u-v+n+1} + q_{u-v+n} + \cdots + q_1)$ No Check .042
12	STEP OUT Set OED (See Page II-104)	.042

Note: The above notation (as V_1 ADDRESS → SAR and V_1 SHIFT → SK) as well as the timing listed for setting up addresses in SAR and shifts in SK, does not apply unless SAR and SK are set to an address or shift, respectively, via the plugboard. If data is shifted into these registers, as when the U, V, or W ADDRESS or SHIFT hubs are patched from a Program Step's ADDRESS or SHIFT hubs, the following notation and timing apply:

For Addresses	For Shifts	Timing (milliseconds)		
		Min.	Av.	Max.
U, V, or W of IRVc → SAR	U, V, or W of SRV → SK	.210	.441	.672

* Times listed are simultaneous. Use shifting time as the time for this OED Step if n (number of places shifted) ≠ 0.

** These operations are performed only if this Program Step is the first in a Transcop sequence. For timing see OED 7-8 of OED cycle for corresponding Instruction Word. Apply * when these operations occur. Use time listed for this OED Step when these operations do not occur.

TIMING (MILLISECONDS)
DIVIDE STORE QUOTIENT and DIVIDE STORE REMAINDER
INSTRUCTION WORDS and PROGRAM STEPS

u = the number of digits in V_1

v = the number of digits in V_2 (after shifting)

n = the number of programmed shifts for V_1

$(u-v+n+1)$ = the number of digits in the quotient

$q_{u-v+n+1}$ = the highest order digit of the quotient

q_{u-v+n} = the second highest order digit of the quotient

q_1 = the lowest order digit of the quotient

$(q_{u-v+n+1} + q_{u-v+n} + \cdots + q_1$ = the sum of the value of the digits in the quotient

Normal Division: $.042\ (26-u-v) + 1.2\ (u-v+n+1) + .462 - .042v + .588\ (q_{u-v+n+1} + q_{u-v+n} \cdots + q_1)$

Divisor (V_2) = 0: 1.092

Dividend (V_1) = 0: $1.596 - .042v$

Divide Overflow: $.042\ (26-u-v)$

$(u-v+n+1) < 1$: $.042\ (26-u-v) + .504$

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Right end around shift Registers A and B; beginning with the sign position, examine each of the 12 characters of the word V_2 held in Register B: if a V_2 character is a Zero or a Space code, transmit the corresponding V_1 character to a correspondingly significant character position in Register C and D; if the V_2 character is not Zero or Space, transmit an Ignore code to the correspondingly significant character position in Register C and D.	PR = 42	PROC to MK patching
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Store the final contents of Register D at the location specified by	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

RULES FOR MASK TRANSFER INSTRUCTIONS

1. Permissible Sources for V_1 and V_2:

   Any Word Location in GSB, BTB, I/O, IS, or FS Tracks
   Any Field Location in GSB, BTB, I/O, IS, or FS Tracks
   Notes: if a Field greater than 12 characters is referred to only the lower-order 12 characters of the Field are acquired;

   if a Field less than 12 characters is referred to, the Field is loaded into the lower-character positions of the arithmetic register(s) involved (RA, RB) and the higher-order character positions of the arithmetic register are padded with Space codes.
   RA, RB, RC, RD, or IRVc

   GSAR, PAK, CDR: if GSAR, PAK, or CDR are specified as Sources, a plus (Δ) sign is automatically generated and sent as the lowest-order character of V_1 or V_2. The contents of GSAR, PAK or CDR are then loaded into the arithmetic register(s) involved (RA, RB) and the required number of Space codes are padded into the higher-order character positions.
   The entire contents of a buffer or track (Z-address) must not be specified.

2. Permissible Destinations for R:

   Any Word Location in GSB, BTB, I/O, FS, or IS Tracks
   Any Field Location in GSB, BTB, I/O, FS, or IS Tracks
   Notes: if a Field greater than 12 characters is specified, the contents of the Source of the result (RD) is loaded into lower-order character positions of the Field Location and Space codes are padded into the higher-order character positions;

   if a Field less than 12 characters is specified, only the lower-order characters in the Source of the result are sent to the Field Location.

   RA, RB, RC, RD, IRVn, or SRV

   GSAR, PAK, CDR: if GSAR, PAK, or CDR are specified as a Destination, the sign in the arithmetic register that is the Source of the result and a number of characters equal to the capacity of GSAR, PAK or CDR are sent to these locations. As the highest-order character transmitted is shifted into these locations, the sign is shifted off the right end.
   The entire contents of a buffer or track (Z-address) must not be specified.

3. A Zero or a Space code in a V_2 character position, sign included, will cause the corresponding character in V_1 to be transmitted to RD and RC. (No exceptions, or special cases, as Δ changed to 0 when sent to result, etc.)

4. If a V_2 character position contains anything but a Zero or a Space code

(sign position included) an Ignore code is sent to the correspondingly-significant character position in RD and RC.

5. The V_1 operand and the (masking) V_2 operand are restored in RA, RB and RC respectively.

6. Shifts, any type, can be programmed for V_1, V_2, and R.

OED CYCLE FOR MASK TRANSFER
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK → SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 42 → PR value of S → SR	.210	.441	.672
3	U of IRVc → SAR U of SRV → SK	.210	.441	.672
4	Load V_1 into RA	See Page II-229		
**5	V of IRVc → SAR	.210	.441	.672
	Shift V_1 in RA	.042 (n+1)		
6	Load V_2 into RB V of SRV → SK	See Page II-229		
**7	Advance PAK PAK → SAR	.210	.441	.672
	Shift V_2 in RB	.042 (n+1)		
**8	Load IRVn W or SRV → SK	See Page II-229		
	V_1 → (V_2 MASK) → RC and RD	.63		
**9	W of IRVc → SAR	.210	.441	672
	Shift R in RD	.042 (n+1)		
10	Store R from RD	See Page II-230		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn
** Times listed are simultaneous. Larger of the two determines the time required for this OED Step. ("n" is the number of places V_1, V_2, or R is shifted)

OED CYCLE FOR MASK TRANSFER
(Program Step)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)
3	V_1 ADDRESS → SAR	.042
	V_1 SHIFT → SK	
4	Load V_1 into RA	See Page II-229
*5	V_2 ADDRESS → SAR	.042
	Shift V_1 in RA	.042 (n+1)
6	Load V_2 in RB	See Page II-229
	V_2 SHIFT → SK	
7	(Advance PAK, PAK → SAR)**	
	Shift V_2 in RB	.042 (n+1)
8	(Load IRVn)**	
	R SHIFT → SK	
	V_1 → (V_2 Mask) → RC and RD	.63
*9	R ADDRESS → SAR	.042
	Shift R in RD	.042 (n+1)
10	Store R from RD	See Page II-230
11		.042
12	STEP OUT	
	Set OED (See Page II-104)	.042

Note: The above notation (as V_1 ADDRESS → SAR and V_1 SHIFT → SK) as well as the timing listed for setting up addresses in SAR and shifts in SK, does not apply unless SAR and SK are set to an address or shift, respectively, via the plugboard. If data is shifted into these registers, as when the U, V, or W ADDRESS or SHIFT hubs are patched from a Program Step's ADDRESS or SHIFT hubs, the following notation and timing apply:

For Addresses	For Shifts	Timing (milliseconds)		
		Minimum	Average	Maximum
U, V, or W of IRVc → SAR	U, V, or W of SRV → SK	.210	.441	.672

* Times listed are simultaneous. Use shifting time as the time for this OED Step if n (number of places shifted) ≠ 0.

** These operations are performed only if this Program Step is the first in a Transcop sequence. For timing see OED 7-8 of OED cycle for corresponding Instruction Word. Apply * when these operations occur. Use time listed for this OED Step when these operations do not occur.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Place, in Register B, the contents of the location specified by	V, or	V_2 ADDRESS patching
Shift the contents of Register B in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Left end around Shift Registers A and B, and on a bit by bit basis compare (highest-order character first) the word V_1 held in Register A with the word V_2 held in Register B to determine the relative magnitude of these two words: If $V_1 > V_2$, set Branch Storage to + If $V_1 = V_2$, set Branch Storage to 0 If $V_1 < V_2$, set Branch Storage to -	PR = 37	PROCESS to COMP patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

RULES FOR COMPARE INSTRUCTIONS

1. Permissible Sources for V_1 and V_2:

   Any Word Location in GSB, BTB, I/O, IS, or FS Tracks
   Any Field Location in GSB, BTB, I/O, IS, or FS Tracks
   Notes: if a Field greater than 12 characters is referred to, only the lower-order 12 characters of the Field are acquired;

   if a Field less than 12 characters is referred to, the Field is loaded into the lower-character positions of the arithmetic register(s) involved (RA, RB) and the higher-order character positions of the arithmetic register are padded with Space codes.

   RA, RB, RC, RD, or IRVc.

   GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as Sources, a plus (Δ) sign is automatically generated and sent as the lowest-order character of V_1 or V_2. The contents of GSAR, PAK or CDR are then loaded into the arithmetic register(s) involved (RA, RB) and the required number of Space codes are padded into the higher-order character positions.

   The entire contents of a buffer or track (Z-address) must not be specified.

2. The result of a Compare process is set up in Branch Storage, not stored in an addressable memory location. Branch Storage is set to + , 0, or - depending on whether $V_1 > V_2$, $V_1 = V_2$, or $V_1 < V_2$. See Rule 7 below.

3. If a character in V_1 or V_2 is an Ignore code, comparison is suppressed for that character position.

4. Each of the sets of values for V_1 and V_2 listed in the following table are not compared when they are detected (i.e., the effect is the same as if an Ignore code were in V_1 or V_2) unless a Condition Compare Sub-Step has been executed since the last Compare process.

V_1 =	0	Δ	0	Δ
V_2 =	Δ	0	0	Δ
	a	b	c	d

   If a Condition Compare Sub-Step has been executed since the last Compare process, each of the sets of values listed above is compared. If Set a is detected, V_1 will compare as greater than V_2 for that character position; if Set b is detected, V_1 will compare as less than V_2; and only Sets c and d will compare equally.

5. Shifts, any type, for V_1 and V_2 can be programmed. A programmed shift for R (or an R address) is simply ignored.

6. The V_1 and V_2 operands are left unaltered in RA and RB, respectively.

7. Setting of Branch Storage in the Compare Process

NOTE: In V_1 and V_2 in the Compare Process all characters except a minus sign are plus in the Sign Position.

	Branch Storage is set to:
V_1 and V_2 have like Signs:	
$\lvert V_1 \rvert > \lvert V_2 \rvert$	(+)
$\lvert V_1 \rvert = \lvert V_2 \rvert$	(0)
$\lvert V_1 \rvert < \lvert V_2 \rvert$	(−)
V_1 and V_2 have unlike Signs:	
If the V_1 Sign is plus (or if V_1 is ± 0) and the V_2 Sign is minus (but V_2 is not - 0), then $V_1 > V_2$ is detected	+
If V_1 = - 0 and V_2 = + 0 or vice versa, $V_1 = V_2$ is detected	0
If the V_1 Sign is minus (but V_1 is not - 0) and the V_2 Sign is plus (or if V_2 = ± 0), then $V_1 < V_2$ is detected	−

Note: In the Compare process the values in Univac code of the characters compared are used as the basis for the comparison of the magnitudes of the characters. Thus the following relationships are detected *

DELETE > = > % > Z > Y—S > / > + > : > β > ∑ > ? > * > $ > R — J >) > | > " > t > @ > ¢ >

> I—A > ; > . > , > r > (> & > ' > 9 > 8 — 0 > — > Δ

* 0 and Δ do not compare except in the next compare process after a Condition Compare (See Condition Compare, page II-72).

OED CYCLE FOR COMPARE
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK → SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 37 → PR value of S → SR	.210	.441	.672
3	U of IRVc → SAR U of SRV → SK	.210	.441	.672
4	Load V_1 into RA	See Page II-229		
**5	V of IRVc → SAR	.210	.441	.672
	Shift V_1 in RA	.042 (n+1)		
6	Load V_2 into RB V of SRV → SK	See Page II-229		
**7	Advance PAK PAK → SAR	.210	.441	.672
	Shift V_2 in RB	.042 (n+1)		
**8	Load IRVn	See Page II-229		
	Compare V_1 and V_2 and set Branch Storage accordingly.	.55		
9	Set OED to 12	.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II -100)	.042		

* Omitted if this IW is already in IRVn.

** Times listed are simultaneous. Larger of the two determines time for this OED Step. ("n" is the number of places V_1 or V_2 is shifted).

OED CYCLE FOR COMPARE
(Program Step)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)
3	V_1 ADDRESS → SAR	.042
	V_1 SHIFT → SK	
4	Load V_1 into RA	See Page II-229
*5	V_2 ADDRESS → SAR	.042
	Shift V_1 in RA	.042 (n+1)
6	Load V_2 into RB	See Page II-229
	V_2 SHIFT → SK	
7	(Advance PAK, PAK → SAR)**	.042 (n+1)
	Shift V_2 in RB	
8	(Load IRVn)**	.55
	Compare V_1 and V_2 and set Branch Storage accordingly	
9	Set OED to 12	.042
12	STEP OUT	.042
	Set OED (See Page II-104)	

Note: The above notation (as V_1 ADDRESS → SAR and V_1 SHIFT → SK), as well as timing listed for setting up addresses in SAR and shifts in SK, does not apply unless SAR and SK are set to an address or shift, respectively, via the plugboard. If data is shifted into these registers, as when the U, V, or W ADDRESS or SHIFT hubs are patched from a Program Step's ADDRESS or SHIFT hubs, the following notation and timing apply:

For Addresses	For Shifts	Timing (milliseconds)		
		Minimum	Average	Maximum
U, V, or W of IRVc → SAR	U, V, or W of SRV → SK	.210	.441	.672

* Times listed are simultaneous. Use shifting time as the time for this OED Step if n (number of places shifted) ≠ 0.

** These operations are performed only if this Program Step is the first in a Transcop sequence. For timing see OED 7-8 of OED cycle for corresponding Instruction Word. Apply * when these operations occur. Use time listed for this OED Step when these operations do not occur.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES.	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Left shift the contents of Register A until the most significant character of the word held in Register A is in Register A's most significant character position; count the number of shifts required to do this; place this count in Register B; and send the (normalized) contents of Register A to Register D.	PR = 35	PROCESS to NORM patching
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Store the final contents of Register D at the location specified by	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

RULES FOR LEFT NORMALIZE INSTRUCTIONS

1. Permissible Sources for V_1:

Any Word Location in GSB, BTB, I/O, IS, or FS Tracks
Any Field Location in GSB, BTB, I/O, IS, or FS Tracks
Notes: if a Field greater than 12 characters is referred to only the lower-order 12 characters of the Field are acquired;

if a Field less than 12 characters is referred to, the Field is loaded into the lower-character positions of the arithmetic register involved (RA,) and the higher-order character positions of the arithmetic register are padded with Space codes.

RA, RB, RC, RD, or IRVc

GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as Sources, a plus (Δ) sign is automatically generated and sent as the lowest-order character of V_1. The contents of GSAR, PAK or CDR are then loaded into the arithmetic register involved (RA,) and the required number of Space codes are padded into the higher-order character positions.

The entire contents of a buffer or track (Z-address) must not be specified.

2. Permissible Destinations for R:

Any Word Location in GSB, BTB, I/O, FS, or IS Tracks
Any Field Location in GSB, BTB, I/O, FS, or IS Tracks
Notes: if a Field greater than 12 characters is specified, the contents of the Source of the result (RD) is loaded into lower-order character positions of the Field Location and Space codes are padded into the higher-order character positions;

if a Field less than 12 characters is specified, only the lower-order characters in the Source of the result are sent to the Field Location.

RA, RB, RC, RD, IRVn, or SRV

GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as a Destination, the sign in the arithmetic register that is the Source of the result and a number of characters equal to the capacity of GSAR, PAK or CDR are sent to these locations. As the highest-order character transmitted is shifted into these locations, the sign is shifted off the right end.

The entire contents of a buffer or track (Z-address) must not be specified.

3. No V_2 is involved in a Left Normalize process. If a V_2 address or shift is programmed it is simply ignored.

RULES FOR LEFT NORMALIZE INSTRUCTIONS (Continued)

4. The V_1 operand is shifted left until a character other than a Zero, Minus Sign, Space code, or Ignore code is detected. The computer remembers the normalizing count (i. e., the number of shifts required to do this.)

5. The normalizing count is stored in RB as follows: Sign position of RB is sent a Δ, character position 1 (or character positions 1 and 2) are sent the actual normalizing count, and the higher-order stages of RB are all Zeros.

6. The normalized V_1 operand with plus (Δ) sign is sent to RD as the result.

7. If an operand containing nothing but Zeros, Space Codes, Ignore Codes or Minus Signs is normalized, a NORMALIZE OVERFLOW error occurs (the instruction is not completed and the NORMALIZE OVERFLOW hub emits.)

8. V_1 and R Shifts, any type, can be programmed for this instruction.

OED CYCLE FOR LEFT NORMALIZE
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 35→PR value of S→SR	.210	.441	.672
3	U of IRVc→SAR U of SRV→SK	.210	.441	.672
4	Load V_1 into RA	See Page II-229		
5	Shift V_1 in RA	.042 (n+1)		
6		.042		
7	Advance PAK PAK→SAR	.210	.441	.672
**8	Load IRVn W of SRV→SK	See Page II-229		
	Normalize V_1 in RA; send a normalized V_1 to RD; and place normalizing count in RB	Timing dependent of size of operand		
**9	W of IRVc→SAR	.210	.441	.672
	Shift R in RD	.042 (n+1)		
10	Store R from RD	See Page II-230		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Conditional SRV Clear Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn

** Times listed are simultaneous. Larger of the two determines time for this OED Step. ("n" is the number of places V_1 or R is shifted).

OED CYCLE FOR LEFT NORMALIZE
(Program Step)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)
3	V_1 ADDRESS → SAR	.042
	V_1 SHIFT → SK	
4	Load V_1 into RA	See Page II-229
5	Shift V_1 in RA	.042 (n+1)
6		.042
7	(Advance PAK, PAK → SAR)*	.042*
8	(Load IRVn)*	.63 (Normalize Overflow) 1.23 (Valid Normalize)
	R SHIFT → SK	
	Normalize V_1 in RA; send a normalized V_1 to RD; and place normalizing count in RB	
**9	R ADDRESS → SAR	.042
	Shift R in RD	.042 (n+1)
10	Store R from RD	See Page II-230
11		.042
12	STEP OUT	.042
	Set OED (See Page II-104)	

Note: The above notation (as V_1 ADDRESS → SAR and V_1 SHIFT → SK), as well as the timing listed for setting up addresses in SAR and shifts in SK, does not apply unless SAR and SK are set to an address or shift, respectively, via the plugboard. If data is shifted into these registers, as when the U, V, or W ADDRESS or SHIFT hubs are patched from a Program Step's ADDRESS or SHIFT hubs, the following notation and timing apply:

For Addresses	For Shifts	Timing (milliseconds)		
		Minimum	Average	Maximum
U, V, or W of IRVc → SAR	U, V, or W of SRV → SK	.210	.441	.672

* These operations are performed only if this Program Step is the first in a Transcop sequence. For timing see OED 7-8 of OED cycle for corresponding Instruction Word. Apply** when these operationsoccur. Use time listed for this OED Step when these operations do not occur.

** Times listed are simultaneous. Use shifting time as the time for this OED Step if n (number of places shifted) ≠ 0.

SUPPRESS LEFT ZEROS
SZ
(29)

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place, in Register A, the contents of the location specified by	U, or	V_1 ADDRESS patching
Shift the contents of Register A in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching
Left shift the contents of Register A until the most significant character of the word held in Register A is located in Register A's most significant character position; keep track of the number of shifts required to do this, and then right shift the contents of Register A the same number of places, padding in Space codes in Register A's higher-order character positions. Transmit the final contents of Register A to Register D	PR = 29	PROCESS TO SLZ patching
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Store the final contents of Register D at the location specified by	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

RULES FOR SUPPRESS LEFT ZERO INSTRUCTIONS

1. Permissible Sources for V_1:

   Any Word Location in GSB, BTB, I/O, IS, or FS Tracks
   Any Field Location in GSB, BTB, I/O, IS, or FS Tracks
   Notes: if a Field greater than 12 characters is referred to only the lower-order 12 characters of the Field are acquired;

   if a Field less than 12 characters is referred to, the Field is loaded into the lower-character positions of the arithmetic register(s) involved (RA,) and the higher-order character positions of the arithmetic register are padded with Space codes.
   RA, RB, RC, RD, or IRVc

   GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as Sources, a plus (△) sign is automatically generated and sent as the lowest-order character of V_1 or V_2. The contents of GSAR, PAK or CDR are then loaded into the arithmetic register(s) involved (RA,) and the required number of Space codes are padded into the higher-order character positions.
   The entire contents of a buffer or track (Z-address) must not be specified.

2. Permissible Destinations for R:

   Any Word Location in GSB, BTB, I/O, FS, or IS Tracks
   Any Field Location in GSB, BTB, I/O, FS, or IS Tracks
   Notes: if a Field greater than 12 characters is specified, the contents of the Source of the result (RD) is loaded into lower-order character positions of the Field Location and Space codes are padded into the higher-order character positions;

   if a Field less than 12 characters is specified, only the lower-order characters in the Source of the result are sent to the Field Location.

   The entire contents of a buffer or track (Z-address) must not be specified.
   RA, RB, RC, RD, IRVn, or SRV

   GSAR, PAK, CDR: If GSAR, PAK, or CDR are specified as a Destination, the sign in the arithmetic register that is the Source of the result and a number of characters equal to the capacity of GSAR, PAK or CDR are sent to these locations. As the highest-order character transmitted is shifted into these locations, the sign is shifted off the right end.
   The entire contents of a buffer or track (Z-address) must not be specified.

3. No V_2 is involved in a Suppress Left Zero instruction. If a V_2 address or shift is programmed, it is simply ignored.

4. The V_1 operand is shifted left until a character other than a Zero, Minus sign, Space code, or Ignore code is detected. It is then shifted back

the same number of places and space codes are padded in. (All six bits are examined: only zero, minus sign, Space code, or Ignore code are suppressed. If an operand contains nothing but Zeros, Space codes, Ignore codes or Minus signs, the suppressed operand will contain all Space codes (Sign included, regardless of sign of original operand).

5. The suppressed operand and the Sign of the original operand (except as shown in 4. above) are sent to RD (from which it is stored) and it is also left in RA at the conclusion of the process.

6. V_1 and R Shifts, any type, can be programmed for this instruction.

OED CYCLE FOR SUPPRESS LEFT ZEROS
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 29→PR value of S→SR	.210	.441	.672
3	U of IRVc→SAR U of SRV→SK	.210	.441	.672
4	Load V_1 into RA	See Page II-229		
5	Shift V_1 in RA	.042 (n+1)		
6		.042		
7	Advance PAK PAK→SAR	.210	.441	.672
**8	Load IRVn W of SRV→SK	See Page II-229		
	Suppress Left Zeros in V_1 and send copy of modified V_1 to RD	.042 + .084 (s+1) where s = the number of zeros suppressed		
**9	W of IRVc→SAR	.210	.441	.672
	Shift R in RD	.042 (n+1)		
10	Store R from RD	See Page II-230		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Conditional SRV Clear Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn
** Times listed are simultaneous. Larger of the two determines time for this OED Step. ("n" is the number of places V_1 or R is shifted).

OED CYCLE FOR SUPPRESS LEFT ZEROS
(Program Step)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)
3	V_1 ADDRESS → SAR	.042
	V_1 SHIFT → SK	
4	Load V_1 into RA	See Page II-229
5	Shift V_1 in RA	.042 (n+1)
6		.042
7	(Advance PAK, PAK → SAR)*	.042*
8	(Load IRVn)*	.042 + .084 (s+1) where s = the number of zeros suppressed.
	R SHIFT → SK	
	Suppress Left Zeros in V_1 and send copy of modified V_1 to RD	
**9	R ADDRESS → SAR	.042
	Shift R in RD	.042 (n+1)
10	Store R from RD	See Page II-230
11		.042
12	STEP OUT	.042
	Set OED (See Page II-104)	

Note: The above notation (as V_1 ADDRESS → SAR and V_1 SHIFT → SK), as well as the timing listed for setting up addresses in SAR and shifts in SK, does not apply unless SAR and SK are set to an address or shift, respectively, via the plugboard. If data is shifted into these registers, as when the U, V, or W ADDRESS or SHIFT hubs are patched from a Program Step's ADDRESS or SHIFT hubs, the following notation and timing apply:

For Addresses	For Shifts	Timing (milliseconds)		
		Minimum	Average	Maximum
U, V, or W of IRVc → SAR	U, V, or W of SRV → SK	.210	.441	.672

* These operations are performed only if this Program Step is the first in a Transcop sequence. For timing see OED 7-8 of OED cycle for corresponding Instruction Word. Apply * when these operations occur. Use time listed for this OED Step when these operations do not occur.

** Times listed are simultaneous. Use shifting time as the time for this OED Step if n (number of places shifted) ≠ 0.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Place a Space code in each of the 120 character postions of the track or buffer specified by the	PR = 33 W,	PROC to CH CL patching R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

RULES FOR CHANNEL CLEAR INSTRUCTIONS

1. Permissible Locations that can be padded in with Space codes:

   GSB - Z
   BTB - Z
   I/O-Z, FS-Z,or IS-Z

2. Note that W or R ADDRESS patching defines the location involved. No V_1 or V_2 are used by this instruction. If addresses or shifts are programmed for these quantities (or a shift for R is programmed) they are ignored.

OED CYCLE FOR CHANNEL CLEAR
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 33→PR value of S→SR	.210	.441	.672
3	Set OED to 7	.042		
7	Advance PAK PAK→SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	W of IRVc→SAR	.210	.441	.672
**10	Store Space Codes	5.166	7.665	10.164
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Clear SRV Set OED (See Page II -100)	.042		

* Omitted if this IW is already in IRVn

** Timing listed for this operation applies to BTB, GSB, I/O, FS, and IS Tracks.

OED CYCLE FOR CHANNEL CLEAR
(Program Step)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
3	Set OED to 7	.042		
7	(Advance PAK, PAK → SAR)*	.042*		
8	(Load IRVn)*	.042*		
9	R ADDRESS → SAR	.042		
**10	Store Space Codes	Minimum 5.166	Average 7.665	Maximum 10.164
11		.042		
12	STEP OUT Set OED (Page II-104)	.042		

Note: The above notation (R ADDRESS) as well as the timing listed for setting up this address in SAR does not apply unless SAR is set to an address via the plugboard. If the R address is shifted into SAR, as when the U, V, or W ADDRESS hubs are patched from this Program Step's R ADDRESS hub, the following notation and timing apply:

	Timing (milliseconds)		
	Minimum	Average	Maximum
U, V, or W of IRVc → SAR	.210	.441	.672

* These operations are performed only if this Program Step is the first in a Transcop sequence. For timing, see OED 7-8 of OED cycle for corresponding Instruction Word. If these operations do not occur, this OED Step takes .042 millisecond.

** Timing listed applies to BTB, GSB, I/O and FS Tracks whenever R ADDRESS patching specifies these locations. It applies also to IS Tracks, when they are referred to via R ADDRESS-to- U, V, or W, ADDRESS patching.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Transfer data (1 up to 120 characters) from one Program Control Storage location to another, via BTB, as follows: Transfer, to the Block Transfer Buffer, lowest-order character first, the contents of the location specified by Load the first character received* into BTB's Word 9, Character S.position; load the next character received into BTB's Word 9, Character 1 position, etc. until entire contents of the Source are stored in BTB in this manner; when less than 120 characters are loaded into BTB, do not alter the higher-order stages of BTB (i.e., those which do not receive data). * If BTB is specified as the Source, the first half of this instruction is omitted.	PR = 23 U, or	PROCESS to BT patching V_1 ADDRESS patching
Transfer data, out of BTB, to the location specified by Send the data to the Destination,** lowest order character first, beginning with BTB's Word 9, Character S position; store the character from BTB's Word 9, Character S position in the lowest-order character position of the Destination; store the character from BTB's Word 9, Character 1 position in the next lowest-order character position of the Destination, etc.; transmit out only the number of characters required to fill the capacity of the Destination. ** If BTB is specified as the Destination, no transfer out of BTB occurs (i.e., the second half of the instruction is omitted.)	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) or Sub-Step(s) in accordance with	S, or	STEP OUT patching

RULES FOR THE BUFFER TRANSFER INSTRUCTIONS

1. The Buffer Transfer instruction is generally used when more than 12 characters are to be transferred from one location to another.

2. Permissible Sources for V_1: See Table II-15, Page II-222.

   Note that the U or V_1 ADDRESS specifies the location of the Source data. No V_2 is involved in this instruction. If a V_2 address is programmed, it is ignored.

   BTB can be a Source for this instruction, but not any particular Word or Field location in BTB. That is, this instruction cannot be used to transfer data from one BTB location to another, or from a Source to any particular BTB location, or from any particular BTB location to a Destination.

3. Permissible Destinations for R: See Table II-16, Page II-223.

4. No shifting is permissible in this instruction. If V_1, V_2, or R shifts are programmed, they are simply ignored.

5. See explanation of Buffer Transfer instruction for details on the manner in which data is transferred into BTB from the various Sources and then transferred out of BTB to the various Destinations. Note that all data transmissions to and from BTB begin with BTB's Word 9, character S position.

OED CYCLE FOR BUFFER TRANSFER
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 23→PR value of S→SR	.210	.441	.672
3	U of IRVc→SAR	.210	.441	.672
4	Source→BTB	See Page II-222		
5		.042		
6		.042		
7	Advance PAK PAK→SAR	.210	.441	.672
8	Load IRVn	See Page II-229		
9	W of IRVc→SAR	.210	.441	.672
10	BTB→Destination	See Page II-223		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Conditional SRV Clear Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn

OED CYCLE FOR BUFFER TRANSFER
(Program Step)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)
3	V_1 ADDRESS → SAR	.042
4	Source → BTB	See Page II-222
5		.042
6		.042
7	(Advance PAK, PAK → SAR)*	.042*
8	(Load IRVn)*	.042*
9	R ADDRESS → SAR	.042
10	BTB → Destination	See Page II-223
11		.042
12	STEP OUT Set OED (See Page II-104)	.042

Note: The above notation (V_1 ADDRESS → SAR and R ADDRESS → SAR) as well as the timing listed for setting up these addresses in SAR does not apply unless SAR is set to an address via the plugboard. If the V_1 or R address is shifted into SAR, as when the U, V, or W ADDRESS hubs are patched from either the V_1 ADDRESS or R ADDRESS hubs, the following notation and timing apply:

	Timing (milliseconds)		
	Minimum	Average	Maximum
U, V, or W of IRVc → SAR	.210	.441	.672

* These operations are performed only if this Program Step is the first in a Transcop sequence. For timing, see OED 7-8 of OED cycle for corresponding Instruction Word. If these operations do not occur, this OED Step takes .042 millisecond.

Table II-15. Valid Sources for Buffer Transfer Instructions.

Locations which are Valid Sources for Buffer Transfer Instructions	Time Required to Obtain Source Data and Place it in BTB (milliseconds)		
	Minimum	Average	Maximum
1. Any Word Location in:			
GSB	.630	.861	1.092
I/O, FS, IS Tracks	.630	3.129	5.628
2. Any Field Location in:	x - the number of character positions that must be scanned to arrive at the beginning of the field; y - the number of characters in the field.		
GSB — Field A	(.042y + .126)	(.042y + .357)	(.042y + .588)
GSB — Field B-V if x is odd:	(.021x + .042y + .105)	(.021x + .042y + .336)	(.021x + .042y + .567)
GSB — Field B-V if x is even:	(.021x + .042y + .084)	(.021x + .042y + .315)	(.021x + .042y + .546)
I/O, FS, IS Tracks*	(.042y + .126)	(.042y + 2.625)	(.042y + 5.124)
3. The entire contents of:			
GSB	5.166	5.397	5.628
I/O, FS, IS Tracks	5.166	7.665	10.164
4. The following Registers:			
Register A	.630	.861	1.092
Register B			
Register C			
Register D			
5. BTB itself (but not on a Word & Field Addressable basis): .042			

* The times listed for Field references to the I/O, FS, and IS Tracks assume that ISP is being read at the time the Field reference is initiated. This is generally the case, since ISP is read at all times except during Field references to BTB or GSB and during the interval of time between the completion of such references and the next reading of the Reference Mark on the High Speed Drum. (The latter then causes ISP to be read again.) Since the intervals during which ISP is not read is completely dependent on the program (i.e., on what sequence of buffer and track Field referencing is programmed) it is impossible to take formal account of these intervals in a timing discussion. It can be noted, however, that the time between completion of the buffer Field reference and the reading of the Reference Mark can vary from practically zero up to a maximum of 5 milliseconds.

Table II-16. Valid Destinations for Buffer Transfer Instructions.

Locations which are Valid Destinations for Buffer Transfer Instruction.		Time Required to Transfer Data out of BTB and store it in Destination. (milliseconds)		
		Minimum	Average	Maximum
1. Any Word Location in:				
	GSB	.630	.861	1.092
	I/O, FS, IS Tracks	.672	3.171	5.670
2. Any Field Location in:		x - the number of character positions that must be scanned at the beginning of the field; y - the number of characters in the field.		
	GSB, Field A	(.042y + .126)	(.042y + 2.625)	(.042y + 5.124)
	GSB, Field B-V if x is odd:	(.021x + .042y + .105)	(.021x + .042y + 2.604)	(.021x + .042y + 5.103)
	GSB, Field B-V if x is even:	(.021x + .042y + .084)	(.021x + .042y + 2.583)	(.021x + .042y + 5.082)
	I/O, FS, IS Tracks	(.042y + .168)	(.042y +.2.667)	(.042y + 5.166)
3. The entire contents of:				
	BTP (BTB Pattern)	5.124		
	I/O, FS, IS Tracks	5.208	7.707	10.206
	GSB, GSP, ISP	5.166	7.665	10.164
4. The following Registers:				
	Register A	.630	.861	1.092
	Register B			
	Register C			
	Register D			
	IRVn			
	SRV			
5. BTB itself (but not on a Word & Field Addressable basis): .042				

* The times listed for Field references to the I/O, FS, and IS Tracks assume that ISP is being read at the time the Field reference is initiated. This is generally the case, since ISP is read at all times except during Field references to BTB or GSB and during the interval of time between the completion of such references and the next reading of the Reference Mark on the High Speed Drum. (The latter then causes ISP to be read again.) Since the intervals during which ISP is not read is completely dependent on the program (i.e., on what sequence of buffer and track Field referencing is programmed) it is impossible to take formal account of these intervals in a timing discussion. It can be noted, however, that the time between completion of the buffer Field reference and the reading of the Reference Mark can vary from practically zero up to a maximum of 5 milliseconds.

SEQUENCE OF EVENTS	DEFINING PROGRAMMED QUANTITIES	
	Instruction Word	Program Step
Transfer data (1 up to 12 characters) from one Program Control Storage location to another, via Register D, as follows:	PR = 13	PROCESS to AT patching
Shift source data into register D from the location specified by	U, or	V_1 ADDRESS patching
If the specified Source contains less than 12 Characters, store the contents* of that location in the lower-order stages of Register D, and pad in Space codes in the higher-order positions;		
If the specified Source contains 12 Characters, load Register D with an exact copy of the contents of that location;**		
If the specified location contains more than 12 Characters, place only the lower-order 12 characters of that location in Register D.		
*If GSAR, PAK, or CDR are specified as Sources, a plus sign (Δ) is automatically generated and sent as the lowest-order character of the transfer. The contents of GSAR, PAK, or CDR are then loaded into Register D and the required number of space codes padded into Register D's higher-order positions.		
** If Register D is specified as the Source, a right end around shift of 12 characters occurs during the first half of this instruction.		
Shift the contents of Register D in accordance with the	contents of the U-section of the Shift Revolver, or	V_1 SHIFT patching

Continued on next page

Continued from previous page

Shift the contents of Register D in accordance with the	contents of the V-section of the Shift Revolver, or	V_2 SHIFT patching
Shift the contents of Register D in accordance with the	contents of the W-section of the Shift Revolver, or	R SHIFT patching
Transfer from 1 up to 119 characters as described below to the location specified by If the specified Destination cannot store 12 characters*, send only the lower-order characters in Register D (i.e., beginning with the character in Register D's sign position, send only the number of characters necessary to fill the capacity of the Destination.) If the specified Destination can hold exactly 12 characters**, store an exact copy of the contents of Register D in that location. If the specified Destination can hold more than 12 characters, store an exact copy of the contents of Register D in the 12 lower-order character positions of that location, and pad Space codes in the higher-order character positions of that location. (Z not permitted in Destination address.) * If GSAR, PAK, or CDR are specified as Destinations, the sign in Register D and a number of characters equal to the capacity of GSAR, PAK, or CDR is sent to these locations. As the highest-order character transmitted is entered into these registers, the sign is shifted off the lower-order end. ** If Register D is specified as the Destination, the right end-around shift of 12 characters does not transmit data to any other Destination.	W, or	R ADDRESS patching
Initiate Sub-Instruction(s) and Sub-Step(s) in accordance with	S, or	STEP OUT patching

RULES FOR ARITHMETIC TRANSFER INSTRUCTIONS

1. The Arithmetic Transfer instruction is used to transfer 12 characters or less from one location to another.

2. Permissible Sources for V_1: **See Table II-17, Page II-229.**

   Note that U or V_1 ADDRESS patching specifies the location of the Source data. No V_2 is involved in this instruction. If a V_2 address is programmed it is ignored. If a V_2 shift is programmed it is performed on the data in RD at V_2 Shift time. A Z-address must not be specified for V_1.

3. Permissible Destinations for R: **See Table II-19, Page II-230.**

   Note: A Z-address must not be specified for R.

4. Three Shifts, any type, can be programmed for this instruction.

5. See explanation of Arithmetic Transfer instruction for details on the manner in which data is transferred from the various Sources to RD and then from RD to the various Destinations.

OED CYCLE FOR ARITHMETIC TRANSFER
(Instruction Word)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)		
		Minimum	Average	Maximum
*0	PAK→SAR	.210	.441	.672
*1	Load IRVn	See Page II-229		
2	Switch IRV's From OP of IRVc: 13→PR value of S→SR	.210	.441	.672
3	U of IRVc→SAR U of SRV→SK	.210	.441	.672
4	Source→RD	See Page II-229		
5	Shift RD (V_1 Shift)	.042 (n+1)		
6	V of SRV→SK	.210	.441	.672
**7	Advance PAK PAK→SAR	.210	.441	.672
	Shift RD (V_2 Shift)	.042 (n+1)		
**8	Load IRVn	See Page II-229		
	W of SRV→SK	.210	.441	.672
**9	W of IRVc→SAR	.210	.441	.672
	Shift RD (R Shift)	.042 (n+1)		
10	RD→Destination	See Page II-230		
11		.042		
12		.042		
13	Initiate Sub-Instruction(s) Conditional SRV Clear Set OED (See Page II-100)	.042		

* Omitted if this IW is already in IRVn.
** Times listed are simultaneous. Larger of the two determines the time required for this OED Step. ("n" is the number of places V_1, V_2, or R is shifted.)

OED CYCLE FOR ARITHMETIC TRANSFER
(Program Step)

OED	PRINCIPAL EVENTS	TIMING (milliseconds)
3	V_1 ADDRESS → SAR	.042
	V_1 SHIFT → SK	
4	Source → RD	See Page II-229
5	Shift RD (V_1 Shift)	.042 (n+1)
6	V_2 SHIFT → SK	.042
7	(Advance PAK, PAK → SAR)*	
	Shift RD (V_2 Shift)	.042 (n+1)
8	(Load IRVn)*	
	R SHIFT → SK	.042
**9	R ADDRESS → SAR	.042
	Shift RD (R Shift)	.042 (n+1)
10	RD → Destination	See Page II-230
11		.042
12	STEP OUT	.042
	Set OED (See Page II-104)	

Note: The above notation (as V_1 ADDRESS → SAR and V_1 SHIFT → SK), as well as the timing listed for setting up addresses in SAR and shifts in SK, does not apply unless SAR and SK are set to an address or shift, respectively, via the plugboard. If data is shifted into these registers, as when the U, V, or W ADDRESS or SHIFT hubs are patched from a Program Step's ADDRESS or SHIFT hubs, the following notation and timing apply:

For Addresses	For Shifts	Timing (milliseconds)		
		Minimum	Average	Maximum
U, V, or W of IRVc → SAR	U, V, or W of SRV → SK	.210	.441	.672

* These operations are performed only if this Program Step is the first in a Transcop sequence. For timing see OED 7-8 of OED cycle for corresponding Instruction Word. Apply** when these operations occur. Use time listed for this OED Step when these operations do not occur.

** Times listed are simultaneous. Use shifting time as the time for this OED Step if n (number of places shifted) ≠ 0.

Table II-17 Timing for the Procurement of Operands

Locations from which Source data, V_1, or V_2 can be obtained in the following instructions: Arithmetic Transfer; Add w or w/o Check; Subtract w or w/o Check; All Multiply Instructions; All Divide Instructions; Mask Transfer; Compare; Left Normalize; Suppress Left Zeros; Substitute U,V, or W	Time Required (milliseconds): to obtain Source data and place it in RD, or; to obtain V_1 and place it in RA (or in DQ and DR, to place V_1 in RA and RC), or; to obtain V_2 and place it in RB.		
	Minimum	Average	Maximum
1. Any Word Location in:			
GSB / BTB	.630	.861	1.092
I/O, FS, IS Tracks	.588	3.087	5.586
2. Any Field Location in:	Note: "x" is the number of character positions that must be scanned to arrive at the beginning of the field.		
GSB or BTB — Field A	.630	.861	1.092
GSB or BTB — Fields B-V, if x is odd:	(.021x + .609)	(.021x + .840)	(.021x + 1.071)
GSB or BTB — if x is even:	(.021x + .588)	(.021x + .819)	(.021x + 1.050)
I/O, FS, IS Tracks	.588	3.087	5.586
3. The following Memory Locations: Register A, Register B, Register C, Register D, IRVc, GSAR, PAK, CDR	.588	.819	1.050

* The times listed for Field references to the I/O, FS, and IS Tracks assume that ISP is being read at the time the Field reference is initiated. This is generally the case, since ISP is read at all times except during Field references to BTB or GSB and during the interval of time between the completion of such references and the next reading of the Reference Mark on the High Speed Drum. (The latter then causes ISP to be read again.) Since the intervals during which ISP is not read is completely dependent on the program (i.e., on what sequence of buffer and track Field referencing is programmed) it is impossible to take formal account of these intervals in a timing discussion. It can be noted, however, that the time between completion of the buffer Field reference and the reading of the Reference Mark can vary from practically zero up to a maximum of 5 milliseconds.

Table II-18 Timing for the Acquisition of Instruction Words

Locations from which Instruction Words can be obtained	Time required to obtain Instruction Word and Place it in IRVn (milliseconds)		
	Minimum	Average	Maximum
1. Any Word Location in:			
GSB / BTB	.630	.861	1.092
I/O, FS, or IS Tracks	.588	3.087	5.586
2. RA, RB, RC, RD, IRVc	.588	.819	1.050

Table II-19. Timing for the Storage of Results

Locations at which data can be stored by the following Instructions: Arithmetic Transfer; Add w or w/o Check; Subtract w or w/o Check; All Multiply Instructions; All Divide Instructions; Mask Transfer; Left Normalize; Suppress Left Zeros; Substitute U, V, or W	Time Required (milliseconds)		
	Minimum	Average	Maximum
1. Any Word Location in:			
GSB	.630	.861	1.092
BTB			
I/O, FS, and IS Tracks	.630	3.129	5.628
2. Any Field Location in:			
BTB or GSB — Fields ≤ 12 Characters	x is the number of character positions that must be scanned to arrive at the beginning of the field. z is the number of characters in the field minus 12		
Field A	.630	.861	1.092
Fields B-V If x is odd:	(.021x + .609)	(.021x + .840)	(.021x + 1.071)
if x is even:	(.021x + .609)	(.021x + .819)	(.021x + 1.050)
BTB or GSB — Fields > 12 Characters			
Field A	(.042z + .630)	(.042z + .840)	(.042z + 1.071)
Fields B-V if x is odd:	(.021x + .042z + .609)	(.021x + .042z + .840)	(.021x + .042z + 1.071)
if x is even:	(.021x + .042z + .609)	(.021x + .042z + .819	(.021x + .042z + 1.050)
I/O, FS, or I/O Tracks — Fields ≤ 12 Characters	.630	3.129	5.628
Fields > 12 Characters	(.042z + .630)	(.042z + 3.129)	(.042z + 5.628)
3. The following Memory Locations:			
Register A Register B Register C ** Register D IRVn GSAR PAK CDR SRV	.588	.819	1.050

* The times listed for Field references to the I/O, FS, and IS Tracks assume that ISP is being read at the time the Field reference is initiated. This is generally the case, since ISP is read at all times except during Field references to BTB or GSB and during the interval of time between the completion of such references and the next reading of the Reference Mark on the High Speed Drum. (The latter then causes ISP to be read again.) Since the intervals during which ISP is not read is completely dependent on the program (i.e., on what sequence of buffer and track Field referencing is programmed) it is impossible to take formal account of these intervals in a timing discussion. It can be noted, however, that the time between completion of the buffer Field reference and the reading of the Reference Mark can vary from practically zero up to a maximum of 5 milliseconds.

** If RC is the source of the result and also is addressed as Destination, time required is .042 milliseconds.

III. CENTRAL COMPUTER OPERATING MEMORY

A. PROGRAM CONTROL STORAGE SYSTEM

The Program Control Storage System is the operating memory of the UFC Model 1 Central Computer. It is composed of the following principal parts:

all those memory locations which are illustrated in Figure III-1, Page III-2 and whose addresses are listed in Table III-1, Page III-3;

a Transfer Bus, along which data is transmitted bit-by-bit (i.e. in serial fashion) from one location to another; and

Transfer Address Control, TAC, a group of circuits which controls all data transmissions in the Central Computer and contains the address registers (Storage Address Register, SAR, and Transfer Address Register, TAR) that specify which pair of locations is involved in each storage reference.

One function of Program Control Storage is to carry out the process specified by those instructions listed below whose basic operation is to transfer data from one location to another:

Arithmetic Transfer:

Source ⟶ RD
then
RD ⟶ Destination

Buffer Transfer:

Source ⟶ BTB
then
BTB ⟶ Destination

Channel Clear:

Space Code Padder ⟶ Destination

Load GSAR:

IRVc (7 characters) ⟶ GSAR

Load Shift:

IRVc ⟶ SRV

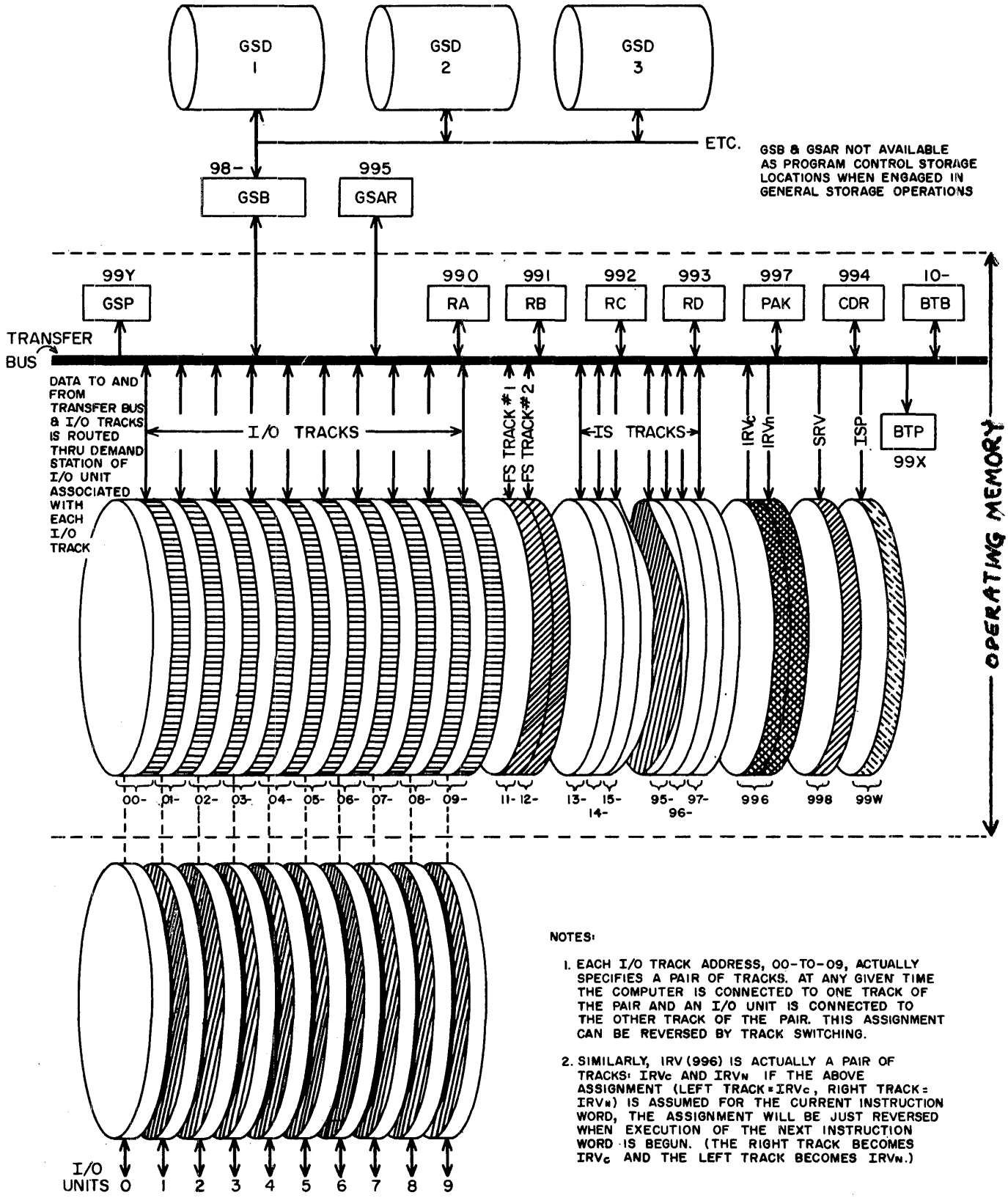
Another function of Program Control Storage is the procurement of operands

Load V_1

Source ⟶ RA
or
Source ⟶ RA and RC

Load V_2

Source ⟶ RB



NOTES:

1. EACH I/O TRACK ADDRESS, 00-TO-09, ACTUALLY SPECIFIES A PAIR OF TRACKS. AT ANY GIVEN TIME THE COMPUTER IS CONNECTED TO ONE TRACK OF THE PAIR AND AN I/O UNIT IS CONNECTED TO THE OTHER TRACK OF THE PAIR. THIS ASSIGNMENT CAN BE REVERSED BY TRACK SWITCHING.

2. SIMILARLY, IRV (996) IS ACTUALLY A PAIR OF TRACKS: IRV_C AND IRV_N IF THE ABOVE ASSIGNMENT (LEFT TRACK = IRV_C, RIGHT TRACK = IRV_N) IS ASSUMED FOR THE CURRENT INSTRUCTION WORD, THE ASSIGNMENT WILL BE JUST REVERSED WHEN EXECUTION OF THE NEXT INSTRUCTION WORD IS BEGUN. (THE RIGHT TRACK BECOMES IRV_C AND THE LEFT TRACK BECOMES IRV_N.)

Figure III-1.

Block Diagram of UFC Model 1 Operating Memory, General Storage System, and I/O Buffer Storage.

Table III-1. Program Control Storage Addresses

PROGRAM CONTROL STORAGE LOCATIONS WHICH ARE WORD, FIELD, AND BLOCKETTE ADDRESSABLE*

LOCATION	ABBREVIATION	PROGRAM CONTROL STORAGE ADDRESSES		
		Addresses of Word Locations (12 characters)	Addresses of Field Locations (Number of characters specified by Field Selection Pattern)	Addresses of Blockette Locations (120 characters)
Input/Output Tracks (High Speed Drum)	I/O Tracks	000-009 010-019 020-029 --- 090-099	00A-00V (not I & O) 01A-01V " " 02A-02V " " --- 09A-09V " "	00Z 01Z 02Z --- 09Z
Block Transfer Buffer	BTB	100-109	10A-10V (not I & O)	10Z
Factor Storage Tracks (High Speed Drum)	FS Tracks	110-119 120-129	11A-11V (not I & O) 12A-12V " "	11Z 12Z
Intermediate Storage Tracks (High Speed Drum)	IS Tracks	130-139 140-149 150-159 --- 970-979	13A-13V (not I & O) 14A-14V " " 15A-15V " " --- 97A-97V " "	13Z 14Z 15Z --- 97Z
General Storage Buffer	GSB	980-989	98A-98V (not I & O)	98Z

*Each of these locations has a 120-character capacity and can store 10 computer words or up to twenty fields.

PROGRAM CONTROL STORAGE LOCATIONS WHICH HAVE ONLY ONE ADDRESS

LOCATION	ABBREVIATION	ADDRESS	CAPACITY
Register A	RA	990	12 characters
Register B	RB	991	
Register C	RC	992	
Register D	RD	993	
Code Distributor Register	CDR	994	1 character
General Storage Address Register	GSAR	995	7 digits
Instruction Revolver (High Speed Drum)	IRV	996	12 characters
Program Address Counter	PAK	997	3 digits
Shift Revolver (High Speed Drum)	SRV	998	12 characters
High Speed Drum Pattern	ISP	99W	120 characters (only parity bit of each character is used)
Block Transfer Buffer Pattern	BTP	99X	120 bits (120 characters are sent to these locations, but only the parity bits of the characters transmitted are stored).
General Storage Buffer Pattern	GSP	99Y	

and the storage of results:

Store R

RD ⟶ Destination
or
RC ⟶ Destination

during the execution of the instructions listed on Page II-11.

A third function of Program Control Storage is the acquisition of Instruction Words:

Load IRVn

Source ⟶ IRVn

As noted in Section II, this function is performed only during the execution of internally-stored programs and during the first Program Step in a Transcop sequence.

Program Control Storage also carries out other "housekeeping-type" data transmissions:

PAK ⟶ SAR	to set up address of next Instruction Word in SAR
OP of IRVc ⟶ PR and SR U of IRVc ⟶ SAR V of IRVc ⟶ SAR W of IRVc ⟶ SAR	to obtain various sections of current Instruction Word, as required.
U of SRV ⟶ SK V of SRV ⟶ SK W of SRV ⟶ SK	to obtain various portions of Shift Word in SRV, as required.
U of IRVc ⟶ RC ⟶ Destination U of IRVc ⟶ PAK V of IRVc ⟶ PAK W of IRVc ⟶ PAK	one or more of these transmissions occur when an Unconditional Jump or Test Demand In Instruction Word is executed or a jump condition is detected in the following instructions Jump on Negative Jump on Plus Jump on Zero Channel Search Probe Test Incoming Control etc.

Only the functions of Program Control Storage during the data transmissions listed in Table III-2, Page III-5 will be discussed in this section. These transmissions are discussed because each is a programmed

Table III-2. Programmed Data Transmissions

DATA TRANSMISSIONS	PROGRAMMING INVOLVED			
	U, or V_1 ADR Patching	V, or V_2 ADR Patching	W, or R ADR Patching	Computer Program's Control of PAK
Buffer Transfer Instruction				
Source → BTB	✓			
and				
BTB → Destination			✓	
Arithmetic Transfer Instruction				
Source → RD	✓			
and				
RD → Destination			✓	
Procurement of Operands				
Load V_1	✓			
Load V_2		✓		
Storage of Results				
Store R			✓	
U of IRVc → RC → Destination		✓		
Acquisition of Next Instruction				
Load IRVn				✓

data transmission and because each has requirements for its execution which must be met by the computer program. For example, in the Buffer Transfer Instruction, only certain Sources and Destinations can be used. As illustrated in Table III-2, the Source in a Buffer Transfer Instruction is specified by U, or V_1 ADDRESS Patching, and the Destination is specified by W, or R ADDRESS Patching. As illustrated in Tables II-15 and II-16, Pages II-222 and II-223, respectively, U and W (or V_1 ADDRESS and R ADDRESS Patching) must specify valid Sources and Destinations. If they do not, the instruction cannot be carried out. One purpose of this section is, therefore, to re-emphasize that only valid Sources and Destinations must be specified for each type of programmed data transmission. Another purpose is to explain the address structure used in Program Control Storage so that U, V, and W are properly coded and storage references to the appropriate memory locations can be carried out. A third purpose of this section is to explain the mechanics of each of the above type of data transmissions in terms of each Source or Destination that can be involved, so that the operating memory of the Central Computer will be efficiently used. Some locations store or supply data faster than others, some have specialized functions, etc.; each of the locations in Program Control Storage, therefore, warrants special study.

To accomplish these objectives the following subjects are discussed.

Program Control Storage Address Structure
- Word Address
- Field Address
  - Field Selection Pattern
  - Determination of the Length of a Field
- Blockette Address
- Word and Field Assignments
- Translation of the Storage Address Register, SAR

Program Control Storage Locations
- Permanent vs. Temporary Memory Locations
- Mechanics of Data Transmissions
  - High Speed Drum
  - BTB and GSB
  - Registers

Buffer Transfer Instruction

Arithmetic Transfer Instruction

Load V_1 and Load V_2

Load IRVn

Store R

B. PROGRAM CONTROL STORAGE ADDRESS STRUCTURE

A Program Control Storage Address is a 3-character quantity that specifies a particular location in the operating memory of the Central Computer.

When data is obtained from storage, the address of the location from which the data is to be obtained is placed in the Storage Address Register, SAR, and PCT sets the Transfer Address Register, TAR, to the address of the intermediate storage to which the data is to be sent. (In the Buffer Transfer and Arithmetic Transfer Instructions, and in the procurement of operands, the intermediate storage is automatically determined by the process specified by the instruction. In the acquisition of Instruction Words, TAR is always set to 996, IRVn.) If the Source specified by the address in SAR, is interpretable and valid, TAC can carry out the data transmission. If the address in SAR is not interpretable or is not valid, the computer hangs up.

When the result of a process is stored (and when the U-section of a Jump Instruction Word is placed in the W-section of the V address) the address of the location to receive data is placed in SAR, and PCT automatically sets TAR to the address of the intermediate storage (RC or RD) from which the result is to be obtained. Here again the intermediate storage is determined by the process being carried out. If the Destination specified by the address in SAR is interpretable and valid for the process specified by the instruction, TAC can carry out the data transmission. If the address in SAR is not interpretable or is not valid, the computer hangs up.

Each of the Program Control Storage Addresses listed in Table III-1 can be interpreted when it is placed in SAR provided it is a valid address for the particular type of storage reference in which it is used. Details on interpretable and process-valid addresses are given below under Translation of SAR.

Table III-3, Page III-8 is an explanation of the format of Program Control Storage Addresses. The following paragraphs clarify the meaning of Word, Field, and Blockette addresses referred to in that table.

1. Word Address

A word is a collection of 12 contiguous characters. Those Program Control Locations which are Word, Field, and Blockette Addressable each can store 10 words, and a particular 12-character area (Word 0 thru Word 9) is assigned for each word. (The significance of each Word Location, as well as that of stored data in general, is discussed under Word and Field Assignments below.) A Word Address thus always refers to a particular collection of 12 characters in a Word, Field and Blockette addressable location.

2. Field Address

A field is a collection of from 1 up to 119 contiguous characters. In the Program Control Storage locations which are

Table III - 3. Explanation of format of Program Control Storage Addresses

Program Control Storage Address	
X X	X
if the two leftmost characters of a Program Control Storage Address are interpreted as numbers in the range 00-98, they specify a particular type of location that is Word, Field and Blockette addressable: 00-09 I/O Tracks 10 BTB 11-12 FS Tracks 13-97 IS Tracks 98 GSB —— and ——	if the rightmost character is: 0-9, the address is a Word Address and specifies a Word Location; if A-V (excluding I and O), the address is a Field Address and specifies a Field Location: and if Z, the address is a Blockette Address and specifies the entire contents of the track or buffer.
if the two leftmost characters are 99, they specify a memory location that has but one address: ——	the particular location is determined by the rightmost character, as illustrated in Table III-1, Page III-3; and the address is called a Register, Revolver, or Pattern Address.

Word, Field, and Blockette addressable, up to twenty fields (A-V, excluding I and O) can be specified. Fields are not assigned any fixed area (i.e., specific number of character positions) in these locations. For example, in one part of a program Field A in BTB could be 8 characters, and in another part of the program it could be 26 or 50 characters. That is, field lengths are programmable. Each field is defined by a specially-coded 120-character quantity called a Field Selection Pattern that is stored in a special-purpose memory (ISP, BTP, or GSP) associated with each of the Word, Field, and Blockette Addressable locations. A Field Address always contains a letter, A-V (excluding I and O) as its lowest-order character; and always refers to the collection of characters, defined at that instant by the Field Selection Pattern, as Field A, B, C, etc.

a. Field Selection Pattern

A Field Selection Pattern is a collection of 120 characters so coded that their parity bits can be used by the computer to define the length of each field in Word, Field, and Blockette Addressable memory locations. Each Field Selection Pattern is composed of a series of B's and D's, or 5's and 6's, or any two characters whose parity bits differ. The B's and D's, or 5's and 6's, or whatever pair of characters is used in programming, are arranged so that the characters with "0" as their parity bits define the start and body of each field, and the characters with "1" as their parity bit define the last character (i.e. end) of each field. A typical Field Selection Pattern is illustrated in Figure III-2a, Page III-10.

b. Determination of the Length of a Field

The following 120-character memory locations on the High Speed Drum are Word, Field, and Blockette addressable:

Input/Output Tracks

Factor Storage Tracks

Intermediate Storage Tracks

The fields for all of these tracks at any given instant is commonly defined by the Field Selection Pattern which is stored on the Field Selection Pattern Track (ISP, See Figure III-2b) at that time. That is, Field A on I/O Track 00 is the same length as Field A on Factor Storage Track 11 or Intermediate Storage Track 35; and the number of characters in Field A is determined by the Field Selection Pattern Stored on ISP.

The fields in BTB and GSB are similarly determined, but each has a separate Field Selection Pattern. The fields for BTB at any given instant* are defined by the Field Selection Pattern stored at that time in the Block Transfer Pattern, BTP (See Figure III-2c). The fields for

* The Block Transfer Buffer is not Field Addressable in Buffer Transfer instructions.

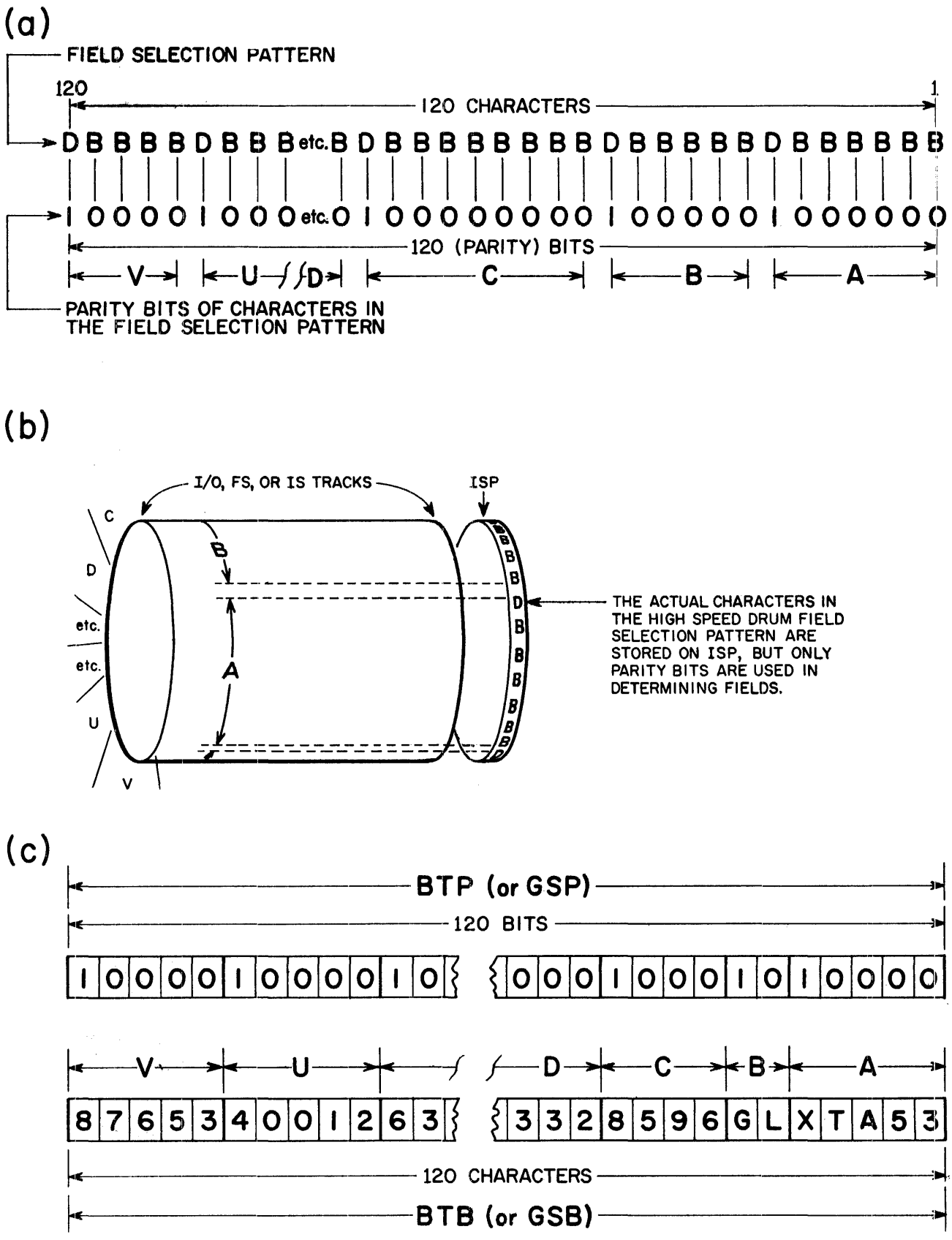


Figure III-2.
Field Selection Patterns

GSB at any given instant* are defined by the Field Selection Pattern stored at that time in the General Storage Pattern, GSP (See Figure III-2c).

By sending various Field Selection Patterns, using the Buffer Transfer instruction, to the special memories (ISP, BTP, and GSP) associated with each Field addressable location, different field lengths can be defined at different times in a program.

Field Selection Patterns can be generated by the program or can be stored in any 120-character location (as on the IS or FS Tracks on the High Speed Drum, or in a 10 word Unit Record Area on a General Storage Drum). Field Selection Patterns are functional, however, only when they are placed in ISP, BTP, and GSP and a Field-Address storage reference is made to the respective 120-character location with which each is associated.

3. Blockette Address

A Blockette is a collection of exactly 120 contiguous characters. It is the entire contents of a buffer or a data-track on the High Speed Drum. A Blockette Address always contains Z as its lowest-order character, and always refers to the entire contents (120 characters) of a buffer or track.

4. Word and Field Address Assignments

a. Word Address Assignments

In order to permit data to be stored in the Central Computer memory in the same order or significance which that information has on a printed sheet, the following Word Address assignments have been made for BTB, GSB, and the I/O, FS, IS Tracks on the High Speed Drum: (See Figure III-3, Page III-12.)

The Word 0, Character 11 position (or position 120) is the most-significant character position on any track or in any buffer;

The Word 0, Character 10 position (or position 119) is the next most-significant character position, etc.; and

The Word 9, Character S position (or position 1) is the least-significant character position on any track or in any buffer.

---

* The General Storage Buffer is not Field Addressable during General Storage operations; i.e., when GSB is transferring data to and from the General Storage Drums.

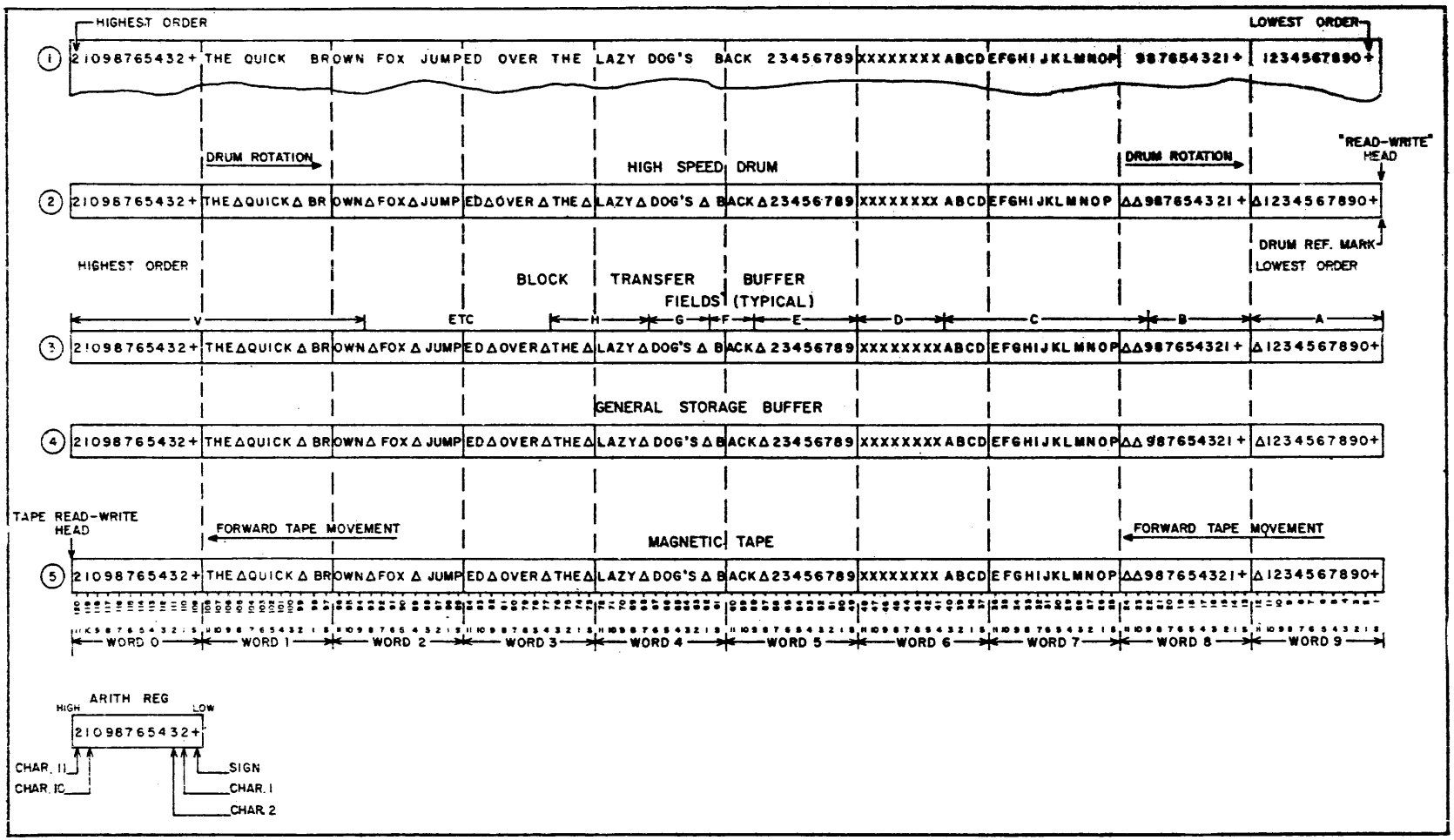


Figure III-3. Word and Field Address Assignments.

To understand how these Word Address assignments facilitate programming, particularly the packing and unpacking of input-output data, consider the 120-character message on the top line of Figure III-3. This message, like all printed or handwritten data, is ordered. For reading purposes, the left-most character "2" is the most significant character; the right-most character "+" is the least significant character. If this message were reproduced on an input medium (as magnetic tape, see line 5 of Figure III-3), or if it were manually inserted into the computer, the normal procedure would be to process each character in succession from the high-order character ("2") to the low-order character ("+"). By common usage, the first 12 characters processed would be regarded as Word 0, the next 12 characters processed would be regarded as Word 1, etc. The tenth set of 12 characters would be regarded as Word 9. With the Word Address assignments noted above, Word 0 of an input message (i.e., the first word) can be placed in a Word 0 Address (most significant location); Word 1 of an input message can be placed in a Word 1 Address (next-most significant location) etc., without special programming, and without destroying, or in any way altering the relationship of any adjacent characters in the message.

Thus, if the magnetic tape data shown on line 5 of Figure III-3 is read into the computer (See line 2 of Figure III-3)

The most significant character of the message, "2", would be stored in the most significant character position (120) on the tape unit's I/O track;

The least significant character of the message, "+", would be stored in the least significant character position (1) on that track; and

The intermediate characters would be stored in relation to their (reading) significance.

Since the order or sequence of the characters stored in the computer is exactly that of the original input message, Field Addressing can be directly employed to unpack data which is less than 12 characters in length or which overlaps two or more Word Address locations (as Fields G and C, respectively, line 3 of Figure III-3), and the data referred to will be in proper sequence.

Since the Block Transfer Buffer and the General Storage Buffer have the same Word Address assignments as tracks on the High Speed Drum, when the 120 character input message is transferred to either of these buffers via a Buffer Transfer instruction it will also appear the same in these buffers as it would on a printed page (See lines 3 and 4 of Figure III-3).

To generate the message shown on the top line of Figure III-3 as output data, the computer would, in general, be programmed to store the sequence of characters shown on line 2 of Figure III-3 on the appropriate I/O Track.

b. Field Address Assignments

The Field Address assignments for BTB, GSB, and the I/O, FS, IS Tracks on the High Speed Drum are also shown on Figure III-3. Field A begins with the lowest-order character position (Word 9, Character S or position 1) on a track or in a buffer, and extends through the higher-order characters as defined by the Field Selection Pattern employed. Field B begins with the next higher-order character after the last character in Field A, etc. The last field defined by the Field Selection Pattern contains the highest-order characters. Since scanning must be done to obtain higher-order data in BTB and GSB when a Field Address is employed, it is obviously advantageous to use Word Addressing wherever possible when these locations' higher-order data is desired.

5. Translation of Storage Address Register, SAR

The Storage Address Register, SAR, is a three-stage, non-addressable register whose prime function* is to store

> the address of the location(s) from which the operand(s) of an instruction is (are) obtained; or
>
> the address of the location to which the result of an instruction is to be sent; or
>
> the address of the location from which an Instruction Word is to be obtained;

during Program Control Storage References.

SAR receives the addresses it holds during the internally-stored program when the U, V, and W-sections of IRVc (i.e., the current Instruction Word) are shifted into SAR as part of the execution of an instruction. SAR receives the address it holds during plugboard-defined programs when hubs in the Plugboard Addressing System are enabled (via patchcord wiring from the address hubs of Program Steps), and set SAR to the address that corresponds to the Plugboard Addressing System hub enabled. (See Page II-47.)

Table III-4 outlines the manner in which addresses in SAR are translated. Note that the two highest order stages of SAR actually store only the excess-three bits of the characters they receive, and that these stages can only be interpreted if they hold the excess-three bits of the digits 0-9. The lower order stage stores the (6-bit) Univac code of each character it receives. However, only certain values are permitted (i.e., can be interpreted) in this stage also.

If only the addresses listed in Table III-1 are ever used in coding U, V, and W, an interpretable address will always be shifted into SAR. Similarly, the Plugboard Addressing System will always set SAR to an interpretable address. (See cautions on Page II-46.) If alternates are used, Table III-4 should be consulted.

* Other uses of SAR are discussed in Sections II and V.

C. PROGRAM CONTROL STORAGE LOCATIONS

1. General Information

The FS and IS Tracks on the High Speed Drum are the principal permanent memory locations in the Central Computer. The other locations are all "scratch pad" type memories; i.e., they are temporary memories whose prime function is something other than storage. When these locations carry out their principal functions, their previous contents are destroyed or modified. Use of these locations as temporary memories, however, is an invaluable tool in Model 1 programming provided a program is designed so that the prime function of these locations will not interfere with their storage functions.

The General Storage System, although not part of the operating memory of the Central Computer, can be conveniently employed as permanent storage for large internally-defined programs. Use of a portion of General Storage in this manner is particularly advantageous for "straight-line" programming wherein a minimum of loops is used in the program's logic, and, considerably therefore, more memory is required to store the program. Instruction Words in such programs can be loaded into GSB on a time-shared basis, and used out of GSB, directly; or they can be first transferred to BTB, or the IS Tracks and used from these locations, as convenient.

The address (or addresses) of each Program Control Storage Location is (are) given in Table III-1. Table III-5, Pages III-18 through III-20 outlines these additional facts about each location:

its principal function
its peculiarities as a storage location if it has any; and
its availability in internally-stored vs. plugboard-defined programs.

The comments on the availability of each location are general remarks that apply only when the location is a valid Source or Destination for the particular storage reference in which it is used.

2. Mechanics of Data Transmissions

a. Data Transmissions to and from the High Speed Drum

The High Speed Drum is an enclosed, rotating cylinder which is coated with a magnetizable surface. As illustrated in Figure III-4 , the High Speed Drum contains the following Program Control Storage Locations:

Table III-4. Translation of Storage Address Register

In the two higher-order stages of SAR, only the excess-three bits of characters are stored.			In the lowest-order stage, six-bit characters are stored.		
Only digits are interpretable	These bits are translated to recognize each digit	The following characters will uniquely* translate to a digit	These characters are interpretable	These bits are translated to recognize each character	The following will uniquely* translate to the required character
0	(00--)	0(digit),;,), or +	0	(0000--)	only 0
1	(0-00)	1, A, J, or /	1	(000-00)	only 1
2	(-101)	2, B, K, or S	2	(00-101)	only 2
3	(-110)	3, C, L, or T	3	(00-110)	only 3
4	(-111)	4, D, M, or U	4	(00-111)	only 4
5	(-000)	5, E, N, or V	5	(00-000)	only 5
6	(-001)	6, F, 0(letter), or W	6	(00-001)	only 6
7	(-010)	7, G, P, or X	7	(00-010)	only 7
8	(1-11)	8, H, Q, or Y	8	(001-11)	only 8
9	(11--)	9, I, R, or Z	9	(0011--)	only 9
			A	(010100)	only A
			B	(010101)	only B
			|		only
			|	etc.	correct
			|		character
			H	(011011)	only H
			J	(100100)	only J
			|		only
			|	etc.	correct
			|		character
			N	(101000)	only N
			P	(101010)	only P
			|		only
			|	etc.	correct
			|		character
			V	(111000)	only V
			W	(111-01)	only W
			X	(111-10)	only X
			Y	(111-11)	only Y
			Z	(1111--)	only Z

* If characters which are not uniquely translatable are placed in SAR, the data transmitted will be garbled. (See next page for list of characters which cannot be uniquely translated).

continued from previous page

Characters which are not uniquely translated in SAR		
Two higher-order stages		Lowest-order stage
i	'	i
Δ	&	Δ
-	(	-
r	#	'
,	¢	&
.	@	(
t	$	%
"	*	=
|	?	delete
Σ	%	
β	=	
:	delete	

Uninterpretable SAR Addresses

00-98 with the right most character equal to any of the following characters:

I	,	)	#
0 (letter)	.	Σ	$
W	;	β	¢
X	t	:	*
Y	"	+	@
r	|	/	?

99 with anything but

0 (digit)	3	6	W
1	4	7	X
2	5	8	Y

as the rightmost character

If an uninterpretable address is formed in SAR the computer will hang up when it attempts to use the uninterpretable address.

Table III-5. Comparison of Program Control Storage Locations

S = Source
D = Destination
W = Word
F = Field
B = Blockette

PROGRAM CONTROL STORAGE LOCATIONS	PRINCIPAL FUNCTION & GENERAL CHARACTERISTICS	AVAILABLE TO	
		Internally-stored Programs as:	Plugboard-Defined Programs as:
I/O Tracks	20 (i.e., ten pairs of) tracks on the High Speed Drum that are used as buffers for input-output data. Each I/O Track Address to which an I/O Unit has been assigned specified a pair of tracks: at any given time, one track of this pair is connected to the computer and the other track of the pair is connected to the I/O Unit. The computer and I/O Unit can thus time-share Storage references on their respective tracks. Track switching (via the I/O Unit's Demand Station) is programmed to reverse the track assignments. Each I/O Track Address will refer Program Control Storage to the track to which the computer is connected at that time.	S or D; on a W, F, and B basis	S or D; on W, F, and B basis. However, only the Word, Field or Blockette part of the address can be specified by enabling hubs in the Plugboard Addressing System. The actual I/O Track must by specified by placing the associated I/O Unit "on demand". Since only one I/O Unit can be "on demand" at a time only one I/O Track can be referred to in this way per Program Step. An I/O Track can also be made available in a plugboard-defined program if the address hub(s) of a Program Step in Transcop and Breakpoint sequences is (are) patched to the U, V, or W ADDRESS hubs in the Plugboard Addressing System, and the corresponding U, V, or W-section(s) of the Instruction Word in IRVc at that time refers to an I/O Track.
	If an installation employs less than ten I/O Units, a trivial equipment change can be made so that the I/O Tracks not associated with an I/O Unit will function as IS Tracks. In this case only one track of the pair is available.	S or D; on a W, F, and B basis	When I/O Tracks function as IS Tracks they are available to plugboard-defined programs only in Transcop and Breakpoint sequences as noted above for normal use of I/O Tracks.
Block Transfer Buffer	120 characters of magnetic core memory. Used principally as an intermediate storage in the Buffer Transfer instruction. Also can be used as a rapid-access Program Control Storage location.	S and D; on a W, F, and B, basis except in the Buffer Transfer instruction.	S or D; on a W, F, and B basis, except in the Buffer Transfer instruction.
FS Tracks	2 separate tracks on the High Speed Drum that are generally used for permanent storage of constants.	S or D; on a W, F, and B basis	S or D; on a W, F, and B basis

continued on next page

Table III-5 Comparison of Program Control Storage Locations

continued from previous page

IS Tracks	85 tracks on the High Speed Drum that are used to store internally defined programs (Instruction Words, constants and intermediate results)	S or D; on a W, F, and B basis	IS Tracks not directly available via the Plugboard Addressing System, but can be used in Transcop and Breakpoint sequences, as described above for I/O Tracks.
General Storage Buffer	120 characters of magnetic core memory. Primarily used as the buffer for data transmissions to and from the General Storage Drums; also, available however, when not engaged in General Storage operations, as a rapid-access Program Control Storage location.	S or D; on a W, F, and B basis Note: If a Program Control Storage Reference involving either GSB or GSAR is attempted while these registers are engaged in General Storage operations (i.e., are "locked out"), Program Control will wait until the previously initiated General Storage Operation is completed before continuing the program.	S or D; on W, F, and B basis
Register A Register B Register C Register D	Each of these registers can be used as a 12-character storage location. Since at least one, and in many cases all, of these registers are used in executing the process of an instruction, these registers should only employed as temporary storage.	S or D	S or D
PAK	A three-digit register that will store and deliver only the digit (or excess-three portion) of characters.* Every use of PAK as a D must be carefully considered, since PAK is used in sequencing the internally-stored program.	S or D	S or D
GSAR	A seven-digit register that will store and deliver only the digit (or excess-three) portions of characters.* Primarily used to hold General Storage Addresses during General Storage operations	S or D See "Note" under GSB above	S or D
CDR	A one-character register, usually used as storage for special control characters (derived from input data), and is used to achieve program variance in plugboard-defined programs.	S or D	S or D
Instruction Revolver	Although it has but a single address, IRV is actually two 12 character revolvers on the High Speed Drum: IRVc (the one containing the current instruction) and IRVn (the one into which the next instruction is stored.) The principal function of IRV is carried out in the acquisition and execution of Instruction Words. NOTE: In Transcop and Breakpoint sequences, IRVc contains an Instruction Word. In Transcop sequences, IRVc contains the Transcop Instruction Word. In Breakpoint sequences, IRVc contains the Instruction Word whose OP contained the Breakpoint Special Character. When the V_1, V_2 or IR ADDRESS hubs of Program Steps in such sequences are patched to the U, V, or W ADDRESS hubs the appropriate address-section of the Instruction Word in IRVc is automatically referred to and supplies the required address.	IRVc as S IRVn as D	IRVc as S IRVn as D

continued on next page

* Although PAK and GSAR store only the excess-three bits of a character when they are addressed as a source the zone bits 00 and the appropriate parity bit are added.

Table III-5. Comparison of Program Control Storage Locations

continued from previous page

Shift Revolver	A rapid-access storage location on the High Speed Drum. Used only to store Shift Words. Note: SRV can also be loaded by the Load Shift Instruction Word. If SRV contains a Shift Word and the V_1, V_2, or R SHIFT hubs of Program Steps in Transfer Control and Breakpoint sequences are patched to the U, V, or W SHIFT hubs, the appropriate U, V, or W section of the Shift Word in SRV is automatically referred to and supplies the amount and type of shift to be performed.	only as D	only as D
High Speed Drum Pattern	A special track (120-character location) on the High Speed Drum. Used only to store the (common) Field Selection Pattern for the I/O, FS, and IS Tracks. Although 120 characters are stored in this location, only the parity bits are ever used.	only as D	only as D
Block Transfer Pattern General Storage Pattern	These are 120-bit registers which receive only the parity bits of the 120 characters transferred to them in Buffer Transfer Instructions. Used to hold the Field Selection Patterns for BTB and GSB, respectively.	only as D	only as D

I/O Tracks
FS Tracks
IS Tracks
IRV
SRV, and
ISP

The High Speed Drum is also used as the basic clock (or source of timing pulses) for the system.

Data is transferred to and from the drum's surface via magnetic heads that are mounted in the housing of the drum. Each magnetic head communicates with (i.e., it can read from or record on) the narrow band, called a track, that passes beneath it as the drum revolves.

As illustrated in Table III-5 the various tracks (and their associated magnetic heads) have different functions. They all have the following common characteristics, however, with regard to the manner in which data is transmitted to and from them:

Each can function to transmit data only when its address is placed in SAR and Program Control initiates a storage reference;

Each manipulates data lowest-order character first (i.e., each records data and reads data, least significant character first; to see this note direction of revolution of drums.)

In each case data is recorded or written on a track by sending a series of electrical signals representing "1's" or "0's" to the track's associated magnetic head. The head, upon receipt of such signals, then magnetizes minute areas around the periphery of the track, as each area passes beneath it. Each minute area on the drum surface is polarized in one direction or the other depending on the direction of current in the head at the time that area was passing beneath the head. Polarization in one direction causes a binary "0" to be stored; polarization in the opposite direction causes a binary "1" to be stored. Computer characters are thus stored on a track by recording a magnetic copy of the binary "0's" and "1's" that appear in the Univac code (plus parity bit) for those characters. Seven angular positions, or bit positions, are needed for each character stored. In line with the fact that the lowest order character of a group of characters is recorded first, the lowest order bit of each character is also recorded first. (See Figure III-5, Page III-24)

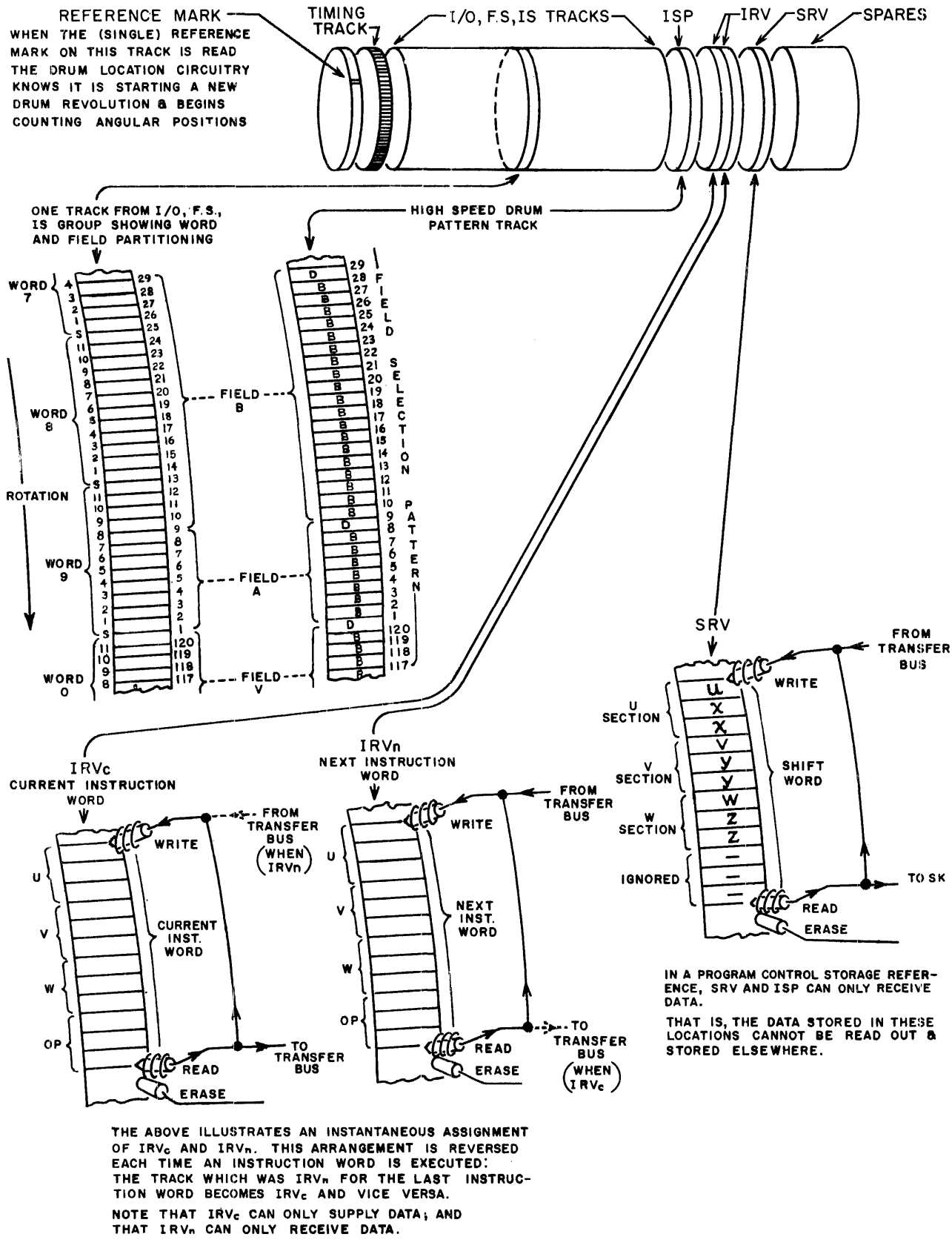


Figure III-4. Program Control Storage Locations on the High Speed Drum

Writing destroys the previously stored information.

Similarly, in each case, data is read from a track when the magnetic field of each minute area induces a signal in the track's magnetic head as that area passes beneath the head. The direction of the current each area produces depends on the polarization given that area when it was recorded. Reading thus develops a series of electrical signals representing "1's" or "0's" which is identical to the series used in recording the data. Seven such signals are produced for each character read. Since the lowest-order character of a group of characters is recorded first, that area is the first to pass under the head in a reading operation. Stored data is thus read lowest-order character first. Reading does not alter the areas read.

Special note should be made of the organization of data on each type of track, as illustrated in Figure III-4. Also note that IRVn, ISP, and SRV can only be used by the programmer to store data; i.e., a programmer cannot transfer data sent to those locations out of those locations. Figure III-5 shows a typical write operation that employs a Word Address to store 12-characters in a specific Word Location on a particular track. Figure III-6 illustrates a typical reading operation using a Field Address. In both of these examples, if the lowest-order character of the address were a Z, the entire track would have been involved, rather than just a Word or Field.

b. Data Transmissions to and from BTB and GSB*

Both BTB and GSB are 120-character magnetic core registers that have an associated locating system.

Each magnetic core is a donut-shaped, magnetizeable substance which can be polarized in one direction (to store a binary "1") or another (to store a binary "0"; the "1" or "0" state of a magnetic core is comparable to the "1" and "0" polarizations of minute areas on a magnetic drum.)

In BTB and GSB, 7 magnetic cores are used as a unit to store one character; there is, therefore, a magnetic core associated with each bit of a character's Univac code (Parity, Zone, and Excess-Three Bits). Each of the 120 stages, or character positions, of BTB or GSB thus contains 7 cores.

---

* This discussion assumes that GSB and BTB can operate as Word, Field and Blockette addressable locations during the storage reference. See Footnotes, Pages III-9 and 11.

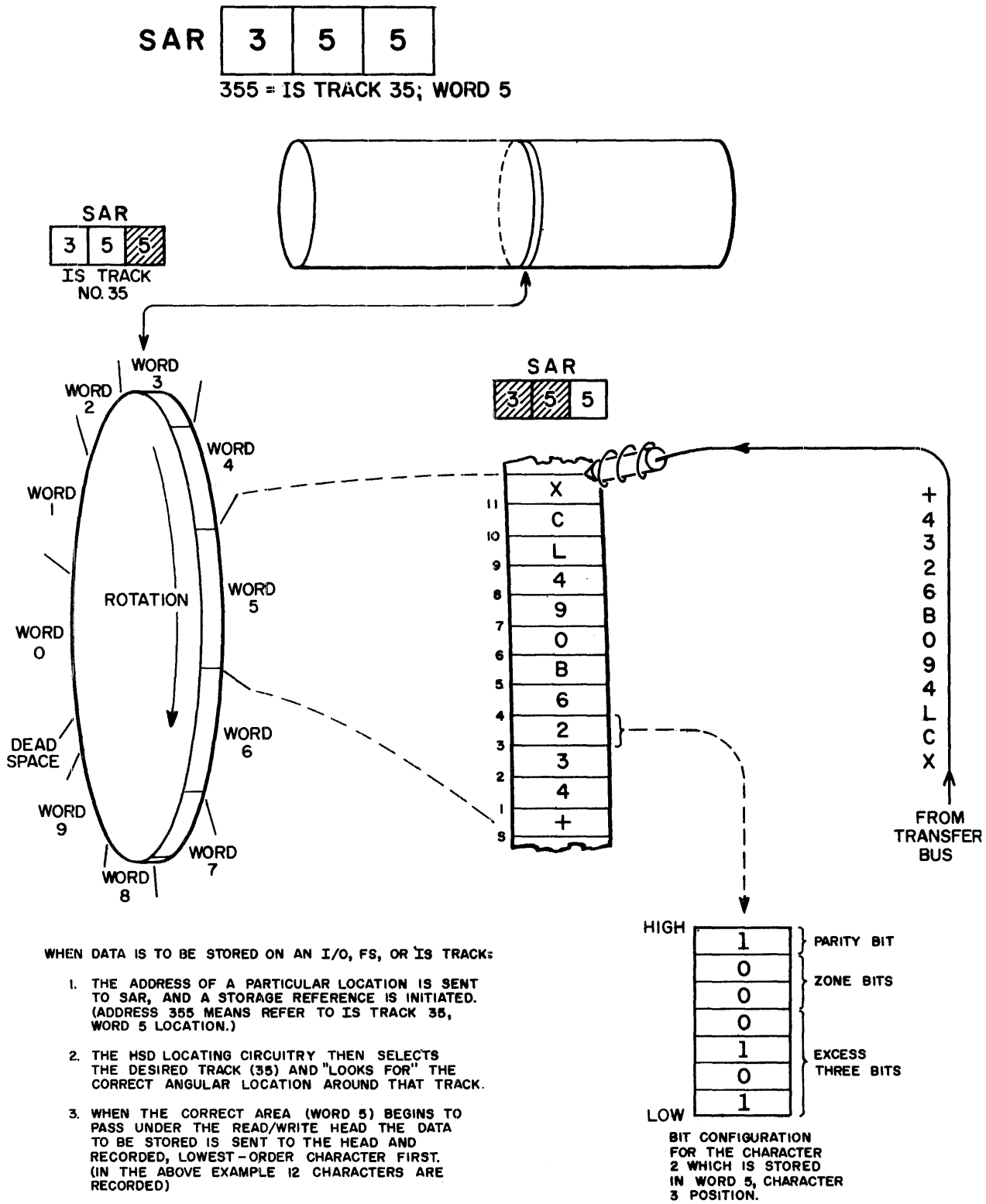
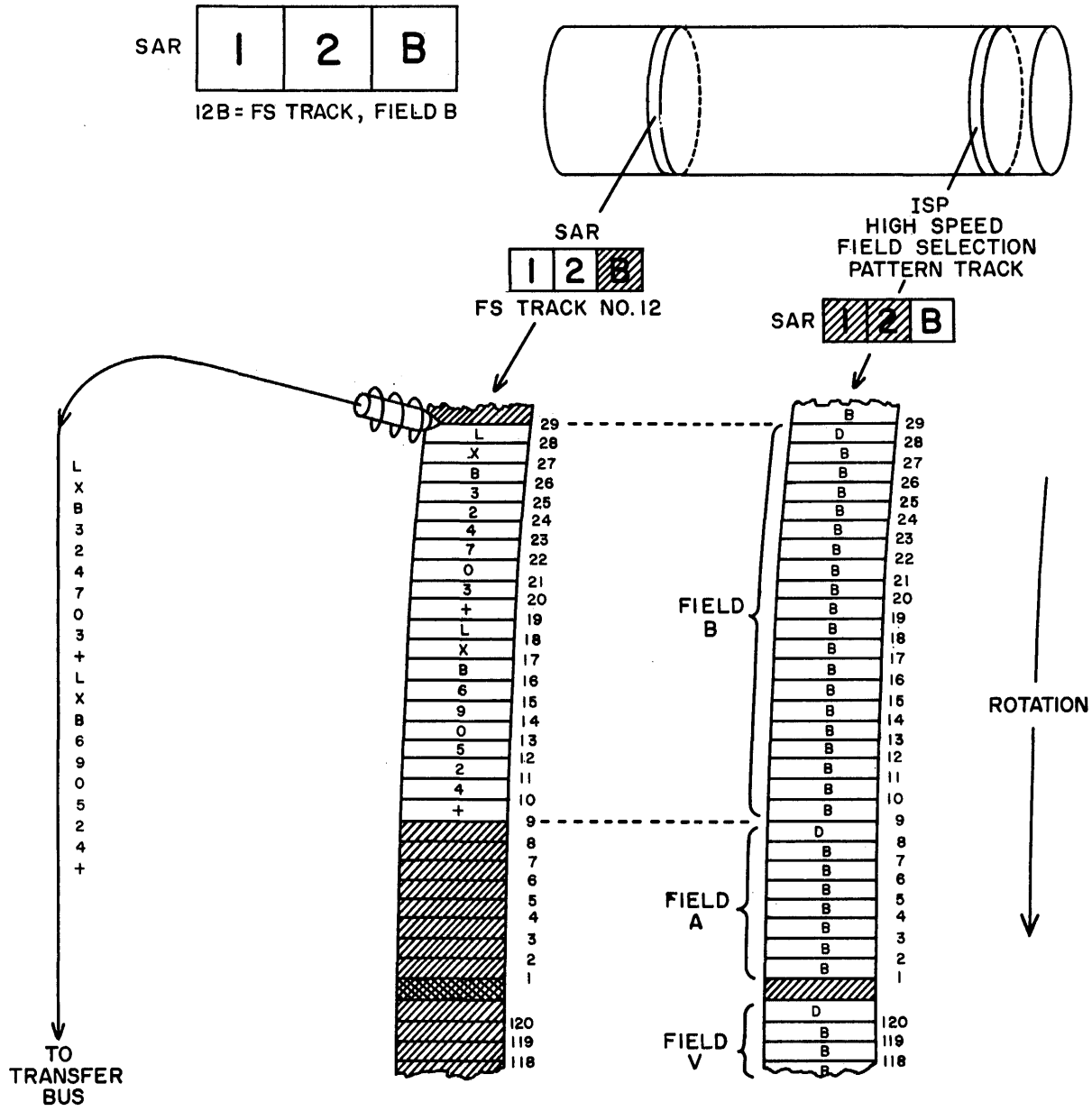


Figure III-5. Typical STORE R operation involving Word Location on High Speed Drum.



WHEN DATA IS TO BE READ FROM AN I/O, FS, OR IS TRACK:

1. THE ADDRESS OF THE PARTICULAR LOCATION IS SENT TO SAR AND A STORAGE REFERENCE IS INITIATED. (ADDRESS 12B ABOVE MEANS REFER TO FS TRACK 12, FIELD B LOCATION)
2. THE HSD LOCATION CIRCUITRY THEN SELECTS THE DESIRED TRACK (12) AND "LOOKS FOR" THE CORRECT ANGULAR LOCATION (FIELD B) ON THAT TRACK. SINCE A FIELD IS INVOLVED THE FIELD SELECTION PATTERN ON ISP (RATHER THAN A WORD COUNT) DETERMINES WHERE THE SPECIFIED AREA ON THE TRACK BEGINS.
3. WHEN THE CORRECT AREA (FIELD B) BEGINS TO PASS UNDER THE READ/WRITE HEAD, THE DATA STORED THERE IS READ (LOWEST-ORDER CHARACTER FIRST). THE NUMBER OF CHARACTERS READ (20 ABOVE) DEPENDS ON THE LENGTH OF THE FIELD AS DEFINED BY THE FIELD SELECTION PATTERN.

Figure III-6. Typical Read Operation (as Source→BTB) involving a Field Location on the High Speed Drum

In writing operations the 7 bits of each character stored are inserted simultaneously (i.e., in parallel) in a character position. Similarly, in reading operations all 7 bits of a character position are read out simultaneously. (A serializer-staticizer register associated with each buffer takes each 7-bit character received in reading operations and supplies it bit-by-bit to the Transfer Bus; in writing operations, this one-character register receives each bit from the Transfer Bus and "collects" or staticizes the bits received, so that all bits of the character to be stored can be stored in the buffer simultaneously.

Data is stored in and obtained from BTB and GSB lowest-order character first.

When a Word Location in BTB or GSB is referred to, the locating system automatically switches to the lowest-order character position of the appropriate Word Location. The reading or writing operation can thus begin immediately. The first character of the data transmission is stored in (or received from) the lowest-order character position. The locating system then advances or switches to the next-highest order character position, and the next character is stored in (or received from) that character position, etc., until 12 characters are stored (or read out). When the transmission U of RC→V takes place and a Word Location in BTB or GSB is specified by V, the locating circuitry switches to the character 4 position of the specified Word Location and writing begins at that point rather than the lowest-order character position of the Word Location.

When a Field Location in BTB or GSB is referred to, the locating system must first "find" the Field involved. It begins its search for the correct Field Location with BTB or GSB's Word 9, Character S character position, and steps along, a character position at a time, until it finds the beginning of the correct Field. The Pattern associated with BTB (i.e. BTP) or GSB (GSP) defines each Field's length, and is referred to at each character position to find the correct Field. When the lowest-order character position of the correct Field is located, reading or writing can begin. The first character of the data transmission is stored in (or received from) the lowest-order character position of the Field. The locating system then advances or switches to the next-highest order character position, and the next character is stored in (or received from) that character position, etc. The number of characters stored (or read out) is determined by the length of the Field, as defined by the Field Selection Pattern. BTP or GSP, depending on whether BTB or GSB respectively are involved in the data transmission, are thus also referred to during the actual reading or writing operation to determine when the operation terminates. Storing data in BTB and GSB destroys the previous contents of the character positions that receive data; reading data from these locations does not alter the data they contain.

c. Data Transmissions to and from Registers

When data is transmitted to and from the following registers: RA, RB, RC, RD, CDR, GSAR, and PAK,

the operation is a shift operation. Data is entered into these registers' high-order end and shifted through the registers a character at a time. The number of characters each register receives depends on the capacity of the register. Data is read out of these registers' low-order end, and is shifted out a character at a time. In general, the number of characters sent to the Destination varies with the capacity of the Destination. When new information is sent the registers, their previous contents are destroyed. When data is read out of a register, it is actually shifted end around and is not changed in any way.

D. BUFFER TRANSFER INSTRUCTION

The Buffer Transfer Instruction is generally used when more than 12 characters (and up to 120 characters) are to be transferred from one location to another. (See Page II-218.)

In general, a Buffer Transfer has two parts:

data from a Source Address is loaded into BTB; and then

data from BTB is sent to a Destination Address.

The rules for Buffer Transfers are given below.

1. Permissible Sources: See Table II-15, Page II-222.

2. Permissible Destinations: See Table II-16, Page II-223.

3. If any memory locations other than those allowed by Rules 1 and 2 are specified in a Buffer Transfer or if an illegal address (see Table III-4) is employed, the program will be interrupted (stopped) when the illegal Source or Destination reference is attempted.

4. Source and Destination Addresses need not specify the same number of characters.

5. The Source data is always stored in BTB* during the first half of a Buffer Transfer, beginning with BTB's Word 9, Character S position; and the Destination data is always transmitted out of BTB** during the second half of a Buffer Transfer, lowest-order character first beginning with BTB's Word 9, Character S position, as follows:

---

* If BTB is specified as the Source, no transmission into BTB takes place; i.e., the first half of the Buffer Transfer Instruction is omitted.

** If BTB is specified as the Destination, no transmission out of BTB takes place; i.e., the second half of the Buffer Transfer Instruction is omitted.

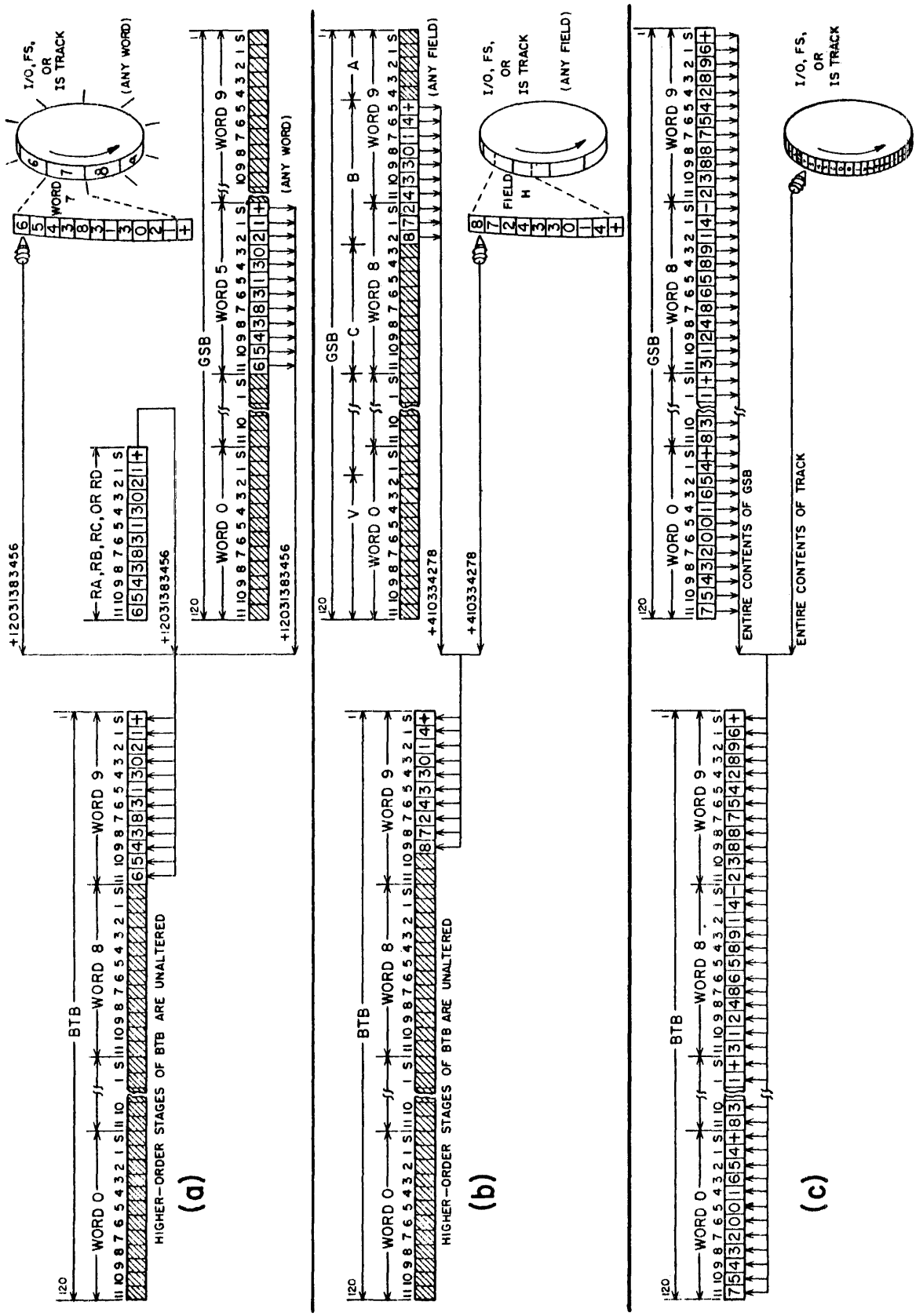


Figure III-7. Firsthalf of Buffer Transfer Instruction.

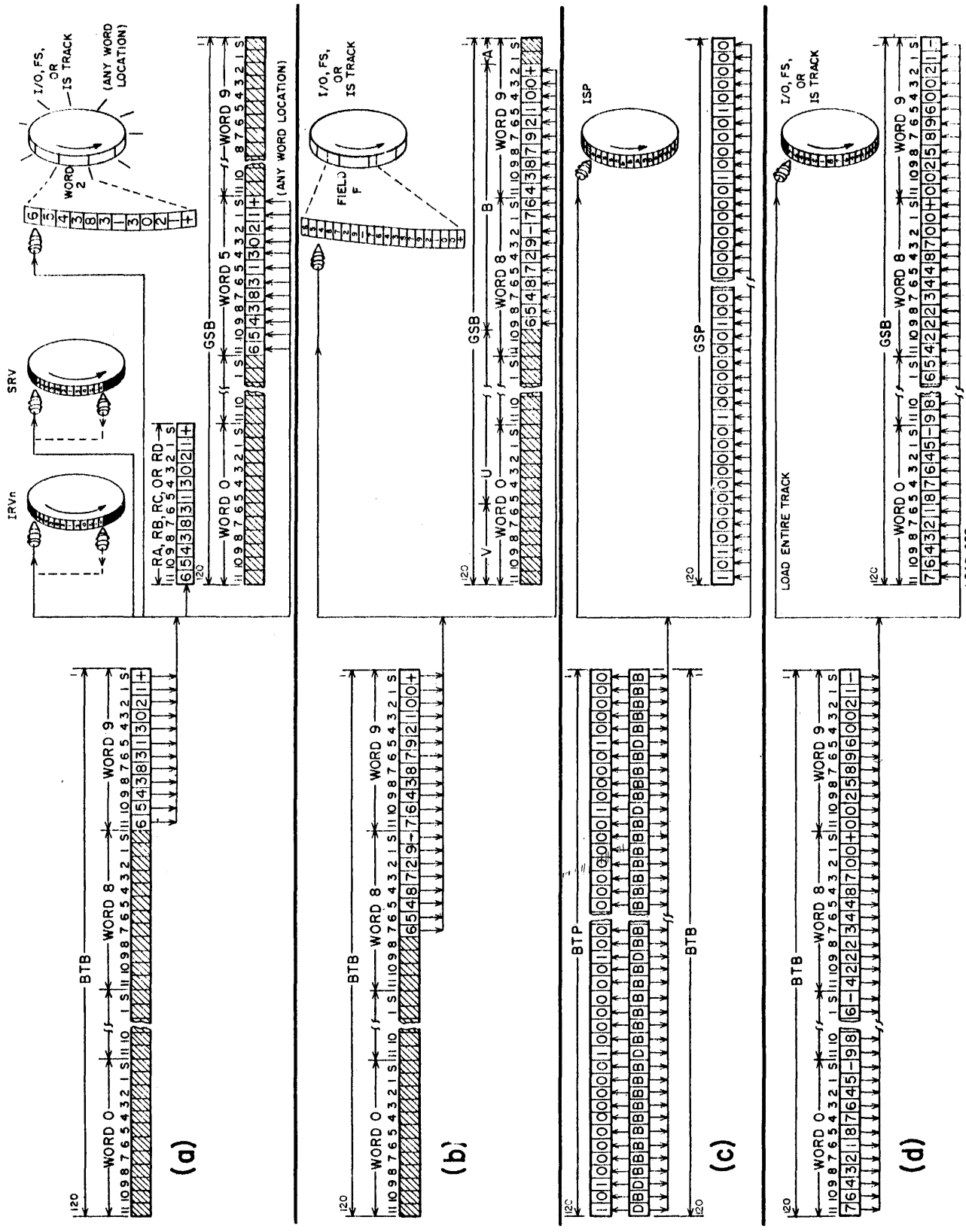


Figure III-8. Second half of Buffer Transfer Instruction.

During first half of a Buffer Transfer: (See Figure III-7, Page III- 28)

5a. If the number of characters specified by the Source Address is less than 120 characters, a copy of the source data is stored in BTB, beginning with BTB's Word 9, Character S position; and the higher-order stages of BTB (i.e. those which do not receive source data) are left unaltered. (See (a) and (b) of Figure III-7.)

5b. If the number of characters specified by the Source Address is 120 characters, an exact copy of the contents of the Source is stored in BTB beginning with BTB's Word 9, Character S position. (See (c) of Figure III-7.

During second half of Buffer Transfer: (See Figure III-8, Page III- 29)

5c. If the number of characters specified by the Destination Address is less than the number of characters specified by the Source Address, not all the Source Data is sent to the Destination. That is, only those character positions in BTB beginning with BTB's Word 9 Character S position, and extending up to the capacity of the Destination, supply data. Similarly, if the number of characters specified by the Destination Address is greater than the number of characters specified by the Source Address, more stages of BTB transmit data out to the Destination than received data from the Source. The Destination's lower-order character positions thus receive a copy of the source data, and the higher-order character positions of the Destination receive a copy of the data held in correspondingly significant BTB character positions. (See (a) and (b) of Figure III-8 to verify that the capacity of the Destination determines the number of characters transmitted during second half of Buffer Transfer Instruction.)

5d. If 120 characters are specified by the Destination Address, the entire contents of BTB are transferred to the Destination. (See I/O, FS, or IS Track and GSB in (d) of Figure III-8.) If BTP or GSP are specified as Destinations, only the parity bits of the 120 characters stored in BTB are sent to those locations (See (c) in Figure III-8). When GSP is sent a Field Selection Pattern during a Buffer Transfer instruction, GSB is not involved. (GSB data is therefore left unaltered.)

Notes:

In Buffer Transfers, BTB is not Word and Field addressable. That is, Program Control Storage ignores the lower-order character in BTB's address, if a BTB address is detected in SAR. This character can. therefore. be anything. Any BTB hub (Word, Field, or Z) can be patched to refer to BTB in a Buffer Transfer instruction; one must always be used, however, if BTB is to be involved as Source or Destination.

If BTB is used as a Source, but not as a Destination also: no transmission into BTB occurs during first half of a Buffer Transfer; however, during the second half of the Buffer Transfer, the data supplied by BTB is always transferred out of BTB beginning with Word 9, Character S position. The number of characters BTB supplies as a Source depends on the number of characters specified by the Destination Address. Although the data transmitted is always sent from BTB beginning with the lowest-order character position of BTB (Word 9, Character S), the Destination address can specify any particular Word or Field in the case of Word and Field addressable Destinations. as Source or Destination.

If BTB is used as a Destination, but not as the Source also: during the first half of a Buffer Transfer, source data is always entered into BTB lowest-order character first. The first (or lowest-order) source character is stored in BTB's Word 9, Character S position, the next-lowest-order character is stored in BTB's Word 9, Character 1 position, etc., until all the Source data is stored in BTB; but no transmission out of BTB occurs during the second half of the Buffer Transfer Instruction. In short, although the Source data can be derived from any Word or Field of a Word and Field addressable Source, it cannot be stored in any particular field in BTB. The number of characters sent to BTB depends on the number of characters specified by the Source Address.

If BTB is used as both the Source and the Destination in a Buffer Transfer Instruction, nothing happens in BTB. Both halves of the Buffer Transfer are suppressed. An "end of operation" pulse is immediately produced, and the program continues. A Buffer Transfer Instruction cannot be used, therefore, to transfer data from one BTB location to another.

The timing involved in Buffer Transfer Instructions is given on Pages II-222 and II-223.

E. ARITHMETIC TRANSFER INSTRUCTION

The Arithmetic Transfer Instruction is used to transfer 12 characters or less from one location to another. (Since BTB is Word and Field Addressable in this instruction, Arithmetic Transfers can be used, among other things, to transmit data from one BTB location to another.)

In general, an Arithmetic Transfer has two parts:

> data from a Source Address is loaded into Register D; and then
>
> data from Register D is sent to a Destination Address.

The first half of an Arithmetic Transfer (Source $\longrightarrow$ RD) is identical to Load V_1 and Load V_2 except that in the latter transmissions, RA (or RA and RC) and RB, respectively, are involved rather than RD. The second half of the Arithmetic Transfer is similar to Store R except that in the latter transmission, RC or RD can be involved, depending on the instruction. Accordingly the same circuitry is used in the computer for the first half of an AT instruction as for Load V_1 and Load V_2; and the same circuitry is used for the second half of an AT instruction as for Store R. (For this reason also Figures III-9 through III-14 are general diagrams applying to all arithmetic-transfer-type data transmissions.)

The rules for Arithmetic Transfers are given below. These rules also apply, with minor variations, to the Load V_1, Load V_2, Load IRVn and Store R data transmissions discussed later in this section. (When reference is made to these rules in subsequent paragraphs, the necessary changes for the wording of the particular Arithmetic Transfer Rule referred to will be noted.)

1. Permissible Sources: See Table II-17, Page 229.

2. Permissible Destinations: See Table II-19, Page 230.

3. If any memory location other than those allowed by Rules 1 and 2 are specified or if an illegal address (See Table III-4) is used, the program is interrupted (stopped) when the illegal source or destination reference is attempted.

4. Source and Destination Addresses need not specify the same number of characters.

5. Data from the Source is always shifted into RD from its high-order end, and 12 characters are always involved; data is always shifted out of RD's low-order end to the Destination and the number of characters received by the Destination depends on the nature and capacity of the Destination:

During first half of an Arithmetic Transfer:

5a. If the Source Address specifies less than 12 characters, there are two types of Sources that can be involved:

> I/O, FS, or IS Track Fields, or BTB or GSB Fields < 12 characters: in this case, a copy of the contents of the Source Address is loaded into the lower-order stages of Register D, and the higher-order stages of Register D (i.e., those not receiving source data) are filled with space codes. (See Figures III-9 and III-11.)
>
> GSAR, PAK, or CDR: in this case, a plus sign (△) is automatically sent to Register D as the first (or low-order) character of the transfer; a copy of the contents of GSAR, PAK, or CDR is then sent to the other lower-order character positions of Register D; and the higher-order character positions of Register D are filled with space codes. (See Figure III-13.)

5b. If the Source Address specifies 12 characters, Register D is loaded with an exact copy of the contents of the Source Address.

> I/O, FS, IS Track Word*Locations: See Figure III-9.
> BTB or GSB Word*Locations: See Figure III-11.
> IRVc, RA, RB, RC, and RD: See Figure III-13.

5c. If the Source Address specifies more than 12 characters, only the 12 lower-order characters of the Source Address are loaded into Register D.

> I/O, FS, or IS Track Fields > 12 characters: See Figure III-9.
> BTB or GSB Fields > 12 characters: See Figure III-11.

* 12-character fields fill arithmetic register same as Word Location data.

During the second half of an Arithmetic Transfer:

5d. If the Destination Address specifies less than 12 characters, there are two types of Destinations that can be involved:

I/O, FS, IS Track Fields, or BTB, or GSB Fields < 12 characters: in this case, data is transferred out of Register D until the number of characters specified by the Destination Address have been transmitted; the transfer operation then terminates.* (See Figures III-10 and III-12.)

GSAR, PAK, or CDR: in this case, the Sign character in Register D and a number of characters equal to the capacity of GSAR, PAK, or CDR are transferred out of Register D, and shifted into these registers from the high-order end. When the last character transmitted* (i.e., the high-order character) is shifted into these registers, the Sign character is shifted off the low-order end. (See Figure III-14.)

5e. If the Destination Address specifies exactly 12 characters a copy of the entire contents of Register D is stored at the Destination Address.

I/O, FS, IS Track Word** Locations: See Figure III-10.
BTB or GSB Word Locations: See Figure III-12.
IRVn, SRV, RA, RB, RC, and RD: See Figure III-14.

5f. If the Destination Address specifies more than 12 characters, a copy of the contents of Register D is stored in the 12 low-order character positions of the Destination and the higher-order character positions in the Destination are filled with space codes.

I/O, FS, or IS Track Fields > 12 characters: See Figure III-10.
BTB or GSB Fields > 12 characters: See Figure III-12.

NOTES: If Register D is used as a Source, but not also as a Destination, an end-around Shift of 12 characters takes place in Register D during the first half of the Arithmetic Transfer. The number of characters transmitted out of Register D during the second half of the Arithmetic Transfer is given under Rules 5d, 5e, and 5f above.

6. Blockette Addresses are not permitted for either Source or Destination. If a Source is referred to via a Blockette Address, the computer hangs up.

7. If a Destination is referred to via a Blockette Address, the transfer never terminates.

8. Do not use Blockette Addresses for any arithmetic transfer type storage references.

---

* The data transmitted is shifted out the low-order end of RD, sent to the Destination, and also re-entered into RD at RD's high-order end. Although the transfer of data to a Destination terminates when the capacity of the Destination is filled, the recirculation (or end-around shift) of RD does not. Twelve characters are always shifted out of RD and RD is left in the same status as at the beginning of the transfer out.

** 12-character fields receive 12 characters the same as Word Locations.

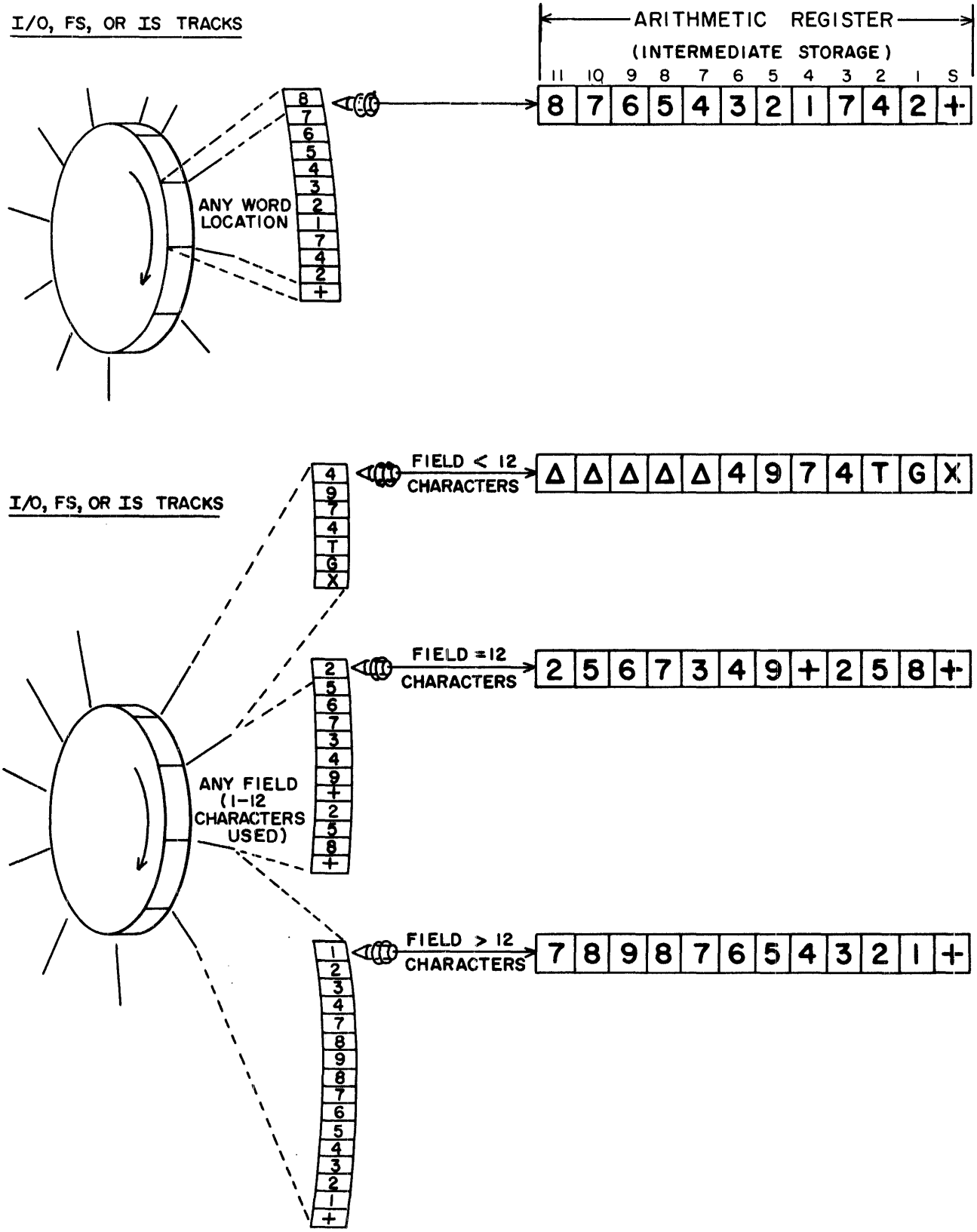


Figure III-9. I/O, FS, or IS Tracks as Sources for the Source→RD, Load V_1, and Load V_2 Data Transmissions.

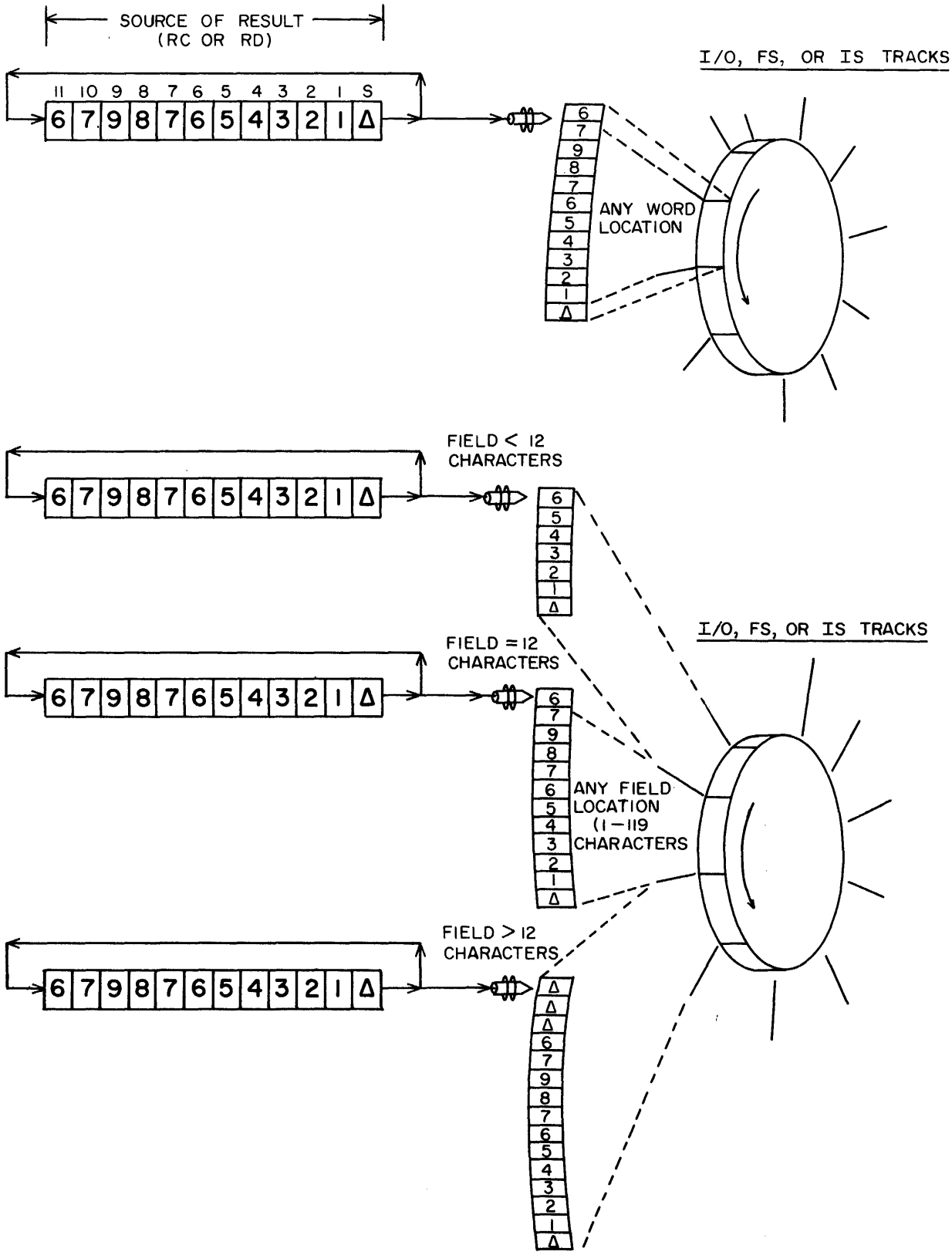


Figure III-10. I/O, FS, or IS Tracks as Destinations for the RD→Destination and Store R Data Transmissions.

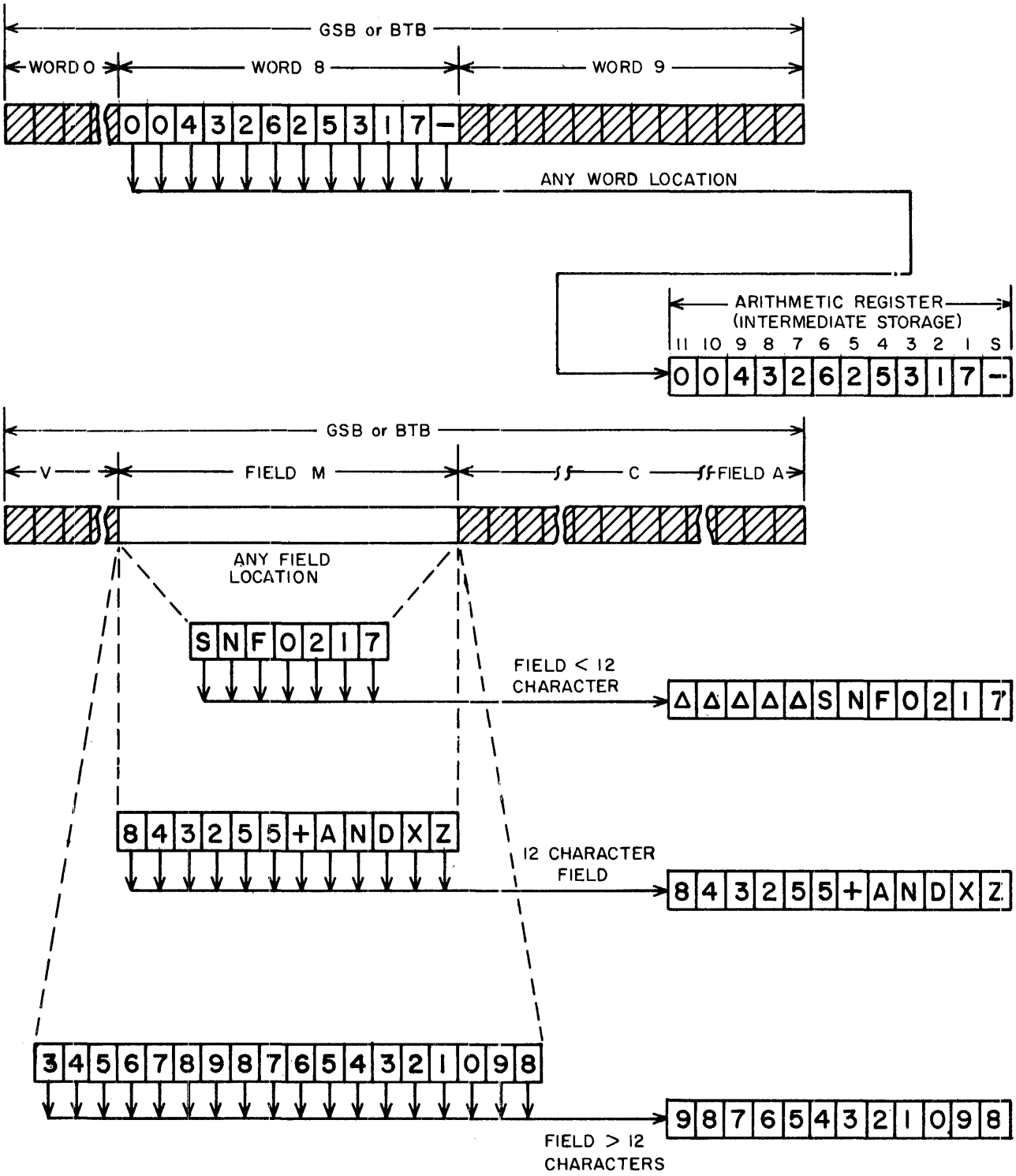


Figure III-11. GSB or BTB as Sources for the Source⟶RD, Load V_1, and Load V_2 Data Transmissions

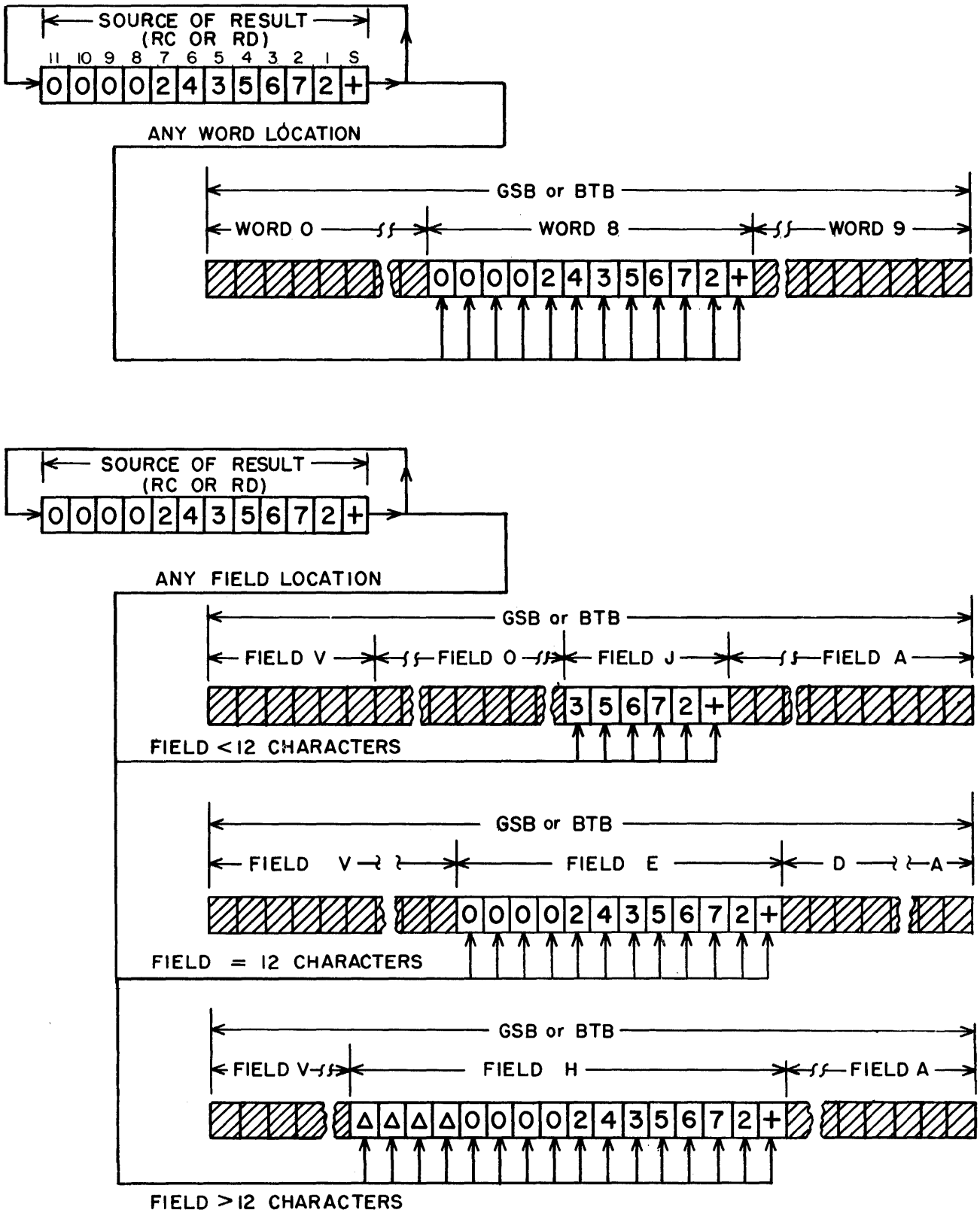


Figure III-12. GSB and BTB as Destinations for the RD→Destination and Store R Data Transmissions.

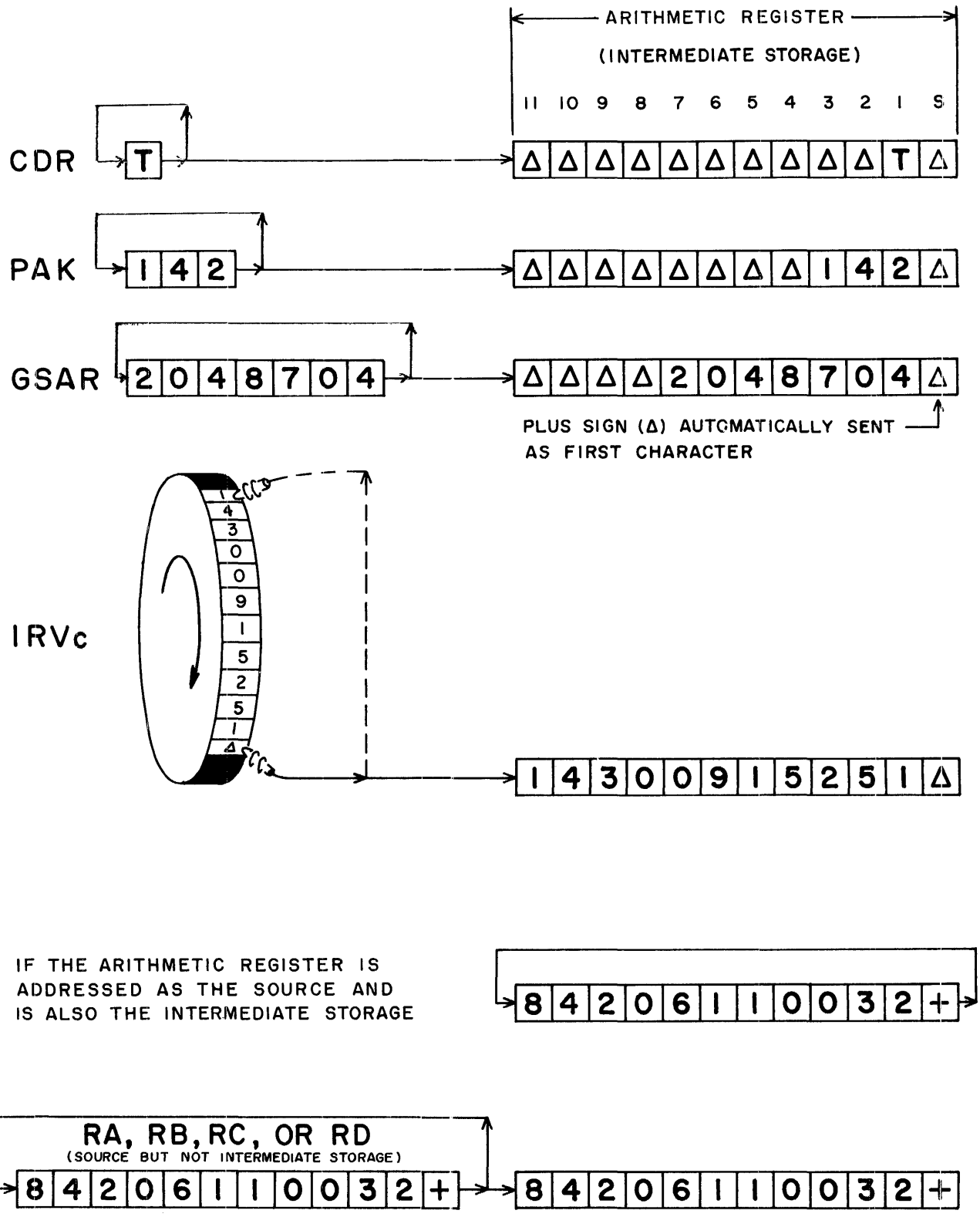


Figure III-13. IRV and the Registers as Sources for the Source→RD, Load V_1, and Load V_2 Data Transmissions.

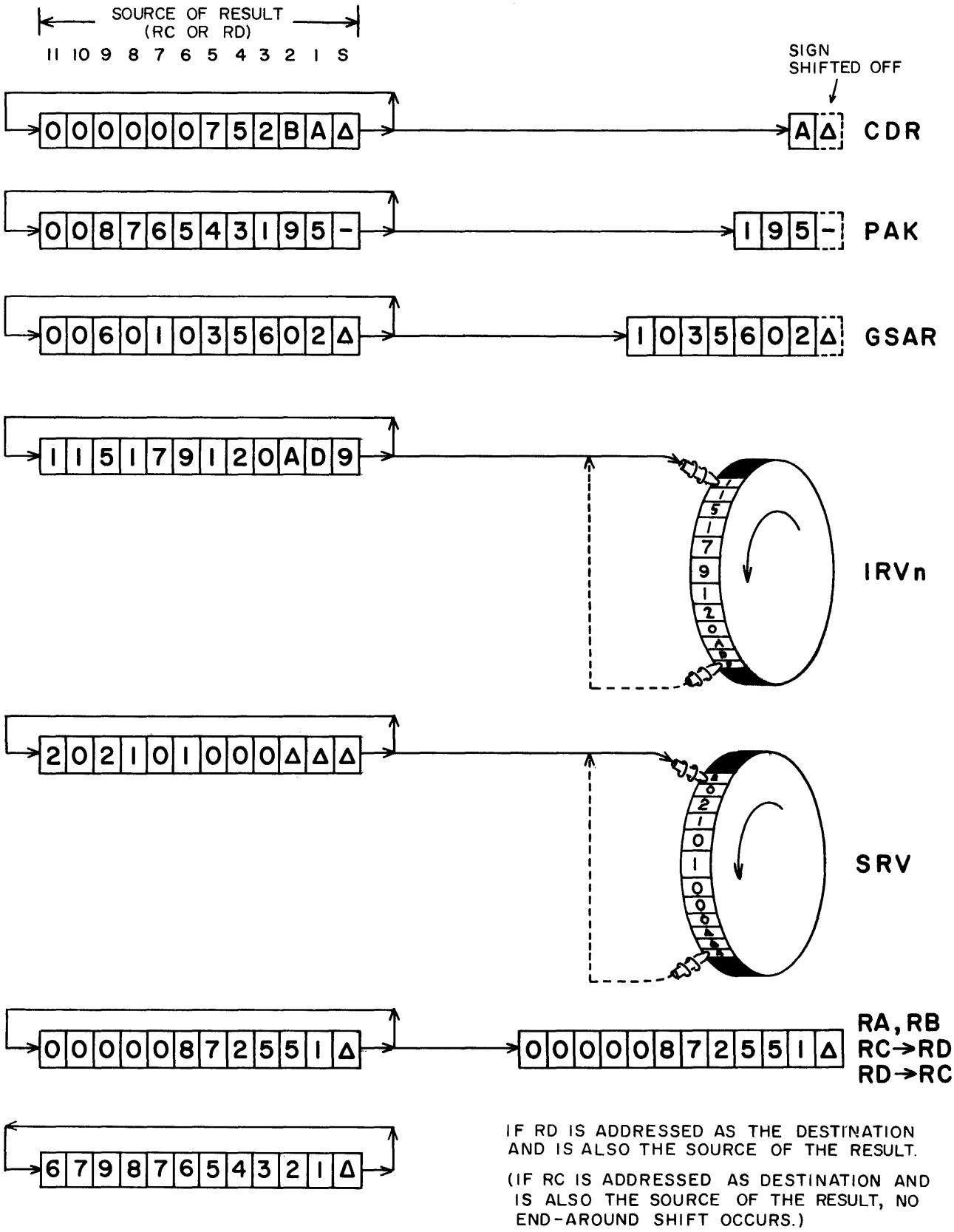


Figure III-14. IRV, SRV, and the Registers as Destinations for the RD→Destination and Store R Data Transmissions.

If Register D is used as a Destination, but not as a Source, data is placed in Register D in accordance with Rules 5a, 5b, and 5c above during the first half on the instruction. During the second half of the Arithmetic Transfer, an end-around shift of 12 characters occurs in Register D, and the transfer terminates.

If Register D is used as both Source and Destination, an end-around shift of 12 characters occurs during each half of the instruction, and the program continues. Register D data left in unaltered condition.

The timing for arithmetic-transfer-type data transmissions is given on Pages III-229 and III-230.

F. LOAD V_1

The data transmission that loads the first value, V_1, used in an instruction into the arithmetic section is the same as the Source ⟶ RD transmission except that the Source data (V_1) is placed in RA (or RA and RC) not RD. (See Table II-4, Page II-11.) The same basic rules (1 - 5c) outlined in Paragraph E, therefore, apply except that RA (or RA and RC) should be substituted for RD in each rule, as appropriate.

G. LOAD V_2

The data transmission that loads the second value, V_2, used in an instruction into the arithmetic section is also the same as the Source ⟶ RD transmission except that the Source data (V_2) is placed in RB not RD (See Table II-4, Page II-11). The same basic rules (1 - 5c) outlined in Paragraph E, therefore, apply except that RB should be substituted for RD in each rule, as appropriate.

H. LOAD IRVn

The acquisition of the next Instruction Word is also a data transmission similar to the Source ⟶ RD transmission. However, the intermediate storage used is always IRVn; and, as outlined on Pages II-20 through II-22 and Page II-229, only the following locations can be used as a Source for Instruction Words:

> Any Word Location in GSB, BTB, or on the I/O, FS, or IS Tracks; RA; RB; RC; RD; and IRVc.

I. STORE R

The Store R data transmissions are identical to the RD ⟶ Destination transmission except that RC or RD is used as the source of the result, depending on the instruction being executed. Rules 5d - 5f in Paragraph E thus apply directly, if RD is the source of the result. If RC is the source of the result, substitute RC for RD as appropriate in each rule. Note in Figure III-14 that, if RC is the source of the result and is also addressed as the Destination for R, no end-around shift occurs; i.e., the Store R operation terminates immediately.

The transmission U in RC ⟶ Destination that occurs in the Unconditional Jump Instruction Word, and in the Jump on Negative, Jump on Plus, and Jump on Zero Instruction Words when a jump takes place in those instructions is a Store R (RC ⟶ Destination) type of data transmission. However, only the Destinations listed on Page II-109 are permitted. If the V address is a Word Location on an I/O, FS, or IS Track or in BTB or GSB, only three characters (the lower order three, Sign position included, in RC) are recorded in the W-section of the V address. If the V-address is RC, no data is transmitted out of RC.

IV ARITHMETIC SECTION

All of the arithmetic and most of the logical instructions of the computer are performed in the Arithmetic Section. The prime function of the Arithmetic Section is then to carry out the process specified by certain (most) instructions.

The principal parts of the Arithmetic Section are

the four arithmetic registers:

Register A, RA
Register B, RB
Register C, RC
Register D, RD

a serial adder/subtractor,

a comparator; and

the necessary counters and control circuitry required for the execution of each of the processes involved in the instructions listed in Table IV-1, Page IV-2.

Details on the operation of the Arithmetic Section during each of the instructions listed in Table IV-1 are presented in Section II, Paragraph D, and are not, therefore, repeated here.

Table IV-1, however, summarizes the following information regarding the operation of the Arithmetic Section:

the specific instructions which involve the Arithmetic Section;

the initial* and final* contents of each of the Arithmetic registers: RA, RB, RC, and RD for those instructions; and

the particular register(s) in which the result is formed and from which the result is obtained in each instruction.

All characters in the Arithmetic Section are manipulated by the circuitry in Univac code, an excess-three binary-coded decimal notation. (See Pages A-16 and A-54 in Appendix A). However the operations are performed logically the same as one would write them down on paper. For this reason, problems can be formulated for the computer in ordinary decimal notation, using the alphanumeric symbol (as 8, or A, or 4, etc.) rather than the binary-coded decimal notation for each character. The binary-coded decimal notation need be used only when data is being manually inserted into the computer via the push-buttons on the control panels and console.

---

* The initial contents of a register means the contents of the register after the source data involved in the instruction has been loaded into the Arithmetic Section. The final contents of a register means the contents of the register at the end of the instruction, assuming that the result of the instruction is not stored in the register. If the result is stored in the register, then the result is the final contents of the register. This is valid only if no check is programmed.

Table IV-1. Initial and Final Contents of the Arithmetic Registers.

INSTRUCTIONS	CONTENTS OF								Result formed in	Result stored from
	Register A		Register B		Register C		Register D			
	Initial	Final	Initial	Final	Initial	Final	Initial	Final		
Add	Augend	Augend	Addend	Addend	PC	UN	PC	R	RD	RD
Add & Check	Augend	Augend	Addend	Addend	PC	Negative Zero	PC	R	RD	RD
Subtract	Minuend	Minuend	Subtrahend	Subtrahend	PC	UN	PC	R	RD	RD
Subtract & Check	Minuend	Minuend	Subtrahend	Subtrahend	PC	Negative Zero	PC	R	RD	RD
Multiply, Store Upper	Multiplicand	Multiplicand	Multiplier	Multiplier	PC	PROD U	PC	PROD L	RC & RD	RC
Multiply, Store Upper & Check	Multiplicand	Multiplicand	Multiplier	Multiplier	PC	Negative Zero	PC	Negative Zero	RC & RD	RC
Multiply, Store Lower	Multiplicand	Multiplicand	Multiplier	Multiplier	PC	PROD U	PC	PROD L	RC & RD	RD
Multiply, Store Lower & Check	Multiplicand	Multiplicand	Multiplier	Multiplier	PC	Negative Zero	PC	Negative Zero	RC & RD	RD
Divide, Store Quotient	Dividend	UN	Divisor	UN	Dividend	Remainder	PC	Quotient	RC & RD	RD
Divide, Store Quotient & Check	Dividend	UN	Divisor	UN	Dividend	UN	PC	UN	RC & RD	RD
Divide, Store Remainder	Dividend	UN	Divisor	UN	Dividend	Remainder	PC	Quotient	RC & RD	RC
Divide, Store Remainder & Check	Dividend	UN	Divisor	UN	Dividend	UN	PC	UN	RC & RD	RC
(Any) Divide Where Divisor = 0	Dividend	UN	Divisor	UN	Dividend	Zero	PC	Zero	RC & RD	RC or RD
(Any) Divide Where Quotient Digit < 1	Dividend	UN	Divisor	UN	Dividend	Dividend (Remainder)	PC	Zero	RC & RD	RC or RD
Mask Transfer	Operand	Operand	Mask	Mask	PC	R	PC	R	RD	RD
Compare	V_1 Operand	V_1 Operand	V_2 Operand	V_2 Operand	PC	PC	PC	PC	Branch Storage Set to +, 0, -	
Suppress Left Zeros	V_1 Operand	R	PC	PC	PC	PC	PC	R	RA & RD	RD
Left Normalize	V_1 Operand	Normalized Operand	PC	Normalized Count	PC	PC	PC	Normalized Operand	RA, RB, & RD	RD
Substitute (U, V, or W)	V_1 Operand	V_1 Operand	V_2 Operand	V_2 Operand	PC	R	PC	R	RD	RD
Jump on Plus	PC	PC	PC	PC	PC	RC used as intermediate storage for U. Final Contents of RC are unusable.	PC	PC	RC	RC
Jump on Minus										
Jump on Zero										
Unconditional Jump										
Arithmetic Transfer	PC	PC	PC	PC	PC	PC	Receives Source Data	Copy of Source Data	RD	RD

R - Result

PC - Previous contents

UN - Unusable

RA - Register A

RB - Register B

RC - Register C

RD - Register D

PROD U - higher-order characters of product

PROD L - lower-order characters of product

V INPUT/OUTPUT SYSTEM

A. GENERAL INFORMATION ON DATA TRANSMISSIONS TO AND FROM THE CENTRAL COMPUTER

This section (Input/Output System) and Section VI (General Storage Data Transmissions) describe the manner in which data is transmitted to and from the operating memory of the Central Computer, and the manner in which the Central Computer functions in automatic system-control. Two principal facts are stressed in these two sections:

> data can and should be supplied to, and received from, the operating memory of the Central Computer on a timed-shared basis with other operations in the system. (This means that by using the appropriate combination of Instruction Words, Sub-Instructions and Sub-Steps as required, the programmer can cause the Central Computer to be carrying out a logical or arithmetic operation at the same time that both General Storage and each I/O Unit are supplying data to, or receiving data from the operating memory of the computer.)
>
> the control information that can be exchanged between the Central Computer and other parts of the system should be optimumly utilized to permit the computer to adjust its program automatically, and thereby efficiently monitor the over-all operations of the system (this implies, of course, that the control information from other parts of the system be thoroughly understood, and skillfully applied in the design of the system program; it also implies, in the case of the Input/Output System, that part of the system programming must actually be done on the external devices, called I/O Units, that produce the control information; i.e., the control information must be identified for the computer program, and some method set up in the I/O Unit whereby certain control information can be applied to the Central Computer, as required. Conversely, the control information from the Central Computer must be identified for some I/O Units.)

A key point is that General Storage and each UFC I/O Unit are designed so that, once given an operation to perform, each can carry out its respective function completely independent of the other and of the Central Computer. An efficient system program will, therefore, not merely be concerned with the optimum sequence of arithmetic and logical operations in the Central Computer, but will also be designed to permit the Central Computer to monitor the General Storage System and each I/O Unit in the Input/Output System so that via time-shared operations:

> any data required by the program from any source is always available in the operating memory at the required time; and so that
>
> all data processed by the program is suitably stored or sent to an output device, as required.

To establish a correct notion regarding system operations, it is convenient, therefore, to associate the General Storage and Input/Output Systems, even though this may appear to be artificial in some respects. For example, as explained in Section VI, the General Storage Drums (unlike the I/O Units) are primarily intended as large-capacity, random-accress, external memories, and the basic mechanics of their operation is quite similar to that of the principal element of the operating memory of the Central Computer, viz, the High Speed Drum. General Storage, however, is a system memory, not a part of the operating memory of the Central Computer. Since this is the case, data in General Storage must first be placed in the operating memory of the Central Computer before the latter can use it. Similarly, Central Computer data must be placed in the General Storage Buffer before General Storage can manipulate that data. This is the same situation that exists in the Input/Output System. For example, the data on the medium manipulated by an input device is not available to the computer until (a) the input device is given a loading operation to perform, (b) it transcribes the data on the medium it manipulates, and stores a copy of that data in the operating memory of the Central Computer, and (c) the buffer memory (I/O Track) containing the input data is made available to the computer. Similarly data in the operating memory of the computer is not reproducible by an output device until (a) that portion of the operating memory containing the output data is made available to the output device, and (b) an operation is given the output device that permits it to unload the output data, and activate the required mechanisms to reproduce the output data. In certain system considerations, therefore, the General Storage Drums can and ought to be thought of as pseudo-I/O Units whose functions (and basic program control) are similar to that of the I/O Units discussed in this section.

Other system considerations also associate the General Storage and Input/Output Systems even though an opposite perspective is assumed. For example, the I/O media (particularly punched cards and magnetic tapes) will sometimes be regarded as external memories, and their associated I/O Units will be regarded as psuedo-storage systems.

To summarize the relationships, as well as the principal differences, between the General Storage and Input/Output Systems Table V-1 lists

the basic devices included in each system,
the manner in which operations are initiated in each system,
the particular portion of the operating memory of the Central computer with which each is associated as well as the general nature of the data transmissions that occur in each; and
the control information each system supplies the Central Computer.

Details on the operation of the Input/Output System are presented below. As noted above, the General Storage System is discussed in Section VI.

Figure V-1, Page V-6, a partial block diagram of the UFC Model 1 Input/Output System, illustrates that the following are the principal parts of that system:

Table V -1. A Brief Comparison of the General Storage and Input/Output Systems

	General Storage	Input/Output Systems
Basic devices included in each System	General Storage Drums	UFC I/O Units The following UFC I/O units are currently being built. A variety of other devices are under development. UFC Model 1 Console System UFC Inquiry Typewriter UFC 90-Column Card System UFC 90-Column Card System (with Post-Read Checking) UFC 80-Column Card System (Bull Unit) UFC Magnetic Tape Unit UFC High Speed Printer UFC High Speed Paper Tape System A special-purpose, magnetic tape equipment, the UFC Sort-Collate System, is also available as an auxiliary device for use in UFC Model 1 Systems. This device permits off-line sorting and collating operations to be performed, and also allows combination collating-and-updating operations to be carried out during psuedo on-line operations.
Manner in which Time-Shared Operations are initiated	By the following General Storage Sub-Instructions or Sub-Steps: Clear GSB to Ignores Read Unit Record Write Unit Record Write Unit Record & Check Channel Search = Channel Search ≠	By Demand In Instruction Words, or Sub-Steps during which signals are sent to an I/O Unit via the Computer - I/O Control Lines (A-J). These signals are sent to an I/O Unit when that I/O Unit is "on demand" and ready. Some signals on the C-I/O control lines (A-J) merely condition the I/O Unit for subsequent operation, others specify a particular operation that is to be performed. In some I/O Units the pattern of signals for each operation is fixed. In others, the C-I/O control lines are programmable on the I/O Unit.

(Continued on next page)

Table V -1. A Brief Comparison of the General Storage and Input/Output Systems (Continued from preceding page)

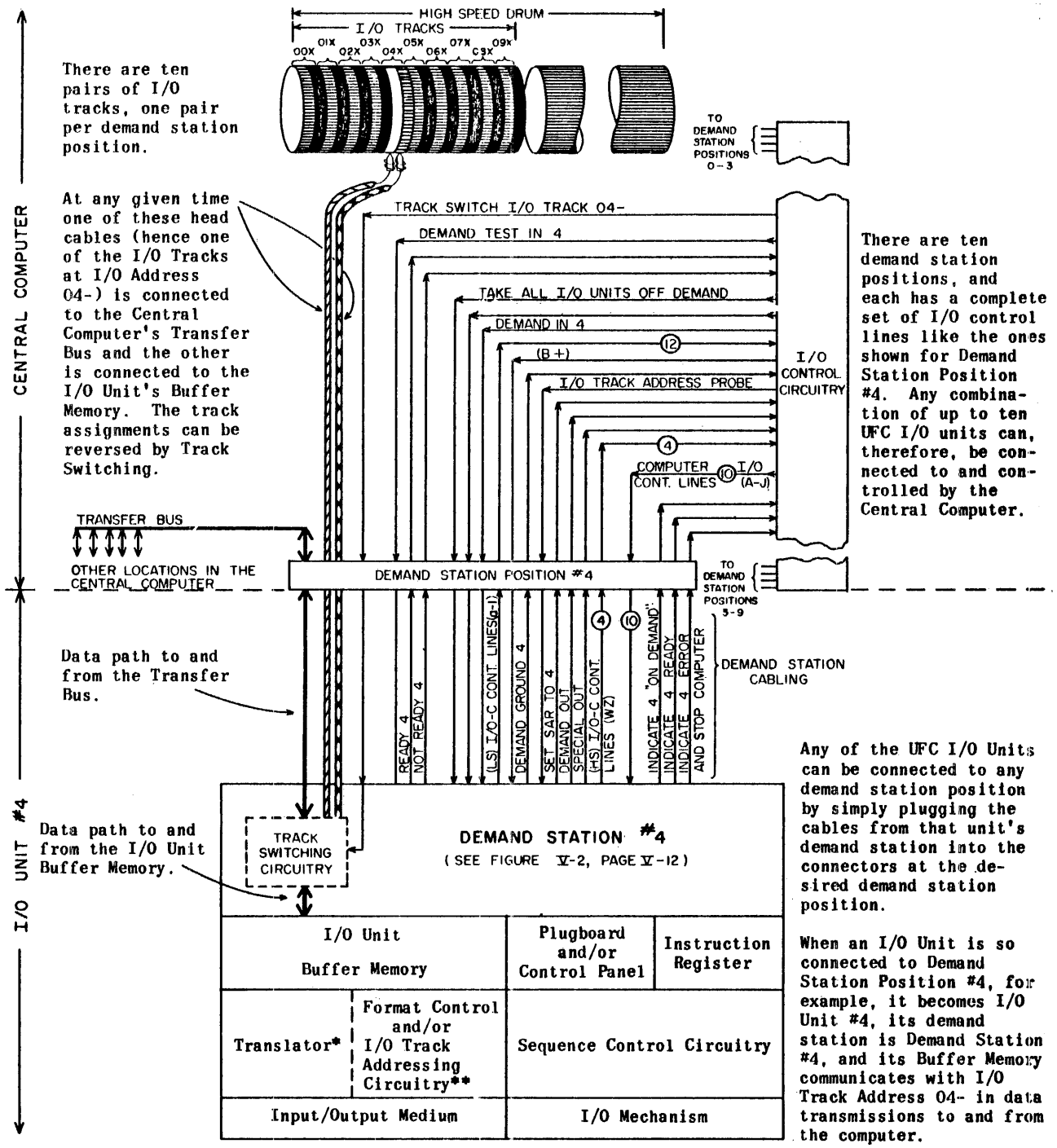
	General Storage	Input/Output Systems
The portion of the operating memory of the Central Computer that is used, and the nature of the data transmissions in each System	The General Storage Buffer is used as the communication link between the Central Computer and General Storage Drum. During General Storage operations, GSB is not Word or Field Addressable; data always leaves or enters GSB to and from the General Storage Drums beginning with GSB's Word 9, Character S position. No format control or translation of data takes place. When GSB is not engaged in General Storage operations, it is part of the operating memory of the Central Computer and as such is Word, Field and Blockette addressable.	When each I/O Unit is connected to the Computer, it is assigned to an I/O Track Address. Each I/O Track Address specifies a pair of tracks: at any given time the Computer is connected to one track of the pair and the I/O Unit is connected to the other track. Each I/O Track Address thus always has one track that is in the operating memory of the Central Computer and one that is not. This assignment can be reversed by a programmed Track Switch operation. The advantage of this type of input/output buffer memory is that the computer and I/O Unit can time-share storage references at the same I/O Track Address as well as time-share their respective operations. The computer can thus be unloading (i.e., using the data on) one track while an input device loads the other; the track assignment can then be reversed and the computer can unload the new input data, while the input device continues operation loading the other track. Data transmissions to and from an I/O Unit take place between the I/O Unit's buffer memory and the I/O Track. Input data flow is I/O medium $\longrightarrow$ I/O Unit Buffer Memory $\longrightarrow$ I/O Track. Some I/O Units have a translator which permits one or more codes other than Univac code to be used on their I/O medium. All I/O Units' Buffer Memories, however, transmit data to and from their I/O Unit's associated I/O Track in Univac code. In some I/O Units, format editing of the in-out data can be programmed; some I/O Units also include an I/O Track Addressing System, and in-out data can be stored or obtained from a particular location on the I/O Track.

(Continued on next page)

Table V -1. A Brief Comparison of the General Storage and Input/Output Systems (Continued from preceding page)

	General Storage	Input/Output Systems
Con- trol Infor- mation Sent to the Com- puter	Channel Search Operations General Storage interlocks GSB and GSAR, and notifies the Computer of the status of the search and the results of the search. That is, the Computer program can test to see if a previously initiated channel search is completed or not:* if the channel search is not completed, the computer program can continue immediately with some other operation, or it can wait depending on what is programmed. (See Channel Search Probe and Channel Search Probe & Wait, Section VI, Page VI-29.) if the channel search is completed, the data obtained in the search is available to the computer program for processing, and the reported results of the search are available for program variance. Other Operations No control information, as such is sent to or set up in the computer. GSB and GSAR are interlocked, however, as in the Channel Search operations. Although the program cannot test to see if these other operations are completed, it will always observe the correct procedure with respect to General Storage if GSB or GSAR are addressed prior to beginning all General Storage operation. By interlocking GSB and GSAR General Storage, in a sense, thus controls the computer program.	Demand Test In sequences The I/O Unit tested immediately notifies the Computer that it is READY or NOT READY for subsequent use. Demand In sequences Two sets of Control information can be sent to the Computer during these sequences (each must be programmed as to their definition and time of occurrence on the I/O Unit itself). (LS) I/O-C control lines (a-l) These are B+ lines that cause their correspondingly named hubs on the Program Control Plugboard to emit B+ shortly after the I/O Unit involved goes "on demand". (HS) I/O-C control lines (W-Z) These are pulse lines that set High Speed I/O-Computer Control Line Storage in the Computer when the I/O Unit "on demand" goes ready. The (LS) I/O-C CONTROL LINE hubs (a-l) on the Program Control Plugboard are patched to SELECTOR PICKUP hubs. By probing the appropriate Selectors the computer program can identify the control information sent and achieve program variance accordingly. Test Incoming Control Instruction Words or SPECIAL OUT - to - W, X, Y, Z IN patching can be used to find out which (HS) I/O-C control lines W-Z set High Speed I/O-C Control Line Storage, and the program can be varied or sequenced accordingly.

* The computer actually indicates the interlock on GSB and GSAR each time a General Storage operation is initiated; General Storage maintains it until the operation is completed.



*All data transmissions between an I/O Unit's Buffer Memory and the I/O Unit's associated I/O Track involve only Univac coded characters. Most UFC I/O Units employ some type of translator, and can, therefore, manipulate data in another (one or more) code(s) between the I/O Unit's Buffer Memory and the input/output medium.

**Some I/O Units have plugboards on which the format of input/output data can be varied, and on which a limited amount of editing can be programmed; some have an I/O Track Addressing system that can be programmed to allow input/output data to be delivered to and received from specific locations on the I/O Track; some I/O Units, as the UFC Magnetic Tape Units do not permit any translation, formating, editing, or I/O Track addressing (i.e., data is stored in the computer exactly as it is read from the tape and data is recorded on the tape exactly as it is received from the computer.)

Figure V-1. Partial Block Diagram of UFC Model 1 I/O System

I/O Control circuitry
Demand Station Positions (0-9)
I/O Tracks
Transfer Bus
} These are in the Central Computer

Any combination of up to ten UFC I/O Units, each of which has the following general composition:

Demand Station
Instruction Register
Plugboard and/or Control Panel
Sequence Control circuitry
I/O Mechanism
I/O Medium, and an
I/O Unit Buffer Memory which may or may not have an associated

Translator, and/or
Format Control, and/or
I/O Track Addressing system.

Those parts of the system which function the same regardless of the I/O Unit involved (as the I/O Control circuitry, Demand Station Position, and each I/O Unit's Demand Station) are discussed in detail below. U 1300, Univac File-Computer System Specifications, can be consulted for further information on the mode of operation and composition of each UFC I/O Unit. Other detailed information on each I/O Unit is in preparation.

B. CENTRAL COMPUTER PORTION OF I/O SYSTEM

1. I/O Control Circuitry

The I/O Control circuitry is the collection of circuits in Program Control whose principal function is to (via Demand Station Positions) communicate with the Demand Station in each I/O Unit in accordance with the system program.

The operation of this circuitry is initiated in internally-stored programs by the execution of the Test Demand In, Demand In, or Test Incoming Control Instruction Words described on Pages II-127 through II-145. The operation of this circuitry is initiated in plugboard-defined programs by the I/O Control Sub-Steps discussed on Pages II-73 through II-79.

The principal parts of the I/O Control circuitry are

the selection circuits which determine which I/O Unit is to be involved each time the system program specifies an operation in the I/O System. As noted above this circuitry can be operated either by an internally-stored program or a plugboard-defined program. Where necessary (i.e., in those cases where

the I/O Unit selected for operation is to reply to the computer) this circuitry is designed to "remember" which type of program initiated the operation, and it routes the I/O reply accordingly. Internally-stored programs thus receive I/O Unit replies internally in Program Control, whereas plugboard-defined programs receive I/O Unit replies at hubs on the Program Control Plugboard.

Ten identical* sets of lines, each set of which connects Program Control to a Demand Station Position. One set of these lines, to Demand Station Position #4, is illustrated in Figure V-1. Note that the set of lines shown is arranged in groups. Note also that the lines emanating from the computer are identified between I/O Control and Demand Station Position #4, whereas the lines emanating from I/O Unit #4's Demand Station are identified between Demand Station #4 and Demand Station Position #4. When the system program specifies that the computer and a particular I/O Unit are to communicate, one of these sets of lines is activated for use by the selection circuits of I/O Control. The function of each of these lines is described below in Paragraph V C1.

Storage Address Register, SAR. Although SAR has other functions as well, it is Program Control's source of the I/O control characters (abc) and (xxx) in the execution of I/O Control Instruction Words. (See Pages II-127 through II-145). By the same token all of the I/O Control Sub-Step hubs on the Program Control Plugboard can be regarded as part of the I/O Control circuitry.

High Speed I/O-Computer Control Line Storage. This is a special-purpose 4-bit (non-addressable) memory in the Central Computer that is used to remember certain control information, the (HS) I/O-Computer control line signals W, X, Y, Z, from the I/O Units. As illustrated later in this section, each time an I/O Unit is demanded, I/O Control clears High Speed I/O-Computer Control Line Storage. The (HS) I/O-Computer control lines (W-Z) from the Demand Station of the I/O Unit "on demand" then can set this special-purpose memory as required. (If the basic rule is observed that HS I/O-C Control Line Storage is to be tested each time a SPECIAL OUT, described below, is produced by an I/O

* Additional connective lines have been included in the set to Demand Station Position 0, so that certain specialized operations of the UFC Inquiry Typewriter can be achieved.
All other sets are identical.

Unit the program can readily recognize HS I/O-C control line information from each I/O Unit as appropriate.) The following important facts concerning HS I/O-C Control Line Storage should be thoroughly understood:

There are ten sets of inputs to this memory: one from each of the ten I/O Units' Demand Stations. Only one I/O Unit's Demand Station can be connected to HS I/O-C Control Line Storage at a time. An I/O Unit's Demand Station is connected to HS I/O-C Control Line Storage when that I/O Unit is placed "on demand" and goes ready. If any of the W, X, Y, and/or Z lines in the I/O Unit's Demand Station are energized at this time, the corresponding signals W, X, Y, Z are sent to HS I/O-C Control Line Storage. If none are energized this memory remains cleared. The testing of HS I/O-C Control Line Storage is done completely inside the Central Computer; no I/O Unit's Demand Station is involved.

2. Demand Station Positions

Each Demand Station Position is fundamentally a set of connectors that links together a particular set of lines from I/O Control and the cabling from a Demand Station. That is, each Demand Station Position is a tie-point for a full set of I/O control lines that connect to I/O Control inside the computer and to a Demand Station outside the computer. The lines to and from I/O Control and the Demand Station Position are permanently wired lines. Those to and from a Demand Station and the Demand Station Position are connected to (and disconnected from) the Demand Station Position when an I/O Unit is physically connected to (or disconnected from) the computer.

An I/O Track Address is associated with each Demand Station Position. The two are correspondingly numbered: I/O Track Address 00x is connected to Demand Station Position 0, I/O Track Address 01x is connected to Demand Station Position 1, etc. When an I/O Unit's Demand Station is connected to a particular Demand Station Position, then, the I/O Track Address with which that I/O Unit communicates is automatically determined.

3. I/O Tracks on the High Speed Drum

Each I/O Track Address actually specifies a pair of tracks, and the head cabling to each pair of tracks is permanently connected to a correspondingly-numbered Demand Station Position. As illustrated in Figure V-1, at any given time one track of the pair is connected to the computer and the other track of the pair is connected to the Buffer Memory of the I/O Unit whose Demand Station is plugged to the Demand Station Position associated with the pair of tracks. Both the Central Computer (via its Transfer Bus) and the I/O Unit (via its Buffer Memory) can independently time-share storage references to their respective tracks. The track assignments can be reversed by a Track Switch operation. If this is done, the track formerly connected to the Central Computer's Transfer Bus can be made

available to the I/O Unit's Buffer Memory, and what formerly was the I/O Unit Buffer Memory's track can be made available to the Central Computer's Transfer Bus.

This design of I/O Track logic is an important feature to exploit to successfully time-share input-output operations. Not only can the Central Computer and each I/O Unit carry out their respective operations independently, but the Central Computer and an I/O Unit can use the same I/O Track Address simultaneously. An (instantaneous) Track Switch operation can then be carried out to make the data stored by the computer (I/O Unit) available to the I/O Unit (computer). In output operations, for example, the computer can load the track to which it is connected, while the I/O Unit unloads (previously stored) data from the other track, i.e., from the track to which it is connected. A Track Switch operation can then be carried out, and the new output data made available to the I/O Unit, while the computer loads the next group of output data on the alternate track. Similar techniques can be employed for input operations.

Note on Figure V-1 that the data transmitted to and from an I/O Track Address by the computer actually passes through the I/O Unit's Demand Station. If no I/O Unit is connected to a particular Demand Station Position, the computer cannot refer to the I/O Track Address associated with that Demand Station Position. In this connection there are several cases to consider.

Installation uses less than 10 UFC I/O Units:

In this case a minor field modification (described later in this section) can be made, so that one track of the pair of each set of unused I/O Tracks is available to the Central Computer as an IS Track, if this is desired.

Installation uses 10 UFC I/O Units (i.e., no field modification is carried out) but one or more is disconnected because

the TEST/NORMAL switch for that I/O Unit (on Program Control Cabinet #1) is in the TEST position; or because

one or more I/O Units' Demand Station's cabling is disconnected from the computer.

4. Central Computer Transfer Bus

The Central Computer's Transfer Bus is associated with the Input/Output System only insofar as it is involved (via a Demand Station) in the transfer of data to and from one Program Control Storage Location and an I/O Track.

C. PORTION OF I/O SYSTEM IN EACH UFC I/O UNIT

1. Demand Station

As noted above, an I/O Unit's Demand Station is part of control circuitry of the I/O Unit; and is, specifically, the means whereby the computer (via I/O Control) communicates with the I/O Unit.

When an I/O Unit is physically connected to the computer, cabling from its Demand Station is plugged into one of the ten Demand Station Positions (0-9) in the computer. This cabling is the only wiring between the I/O Unit and the computer. Any UFC I/O Unit can be plugged into any Demand Station Position.* Once plugged, however, it is designated as I/O Unit "b" (where "b" is numerically the same as the Demand Station Position -0-9 to which it is connected), and it communicates only with a correspondingly numbered I/O Track (0"b"-) on the High Speed Drum in data transmissions.

The principal circuits in a Demand Station are the

Demand Test In circuits
Demand In circuits
Track Assignment & Track Switching circuits

The operation of each of these groups of circuits is discussed below, and illustrated in Figure V-2, Page V-12).

a. Demand Test In Circuits

The Demand Test In circuits permit the computer to test the operating (or ready/not ready) status of an I/O Unit without affecting the demand status of that I/O Unit (or that of any other I/O Unit), and without delaying the central computer program in any way. The principal element in these circuits is the Ready flip-flop.

When the Ready flip-flop in an I/O Unit's Demand Station is set to 0, the I/O Unit is said to be in a not ready condition; when this flip-flop is set to 1, the I/O Unit is said to be in a ready condition.

The setting of the Ready flip-flop (0 or 1) is

* The UFC Inquiry Typewriter must be connected as I/O Unit 0. No other restrictions exist.

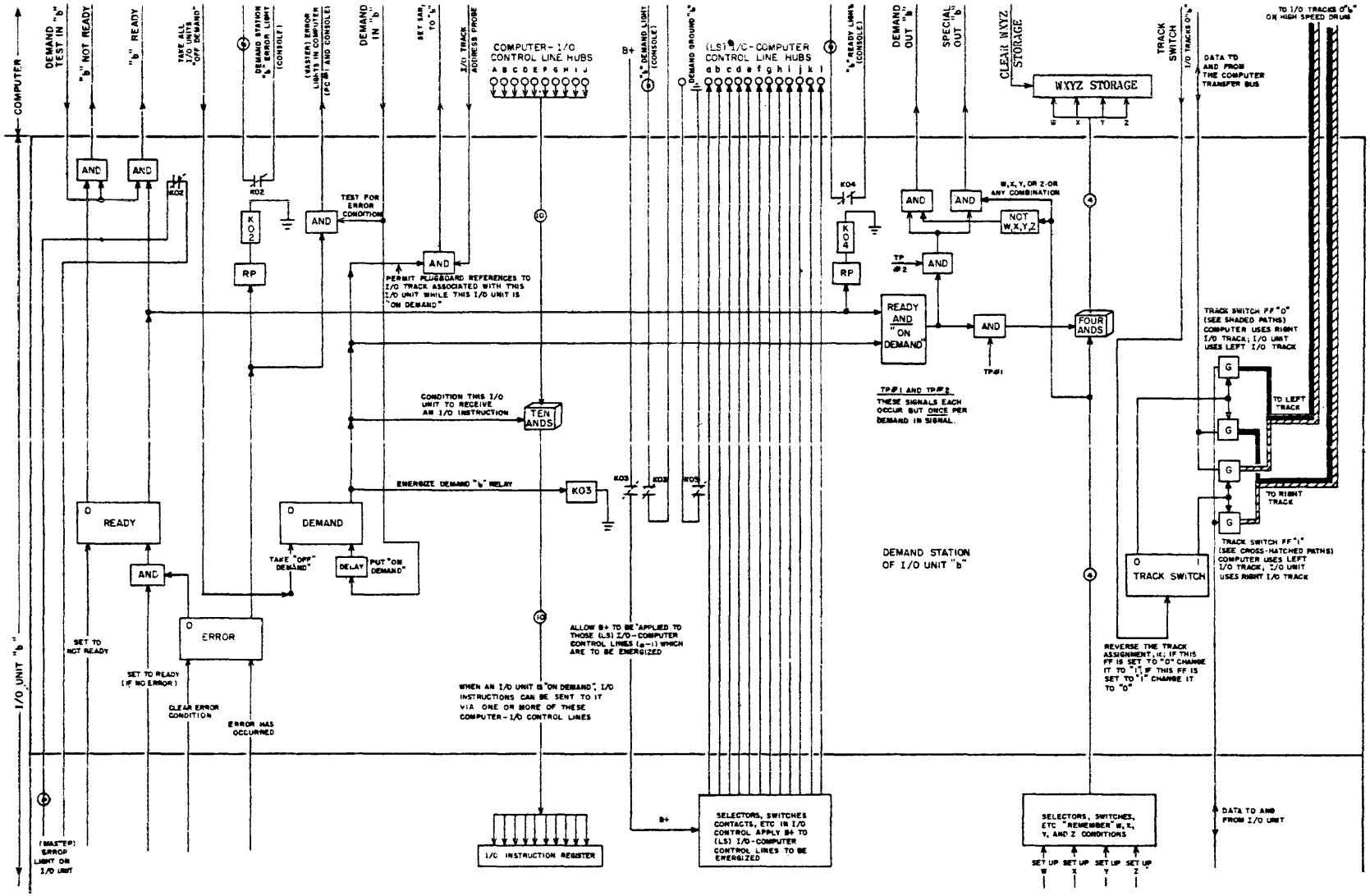


Figure V-2. Detailed Block Diagram of a Demand Station

controlled entirely within the I/O Unit by the signals SET TO NOT READY and SET TO READY (IF NO ERROR). These signals are generated by the operating control circuitry (called Sequence Control circuitry) in each I/O Unit. The manner in which the ready condition is set-up and removed varies with each I/O Unit, but is one* of the particulars concerning the operation of each I/O Unit that the programmer should know. In general, the following are the basic rules whereby each I/O Unit remembers its operating status via the Ready flip-flop:

If certain manual operations are required to initially prepare an I/O Unit (internally) for operation, the Ready flip-flop cannot be set to 1 (i.e., the I/O Unit will not initially go ready) until these required operations are performed.

Once initially ready (i.e., its Ready flip-flop is set to 1) an I/O Unit can be given an I/O Instruction to perform by placing the I/O Unit "on demand", and sending it signals over the C-I/O control lines A-J. When the I/O Unit begins an operation, its Sequence Control circuitry sets the Ready flip-flop to 0; the I/O Unit then goes not ready.

At some point in the operation in which it is engaged, the I/O Unit will be restored to a ready condition (i.e., its Ready flip-flop will be reset to 1) if no error exists in the I/O Unit. The exact point at which this occurs varies from I/O Unit to I/O Unit; it may be in the middle of the cycle the I/O Unit carries out to perform the operation, or it may be at the end of the cycle, depending on the I/O Unit. An important point to remember is that if power is lost in an I/O Unit or some other error condition occurs, the I/O Unit cannot go ready. (See Error flip-flop, the master error "memory" for each I/O Unit, in Figure V-2.)

If the Ready flip-flop is set to 0 when a DEMAND TEST IN signal is received from the computer (during the execution of a Test Demand In Instruction Word or a Demand Test In Sub-Step), this flip-flop will cause a NOT READY signal to be sent to the computer. If set to 1, it will cause a READY signal to be sent to the computer. Note, in Figure V-2, that either reply, READY or NOT READY, is produced immediately.

* Other particulars include the preparatory manual operations required to set up an I/O Unit for use, the exact manner in which an I/O Unit can be taken off-demand, the definition of C-I/O, (LS) I/O-C, and HS I/O-C control lines in the I/O Unit, and any other programming that is done on the I/O Unit, as I/O Track Addressing, Start-Stop control of the I/O Unit, etc.

When the Ready flip-flop in an I/O Unit's Demand Station is set to 1, the READY "b" light corresponding to that I/O Unit on the Console is lit. An I/O Unit must be ready to complete those Demand In sequences described below which involve an I/O Unit reply (DEMAND OUT or SPECIAL OUT).

b. Demand In Circuits

The Demand In circuits permit control information affecting the operation of the I/O Unit and the computer to be exchanged between the I/O Unit and the computer. The principal components in these circuits are the Demand and the Ready flip-flops.

The Demand flip-flop in an I/O Unit "remembers" the demand status of that unit. If its Demand Station's Demand flip-flop is set to 1, an I/O Unit is said to be "on demand"; if its Demand Station's Demand flip-flop is set to 0, an I/O Unit is said to be "off demand". The Demand flip-flop is set to "1" when the computer executes the appropriate Demand In Instruction Word or Demand In Sub-Step for the I/O Unit to be placed "on demand" (See DEMAND IN signal). The Demand flip-flop is set to "0" when another I/O Unit is placed "on demand". (See TAKE ALL I/O UNITS OFF DEMAND signal).

All UFC I/O Units are taken "off demand" when another unit is placed "on demand".

The Ready flip-flop is operated as described above under the Demand Test In circuits.

The Demand In sequence is described below in detail to illustrate the specific operation of the Demand In circuits.

When the appropriate Demand In Instruction Word or Demand In Sub-Step (0-9) is initiated, the following events take place:

a signal called TAKE ALL I/O UNITS "OFF DEMAND" is produced which sets the Demand flip-flop in each I/O Unit's Demand Station to 0;

a CLEAR signal for High Speed I/O-Computer Control Line Storage is produced which destroys the previous contents of that memory and places it in a cleared condition; and

a DEMAND IN (0-9) signal is produced which tests to see if the I/O Unit to be placed "on demand" contains any error condition: if it does,

an ERROR signal is sent to the computer, computer operation stops, and the I/O Unit's ERROR lights "b" on the Console (no DEMAND OUT or SPECIAL OUT described below are produced; the I/O Unit is, however, placed "on demand"); if no error condition exists the Demand In sequence continues in the normal manner; and

the appropriate I/O Unit (0-9) is then placed "on demand", i.e., the Demand flip-flop in the correct I/O Unit's Demand Station is set to 1*.

When a unit's Demand flip-flop is set to 1:

the Demand relay in its Demand Station is energized; when this relay's contacts close:

B+ can be applied to the (LS) I/O-to-Computer control lines a-l that have been programmed in the UFC I/O Unit to be energized at this time;

all Selectors whose SELECTOR GROUND hubs are patched to DEMAND GROUND "b" then have a ground supply; and

the DEMAND "b" light on the Console is lit.

that UFC I/O Unit (and only that UFC I/O Unit) is conditioned to receive an I/O Instruction via its Demand Station's Computer-to-I/O Control lines A-J;

that UFC I/O Unit's (and only that UFC I/O Unit's) associated I/O Track Address can be referred to in storage references via the I/O WORD and I/O FIELD hubs on the Program Control Plugboard; and

that UFC I/O Unit's Demand Station is conditioned to produce either of the following for the computer when the UFC I/O Unit goes ready:

DEMAND OUT signal; or

SPECIAL OUT signal and signals over one or more of the (HS) I/O-Computer control lines W, X, Y, Z.

(If the I/O Unit is ready when demanded, the out signals are produced without delay.)

* The signal which takes all I/O Unit "off demand" is simultaneous with the DEMAND IN (0-9) signal. The DEMAND IN (0-9) signal's function to set the Demand flip-flop in I/O Unit (0-9) to 1, is therefore appropriately delayed.

Notes:

1. Neither out, DEMAND OUT or SPECIAL OUT, is produced unless the I/O Unit is both "on demand" and ready.

2. SPECIAL OUT is produced if any one or any combination of the (IIS) I/O-Computer control lines W, X, Y, Z in the UFC I/O Unit are programmed to be energized at this time. If no (IIS) I/O-Computer control lines are energized, a DEMAND OUT is produced. Only one out is produced per DEMAND IN signal.

3. The computer program can be designed to use the DEMAND OUT and SPECIAL OUT signals in a variety of ways. The receipt of these signals is the optimum time to send the UFC I/O Unit producing them an I/O Instruction via the Computer-I/O Control lines A-J. It is also the best time to Track Switch when this is required. The Demand In Instruction Word, when it sends I/O Instructions, always sends them immediately after one of these out signals. It also track switches at that time. (See Section II, Pages II-138 and II-139.)

c. Track Assignment and Track Switching Circuits

As has already been emphasized, each I/O Track Address specifies a pair of tracks, one of which connects to the Central Computer via the Transfer Bus and the other of which connects to an I/O Unit's Buffer Memory. Both connections are made in the I/O Unit's Demand Station. The principal element involved is the Track Switch flip-flop. Output enables from this flip-flop are used internally in the Demand Station to assign one track to the computer and another to the I/O Unit. Note: When I/O Tracks are used as such, the reading and writing circuits for both tracks at an I/O Track Address are contained in the Demand Station plugged to the Demand Station Position associated with that I/O Track Address. When I/O Tracks are used as IS Tracks, reading and writing circuits in the computer are employed. (The connection of these circuits is the minor field modification referred to above.) No reading and writing circuits are shown in Figure V-2.

The Track switch flip-flop is operated by I/O Control as specified by Demand In or Test Incoming Control Instruction Words or by Track Switch Sub-Steps. A TRACK SWITCH signal to I/O Unit "b's" Demand Station reverses the setting of the Track Switch flip-flop in that Demand Station. That is, if the Track Switch flip-flop was set to 1, the TRACK SWITCH signal changes its setting to 0; if set to 0, the TRACK SWITCH signal will set it to 1. As illustrated in Figure V-2, the 1 setting of the Track Switch flip-flop specifies one assignment for the I/O Tracks (see cross-hatched assignment) and the 0 setting specifies an opposite assignment (see shaded assignment).

d. I/O Track Address Probe

As explained in Section II, Page II-47, when an I/O WORD or FIELD hub is enabled in a Program Step

the highest-order stage of SAR is set to 0,

the lowest-order stage of SAR is set to 0-9 or A-V, depending on the hub enabled, and

a test is made of the demand status of each I/O Unit.

The I/O TRACK ADDRESS PROBE signal (See Figure V-2) is used to make this test. If an I/O Unit is "on demand" its Demand Station will produce a SET SAR_1 (middle digit position of SAR) TO "b" signal when tested as above, and the I/O Track Address associated with the I/O Unit "on demand" will be completed in SAR. The same sequence of events occurs when prime (') is detected as the middle digit of the U, V, or W address of an instruction word.

Note: Only one I/O Unit can be placed "on demand" at a time; hence the cautions noted on pages II-46 and II-78 must, therefore, be observed.

e. Coding I/O Track addresses in internally stored programs

(1) To obtain V_1 and/or V_2 and to store R

Normal Procedure: (direct addressing)

use 00x to refer to I/O Track 00

use 01x to refer to I/O Track 01

use 09x to refer to I/O Track 09

The above normal procedure is still available and should be used as appropriate.

Alternate Procedure: (relative addressing)

If U, V, or W are coded so that they have a prime (') for their middle digit, as

U = 0'3

the I/O Track associated with the I/O Unit "on demand" is automatically referred to. (In the above example, when U is placed in SAR, V_1 would be obtained from the Word 3 location of the I/O Track whose associated I/O Unit is currently "on demand").

When prime (') is detected in the middle stage of SAR, a test is made of the the demand status of each I/O Unit. The I/O Unit "on demand" then appropriately sets the middle stage of SAR to 0, 1, 2, ..., or 9 so that a storage reference to that I/O Unit's corresponding I/O Track Address can be carried out. (Note: the mechanics of this are basically the same as those employed when the I/O WORD and FIELD hubs are enabled in plugboard programs).

f. Basic Rules for I/O Control

(1) Demand Test In/Demand In sequences.

The appropriate use of these sequences is explained in Figures V-3 and V-4, Pages V-18 and V-19 , respectively.

(2) (LS) and (HS) I/O-Computer Control Lines.

The meaning of these lines must be programmed on each I/O Unit's Plugboard and/or Control Panel; and provision made in the system program for the computer to identify the information from each line when that line is energized. In the case of the (LS) I/O-Computer control lines (a-l) it should be remembered that these lines, when energized, apply B+ to their correspondingly-named (LS) I/O-COMPUTER CONTROL LINE hubs (a-l) only as long as the I/O Unit in which they are energized is "on demand". In the case of the (HS) I/O-Computer control lines it is a fundamental rule that High Speed I/O-Computer Control Line Storage be tested each time a SPECIAL OUT is produced. Particular care must be exercised in designing the "on-off" control of the (LS) and (HS) I/O-Computer control lines in each I/O Unit; i.e., not only do the "memories" that retain this information in the I/O Unit for the computer need to be set as appropriate, but they must be cleared as well when that information has been used as required by the computer.

(3) Computer-I/O Control Lines

These lines are programmable on some I/O Units, but not on others. For example, they are not programmable on the UFC Magnetic Tape Units; they are on the UFC Card Systems.

As explained on Page II-78, the COMPUTER-I/O CONTROL LINE hubs (A-J) on the Program Control Plugboard should never be pulsed unless the particular I/O Unit that is to receive the I/O

If a series of Demand Test In and Demand In Instruction Words or Sub-Steps are programmed in pairs as illustrated below, the computer will, in effect, scan all I/O Units in a regular or a random fashion, as programmed, and keep all I/O Units that are ready for operation in operation without introducing any delay or holding up the computer program. This technique for I/O Control assumes, of course, that it is not necessary to perform a Demand In sequence in a particular I/O Unit when that I/O Unit's ready status is tested. This is commonly the case, for example, if three or four I/O Units are each supplying similar raw data to be processed by the main program: here the objective is to process all the raw data in any order as fast as possible; this means that the units that are ready should be set in operation again immediately after they go ready. By designing an appropriate "loop" of Demand Test In-Demand In sequences, the programmer can make the computer automatically find each I/O Unit that is ready in an optimum time, and keep the input-output operations of the system at maximum speed.

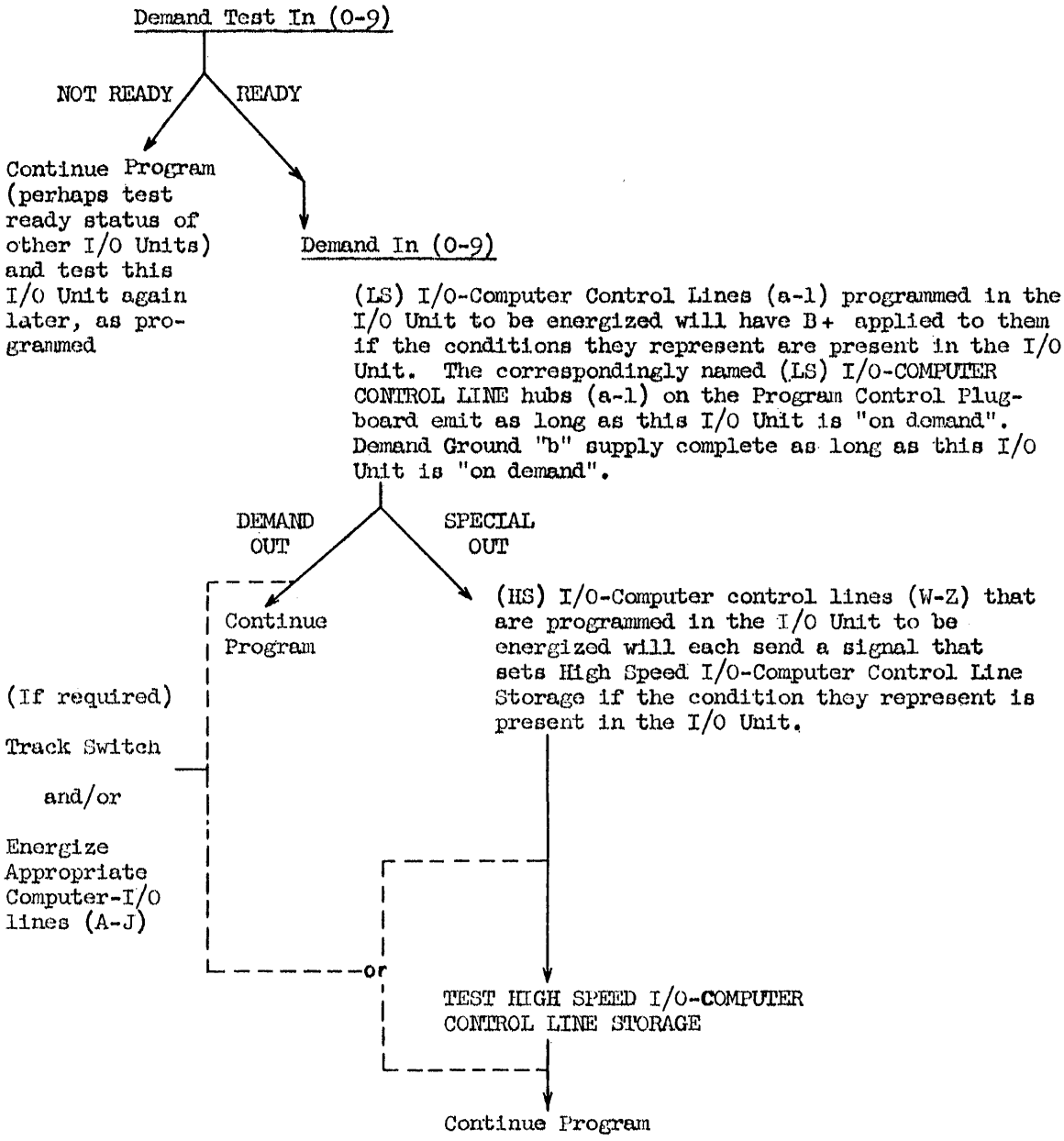


Figure V-3. Demand Test In/Demand In Sequences

If a Demand In sequence in a particular I/O Unit must be executed at a specific point in the computer program, do not precede the Demand In Instruction Word or Sub-Step by a Demand Test In Instruction Word or Sub-Step. (In such a case, nothing is gained from the Demand Test In sequence.) This situation can exist, for example, if the I/O Unit must be placed "on demand" to complete a Program Control Storage Address in the next Program Step wherein an I/O WORD (0-9) or I/O FIELD (A-V) hub is to be enabled; or, if an I/O operation (eg., reading a blockette of information from a UFC Magnetic Tape Unit) is required in a particular I/O Unit before the program can continue (because the data so obtained is to be used next in the computation).

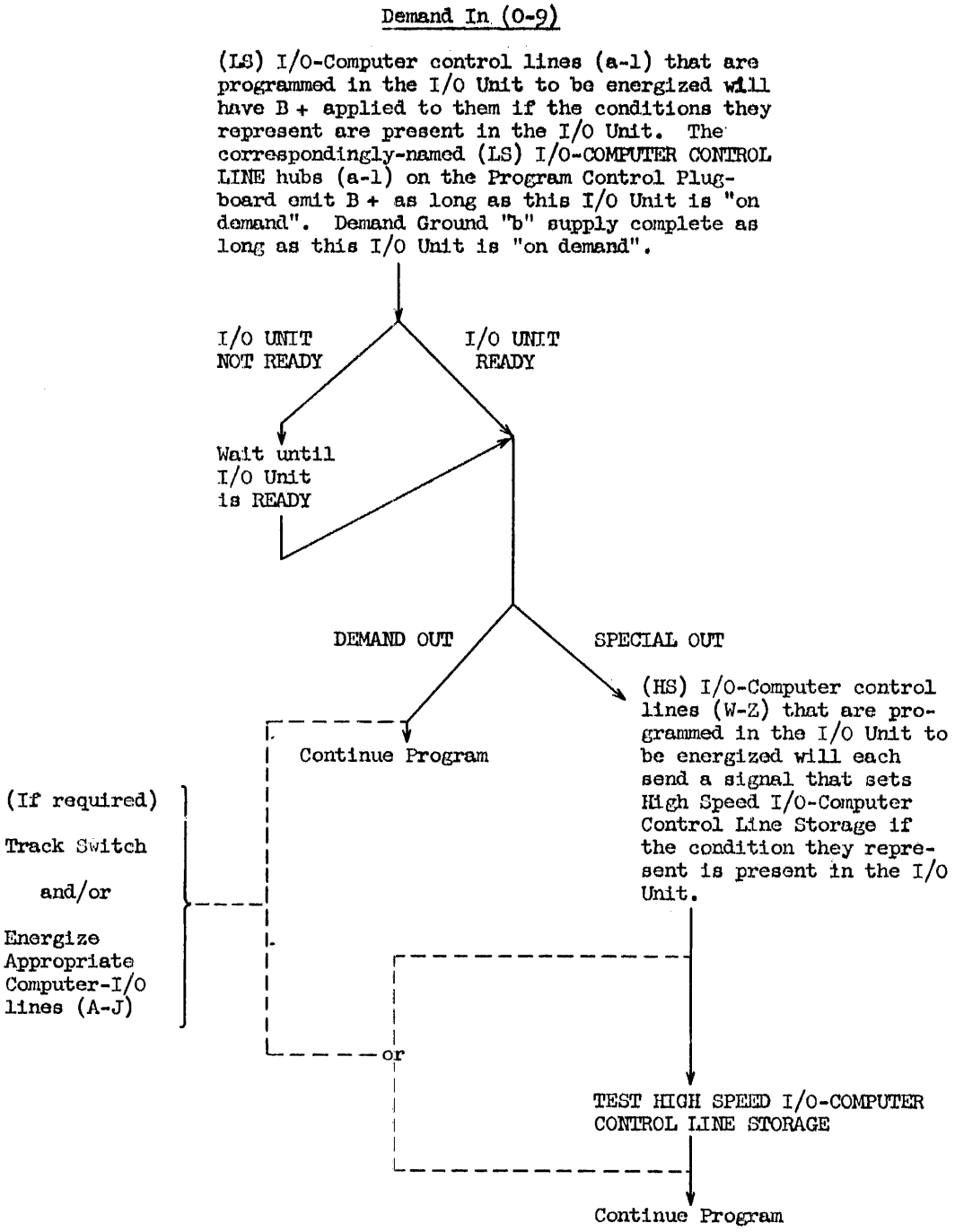


Figure V-4. Demand In Sequences only

Instruction is "on demand" and ready.

(4) Track Switching

An I/O Unit need not be "on demand" or ready for track switching to occur. However, as explained on Page II-79, a TRACK SWITCH (0-9) hub should never be pulsed unless it is known that the I/O Unit associated with the I/O Tracks to be switched is ready.

2. Instruction Register

Each I/O Unit has a "memory" of some type called an Instruction Register, in which each I/O Instruction received from the computer via the Computer-I/O control lines (A-J) is remembered. Sequence Control circuitry interprets the contents of the Instruction Register to determine what is specified by an I/O Instruction. The demand and ready status of an I/O Unit control the inputs to the Instruction Register. Sequence Control clears the Instruction Register when each I/O Instruction has been carried out.

3. Plugboard and/or Control Panel

The programming required on an I/O Unit is performed on its plugboard and/or control panel. Not all I/O Units have a plugboard, but each has a control panel; the latter, in addition to its programming section, also contains all the manual controls and the indicators employed on the I/O Unit.

4. Sequence Control Circuitry, I/O Mechanism, and I/O Medium

As its name implies, the Sequence Control circuitry is the operating control circuitry of the I/O Unit. Its function is to monitor the (independently performed) input/output operations of the I/O Unit in accordance with what is specified in the Instruction Register. This involves activating the I/O mechanism(s) in the I/O Unit as required so that the necessary data transmissions can occur between the I/O medium and the I/O Unit's Buffer Memory; it also involves synchronizing the I/O Unit Buffer Memory with the computer so that data transmissions to and from the I/O Unit's associated I/O Track and Buffer Memory can take place independently of the Central Computer.

As noted above, Sequence Control also generates the signals which operate the Ready flip-flop in the Demand Station, and clear the Instruction Register as required.

Note: An important common feature of all I/O Units is that they have a definite cycle for each operation. If they are given an operation to perform, they will independently carry that operation to completion (i.e., achieve the required data transmission to or from the computer, completely apart from the computer) and then automatically stop. Some units must stop before they can be given another operation; others can be given a continuous mode of operation by appropriately timing the I/O Instructions sent the I/O Unit.

5. I/O Unit Buffer Memory, Translator Format Control, and/or I/O Track Addressing System

Each I/O Unit has a buffer memory which functions as an intermediate storage for the data transmitted to and/or from the I/O Unit's associated I/O Track and its I/O medium. These buffer memories are 120-character memories in most I/O Units. Some I/O Units, however, employ smaller buffers; in some, only a 1-character buffer is used.

All data transmissions to and from an I/O Unit's Buffer Memory and its associated I/O Track involve Univac coded characters only. That is, most I/O Units have a translator associated with their buffer memory* so that data in one or more codes can be manipulated between the I/O Unit's buffer memory and its I/O medium, but only Univac coded characters are exchanged by the I/O Unit's associated I/O Track and Buffer Memory. No conversion routines are, therefore, necessary in the Central Computer program. Likewise, some I/O Units include programmable format control and I/O track addressing. Data in and out of the computer can thus be arranged as required, and obtained from or placed in a specific location on the I/O Track.

* In those I/O Units where no Translator is included, Univac code only is employed on the I/O Medium.

VI GENERAL STORAGE DATA TRANSMISSIONS

A. GENERAL STORAGE SYSTEM

The General Storage System is the random-access, large-capacity external memory of the UFC Model 1 Central Computer. It is composed of the following principal parts:

from 1 up to 10 magnetic drums, called General Storage Drums, GSD, each of which is capable of storing 180,000 7-bit alphanumeric characters;

the General Storage Address Register, GSAR, a 7-digit register that holds the address of the General Storage location involved in a General Storage Reference;

the General Storage Buffer, GSB, a 120-character register used as an intermediate storage for data transmissions to and from the General Storage Drums; also used to hold the Unit Record Identifier in Channel Search operations;

the control, locating, and synchronization circuitry necessary for carrying out the following operations:

Clear GSB to Ignores
Write Unit Record
Write Unit Record & Check
Read Unit Record
Channel Search =
Channel Search $\neq$

The General Storage System's unit of data is the Unit Record. That is, Unit Records are read, stored, and searched for. A Unit Record is a collection of continguous characters which can be a complete file-entry (major unit of a file) or merely a portion thereof. Unit Records are always at least one computer word (12 characters) long, and are always an integral multiple of 12 characters in length (120 characters maximum). The Unit Record Length for each operation is programmed.

The programmer determines how the files in General Storage are organized, and whether a Unit Record in a given General Storage operation represents a complete file-entry or merely a portion thereof. Usually, when files are initially recorded in General Storage, the recording programs are coded so that these programs use the maximum Unit Record Length that will ever be required in storage references to each file. Subsequent General Storage operations, depending on the Unit Record Length programmed for each, will process a complete file-entry (i.e., a Unit Record whose length is the same as that used in the original recording of the file), or they will process only a portion of a complete file-entry. Thus, in addition to its random-access and large capacity features, General Storage also has the programmable property that the data it stores is both selectively alterable and selectively obtainable on a Unit Record basis.

Another important property of the General Storage System is that its operations can be time-shared with those of the Central Computer and each I/O Unit. However, the following should be noted:

> Since GSB and GSAR form part of the General Storage System during operations of that system, they are not available to Program Control Storage when engaged in any General Storage operations. If a Program Control Storage Reference is attempted to GSAR or GSB while these registers are "locked-out", the computer waits* until the previously initiated General Storage operation is finished, before completing its storage reference.

B. GENERAL STORAGE DRUM

Each General Storage Drum, GSD, is a rotating cylinder which is coated with a magnetizable surface. Data is transferred to and from this surface by means of dual-purpose magnetic read/write heads which are mounted in the housing of the drum. The mechanics with which data is read or stored on these drums is basically the same as that described for the High Speed Drum (See Section III). However, as illustrated in Figure VI-1, Page VI-3, a different organization of data is employed:

> each General Storage Drum is divided into three sections, 100 channels (or tracks) per section;
>
> each channel, in turn, is divided into 100 6-character groups (these are used for Unit Record data); and
>
> each channel has an additional Search Control Location at which 6 characters of control data are stored for use in Channel Search operations. (A channel's Search Control Location is the last, i.e., the 101st, 6-character group read each drum revolution.)

There are thus two kinds of locations on a General Storage Drum: Unit Record Areas (URA) and Search Control Locations (SCL).

---

* "Lock-out" problems with GSB and GSAR are easily by-passed, by suitable programming, when optimum operation of the Central Computer is required. For example, the Channel Search Probe Instruction or Sub-Step can be employed to test whether or not Channel Search operations are completed. Computing can resume without delay if the Channel Search is not completed, and another test made later in the program. The maximum times (given later in this section) for Read Unit Record, Write Unit Record, and Write UR and Check operations can be assumed, and computing carried out up to these maximum times between Program Control Storage References to GSAR and GSB. No delay is then introduced into Central Computer operations.

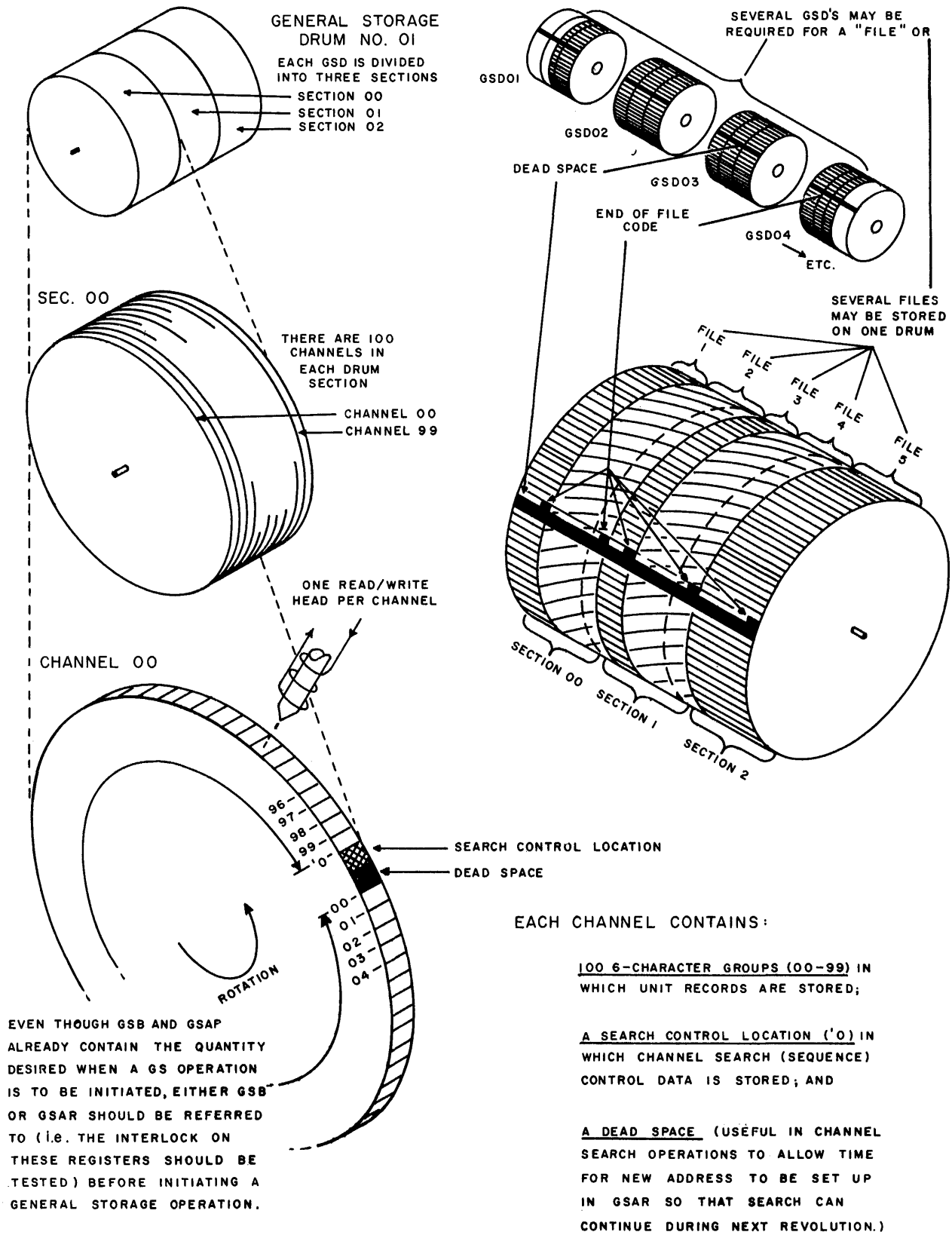


Figure VI-1. Organization of Data on the General Storage Drums

C. GENERAL STORAGE ADDRESS STRUCTURE

Two types of General Storage Addresses are employed: URA Addresses and SCL Addresses.

URA Addresses specify the location of a Unit Record Area. To define the location of a URA: a drum section, a channel within that section, the starting angular position of the URA around that channel, and the number of computer words (i.e. pairs of 6-character groups) in the location must be specified. The following seven-character format is used:

L	DS	CH	AA
Unit Record Length	Drum Section	Channel Number	Starting Angular Address

All characters in a URA Address are numeric. An explanation of the range of values for each portion of a URA Address is given in Table VI-1, Page VI-5. An example of how the General Storage System interprets a typical URA Address is given in Figure VI-2, Page VI-8.

SCL Addresses specify the location of a particular channel's Search Control Location. The following format is employed:

—	DS	CH	'0
Unit Record Length is Ignored	Drum Section	Channel Number	Search Control Location
	Channel's Address		

DSCH must be numeric; ' (prime) must be the second lowest-order character, 0 (zero) the lowest-order; and the highest order character can be any computer character. An explanation of the range of values for each portion of an SCL Address is also given in Table VI-1, Page VI-5. An example of how the General Storage System interprets a typical SCL Address is given in Figure VI-3, Page VI-9.

TABLE VI-1. EXPLANATION OF THE FORMAT OF GENERAL STORAGE ADDRESSES

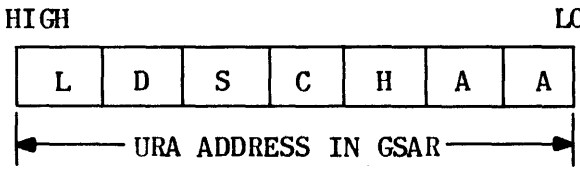
UNIT RECORD AREA ADDRESSES: LDSCHAA	SEARCH CONTROL LOCATION ADDRESSES - DSCH'0
L is the Unit Record Length (number of words in the URA) L = a digit (1,2,3,...9,0) which, if the UNIT RECORD LENGTH SELECTOR is set to GSAR, specifies the number of computer words (i.e. pairs of 6-character groups) in the Unit Record. (0 means ten words in the Unit Record.) Unless the UNIT RECORD LENGTH SELECTOR is set to GSAR, L is ignored, and the Unit Record Length is defined by one of the other ten settings (1,2,3,...9,0) of the UNIT RECORD LENGTH SELECTOR. When the Unit Record Length is defined by the settings (1,2,3,...9,0) these positions have the same meaning as the values of L; e.g., if the UNIT RECORD LENGTH SELECTOR is set to 1, the Unit Record Length is one computer word (12 characters) etc. The Unit Record Length specified by L or by the setting (1,2,3,...9,0) of the UNIT RECORD LENGTH SELECTOR is constant for each operation. (NB: Channel Search Operations)	- = any valid computer character. When a channel's Search Control Location is specified, no Unit Records are involved; hence, the highest-order character of an SCL Address is ignored by the General Storage System.
DS is the Drum Section that contains the URA DS = 00 up to 3n-1, in general; 00-29, maximum range. If n is the number of drums employed in an installation, there are 3n sections, and DS has the following range of correct values: 00-(3n-1). For example, if 10 drums are used, DS can validly be any number in the range 00-29. In such an installation, if DS is given a programmed value of 45, a PROGRAM (Inactive Drum Section) error occurs: the General Storage operation is not performed, and the computer hangs up when it begins the execution of the next instruction.	DS is the Drum Section that contains the SCL DS = 00 up to 3n-1, in general; 00-29, maximum range. Same rule concerning Inactive Drum Section error applies to SCL addresses as for URA addresses.
CH is the channel on which the URA is located CH = 00-99	CH is the channel whose SCL is desired CH = 00-99
AA is the starting angular address of the URA AA = an even number (00-98) indicating the starting angular address of the Unit Record Area on channel DSCH. If AA = an odd number, a PROGRAM error (Odd Angular Address) occurs: the General Storage operation is not performed, and the computer hangs up at the beginning of the execution of the next instruction. If AA and L have values which would extend a General Storage operation beyond the end of the last possible complete Unit Record Area on a channel, a UNIT RECORD LENGTH error occurs: the General Storage operation terminates (Search Control memory location is protected), and the computer hangs up at the beginning of the execution of the next instruction. Eg. it would be impossible to read or write a complete Unit Record, if L = 6 (UR = 72 characters) and AA = 94. (See Table VI-2, next page, for Unit Record Length information.)	'0 is the channel's SCL

Table VI-2. Unit Record Length Information

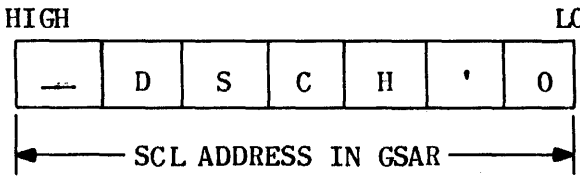
UNIT RECORD LENGTH SELECTOR SWITCH SET TO GSAR: L =	UNIT RECORD LENGTH SELECTOR SWITCH SET TO:	NUMBER OF CHARACTERS PER UNIT RECORD	NUMBER OF COMPLETE UNIT RECORDS PER CHANNEL	LAST UNIT RECORD AREA ON EACH CHANNEL INCLUDES ANGULAR ADDRESSES	NUMBER OF UNIT RECORDS PER DRUM
1	1	12	50	98-99	15000
2	2	24	25	96-99	7500
3	3	36	16	90-95	4800
4	4	48	12	88-95	3600
5	5	60	10	90-99	3000
6	6	72	8	84-95	2400
7	7	84	7	84-97	2100
8	8	96	6	80-95	1800
9	9	108	5	72-89	1500
0	0	120	5	80-99	1500

D. GENERAL STORAGE ADDRESS REGISTER

The General Storage Address Register, GSAR, is a 7-digit register whose principal function is to hold General Storage Addresses during General Storage operations.



(See Figure VI-2)
Page VI-8



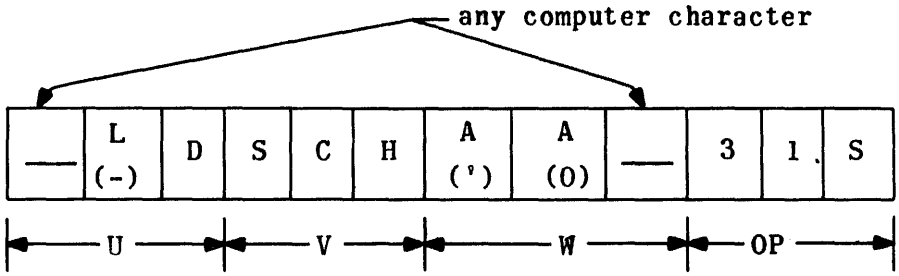
(See Figure VI-3)
Page VI-9

GSAR is also a 7-digit Program Control Storage Location. (Program Control Storage Address: 995)

In internally-defined programs, the address at which a General Storage operation is to be initiated is sent to GSAR in one of two ways:

> the Load GSAR Instruction Word (LA) is executed; or
> GSAR is addressed as a Destination in an Instruction Word.

When the Load GSAR Instruction Word is executed, the address sent to GSAR is derived from the Instruction Word itself:



When GSAR is addressed as a Destination in an Instruction Word, the address it receives is obtained from a Program Control Storage Location via a Program Control Storage Reference.

In plugboard-defined programs, the address at which a General Storage operation is to be initiated can be sent to GSAR only by addressing GSAR as a Destination in a Program Step.

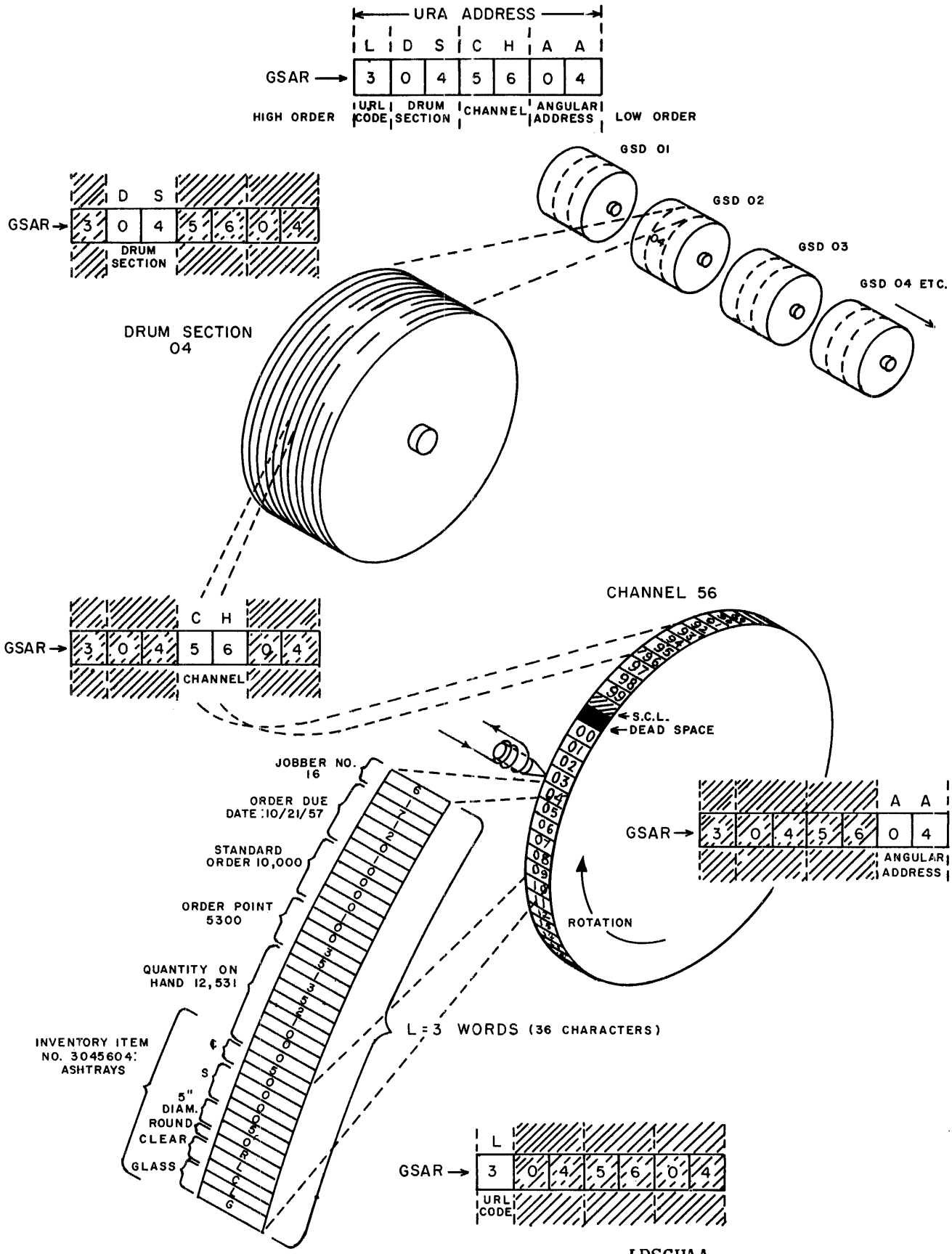


Figure VI-2. Unit Record Area, Address LDSCHAA 3045604

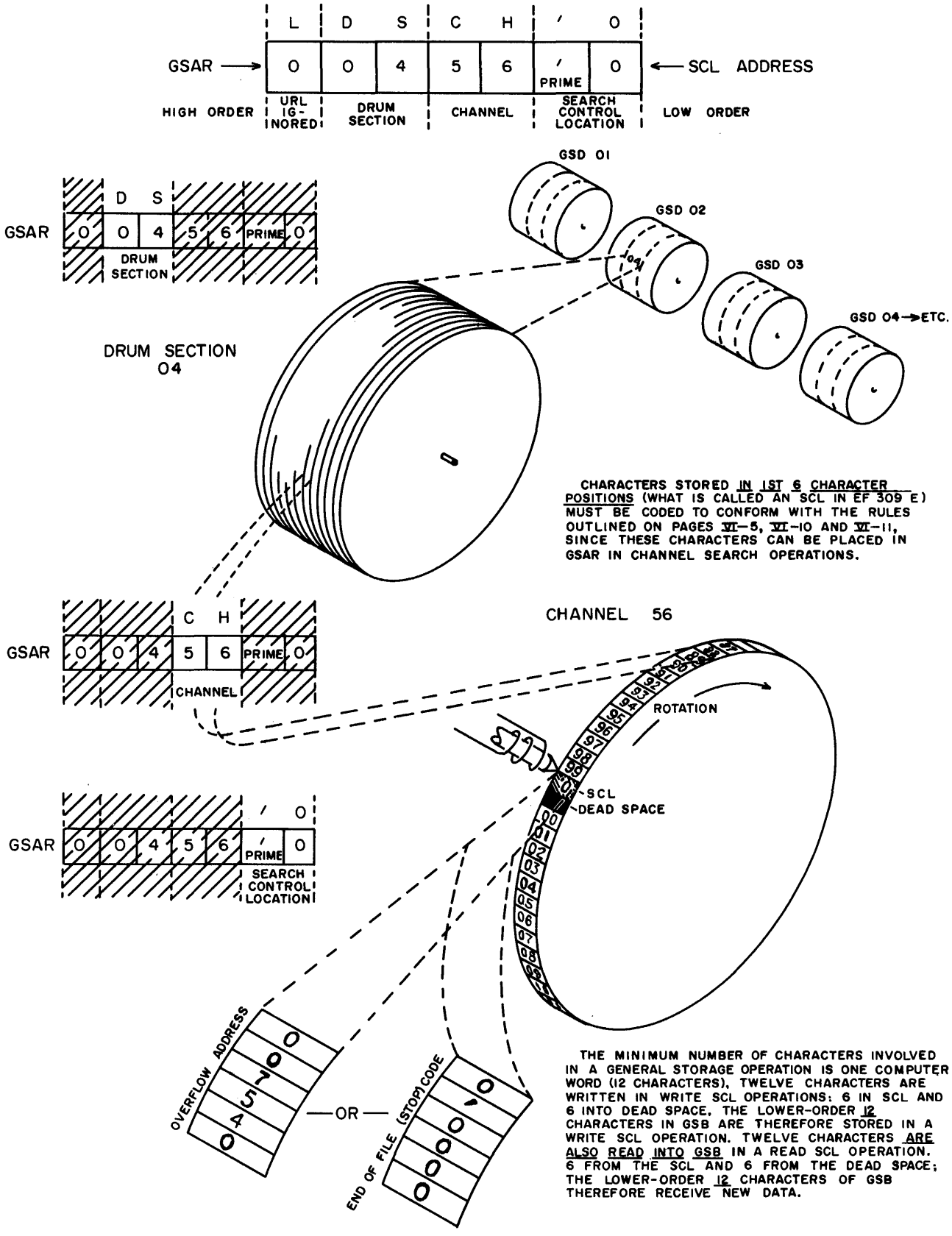


Figure VI-3. Search Control Location, Address $\begin{matrix}-\text{DSCH'O}\\-0456\text{'O}\end{matrix}$

The initial contents of GSAR in a General Storage operation can specify:

a Unit Record Area in General Storage at which a Unit Record is to be stored or read; or

a Unit Record Area in General Storage from which a Channel Search operation is to begin; or

a particular channel's Search Control Location at which special Search Control data, 6 characters, can be stored or read.

GSAR can also be set by the General Storage System during Channel Search operations as described on Page VI-11 and 12 Paragraph D2.

1. Rules For Addresses Sent GSAR by the Computer Program (See Also Table VI-1, Page VI-5)

a. URA Addresses: When Unit Record Area Addresses are formed in GSAR via programming, the following must be noted:

(1) The lower-order 6 characters (DSCHAA) must always be numeric, (0-9). If they are not, two kinds of errors can occur depending on what alpha characters are sent to the DSCHAA positions in GSAR:

(a) If the following alpha characters (zone bits = 00) appear in GSAR

i = Ignore
△ = Space
- = Hyphen
' = Apostrophe (Prime)*
& = And Sign, or
(= Open Parentheses

The General Storage Address is invalid, and the General Storage operation is not performed. The computer detects this when it again attempts a General Storage operation and hangs up, because it finds GSB and GSAR "locked-out".

(b) The zone bits of all other non-numeric characters are not equal to 00. If any of these characters are sent to the GSAR positions DSCHAA, a PROGRAM (α-zone) error occurs: the General Storage operation will not be performed, and the computer will hang up when it begins execution of the next instruction.

* If ' is combined with 0 in the AA positions of GSAR, a channel's Search Control Location (rather than a URA) is specified if DSCH is a valid channel address.

(2) The highest-order character, L:

if the UNIT RECORD LENGTH SELECTOR is set to GSAR, L must be a digit (0-9); if L is an alpha character, (1)(a) or (1)(b) above applies depending on which alpha character is sent to the L position in GSAR;

if the UNIT RECORD LENGTH SELECTOR is set to one of its other ten positions (1,2,3...9,0), L can be any legal computer character (since L is ignored).

b. SCL Addresses: When the address of a channel's Search Control Location is formed in GSAR via programming, the following must be noted:

(1) Prime zero ('0) must be sent to GSAR's next-to-the-lowest and lowest-order character positions, respectively. If '0 is not sent, a number of things can happen depending on what is sent. For example, a Unit Record Area on a channel rather than the channel's Search Control Location might be referred to; or cases (1)(a) or (1)(b) above might apply; or one of the errors listed in Table VI-1, Page VI-5, would occur.

(2) The DSCH positions in GSAR must specify the particular channel whose Search Control Location is to be found. (If an End of File code were placed in the wrong channel's Search Control Location, for example, the "file" actually searched would contain either more or less Unit Records than was intended.)

(3) The highest-order character in GSAR, regardless of how the UNIT RECORD LENGTH SELECTOR is set, can be any computer character. (When '0 is detected in GSAR's lower-order two positions, the highest-order position of GSAR is ignored.)

2. GSAR Set by General Storage in a Channel Search Operation

In Read UR, Write UR, and Write UR and Check operations, the address held in GSAR specifies a particular URA or SCL Location. The General Storage Locating Circuitry translates this address, finds the location specified, and the desired operation is performed. No modification of GSAR occurs; hence, in these operations, the initial and final contents of GSAR are identical.

In Channel Search operations (See Paragraph G), however, GSAR is modified as follows:

a. Prior to a Channel Search operation, GSAR is loaded with the URA Address from which the search is to begin.

b. This location is found, and the Unit Record contained therein is read and compared, character-by-character, with the Unit Record Identifier.

c. Two characters before this first Unit Record has been completely read, the system begins loading the starting angular address (AA) of the next URA in a temporary storage called the Angular Address Register.

d. When the first Unit Record has been completely read (and compared with the Unit Record Identifier) one of two actions is taken with respect to GSAR:

(1) If a "find" occurred, GSAR is left unaltered (since it already contains the address of the URA in which the "find" occurred), and the Channel Search terminates.

(2) If no "find" occurred, the two characters in the Angular Address Register are set in GSAR's AA positions, and the Channel Search continues in the next URA.

e. The above procedure in (d) is repeated after each succeeding Unit Record on the Channel has been read.*

f. If no "find" occurs by the time the last Unit Record on the first channel is read, '0 (prime zero, the SCL angular address) is loaded into GSAR's AA positions.

g. The SCL is then read. If '0 (prime zero) is stored in the SCL's two lower-order character positions, the Channel Search terminates and GSAR is not modified (the first channel's SCL Address is retained in GSAR). If the SCL does not contain '0 as its two lower-order characters, the six characters it contains (overflow address) are loaded into GSAR's DSCHAA positions.

h. As the dead space on the drum passes under the read/write heads, the system's locating circuitry sets up for the Overflow Address, i.e., for the new address from which the search is to continue. The search continues as the new revolution begins.

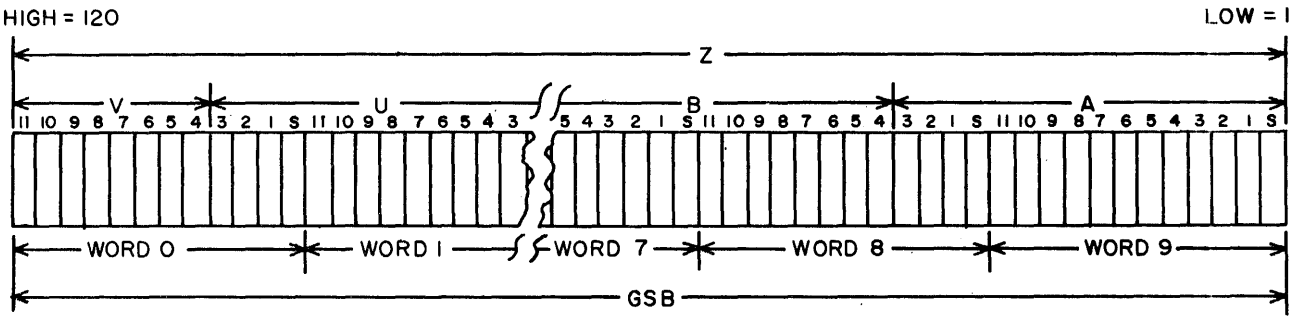
i. Steps (b) through (f) and (h) are repeated until an SCL is read that contains an End of File (Stop) code. When an End of File (Stop) code is detected in an SCL, the operation terminates and the contents of the SCL are not loaded into GSAR. Instead, as described in (g) above, the SCL Address of the channel whose SCL contains the End of File code is left in GSAR.

---

* If a "find" occurs, GSAR will be left set with the specific URA Address at which the "find" occurred.

E. GENERAL STORAGE BUFFER

The General Storage Buffer, GSB, is a 120-character magnetic core register whose principal function is that of a buffer or intermediate storage in the General Storage System during data transmissions to and from the General Storage Drums. GSB also functions to hold the Unit Record Identifier (discussed in Paragraph G below) in Channel Search operations. When not engaged in General Storage operations, GSB is a Program Control Storage Location that is Word, Field, and Blockette addressable. (Program Control Storage Address: 98x; where x = 0-9 in Word Addresses; x = A-V, excluding I and O, in Field Addresses; and x = Z in Blockette Addresses.)



Although GSB is Word, Field and Blockette addressable in Program Control Storage References, it is <u>not</u> in General Storage operations. Data always leaves or enters GSB <u>in General Storage operations</u> lowest-order character first, beginning with GSB's Word 9, Character S position. (Note the rules in Paragraph G relating to position of Unit Records and Unit Record Identifiers in GSB.)*

F. GENERAL STORAGE COMPARATOR

The General Storage Comparator is a group of circuits whose prime function is to compare a Unit Record with an equal number of correspondingly-significant characters in GSB. These circuits function in all Channel Search operations, and in the check-portion (second half) of all Write Unit Record and Check operations.

During Channel Search operations, the contents of each Unit Record Area searched is compared with the Unit Record Identifier (See Paragraph G5(b), Page VI-24) held in GSB. This comparison is made on a bit-by-bit basis, so that parity as well as match or mismatch can be determined. As illustrated in Figure VI-4, Page VI-15, the comparison is begun with the lowest-order character in the Unit Record and in the Unit Record Identifier; the next-lowest-order character of each is then compared; etc. and the highest-order character in each is the last compared. In Channel Search = operations, Ignore codes in the Unit Record Identifier suppress comparison for the character positions in which they occur. In Channel Search $\neq$ operations, Ignore codes in <u>either</u> the Unit Record <u>or</u> the Unit Record Identifier suppress comparison in the character positions in which they occur.

---

* Even though GSB and GSAR already contain the quantity desired when a General Storage operation is to be initiated, either GSB or GSAR should be referred to(i.e., the interlock on these registers should be tested) before initiating a General Storage operation.

During the Check-portion or second half of a Write Unit Record and Check operation, (See Figure VI-7, Page VI-21) the data just recorded is compared with the original data still retained in GSB (i.e., with the data that should have been recorded). As in the case of Channel Search operations, the comparison begins with the character in GSB's Word 9 Character S position and the lowest-order character of the data on the General Storage Drum. Ignore codes do not suppress comparison, however, in this operation of the General Storage Comparator.

The manner in which the General Storage Comparator reports the results of its comparisons is discussed under the Channel Search and Write Unit Record and Check sections of Paragraph G below.

G. GENERAL STORAGE OPERATIONS

1. Clear General Storage Buffer to Ignores.

This operation places Ignore codes in all 120 character positions in GSB. It can be initiated in General Storage (assuming no "lock-out" for GSB or GSAR) as follows:

> If the program is internally-defined, cause a Clear GSB To Ignores Sub-Instruction to be initiated by coding S = K in the OP of the appropriate Instruction Word; or
>
> If the program is plugboard-defined, by appropriate patch-cord-wiring cause the CLEAR GSB TO IGNORES hub on the Program Control Plugboard to be pulsed.

The time required for a Clear GSB to Ignores operation is 5 milliseconds.

2. Write Unit Record.

This operation is used for two purposes:

> To locate a specific Unit Record Area (whose address is held in GSAR) and to record the Unit Record held in GSB in that Unit Record Area; or
>
> To locate a particular channel's Search Control Location (when that channel's SCL Address is held in GSAR) and to record the lower-order 6 characters of GSB (an Overflow Address or an End of File code) in that channel's Search Control Location.

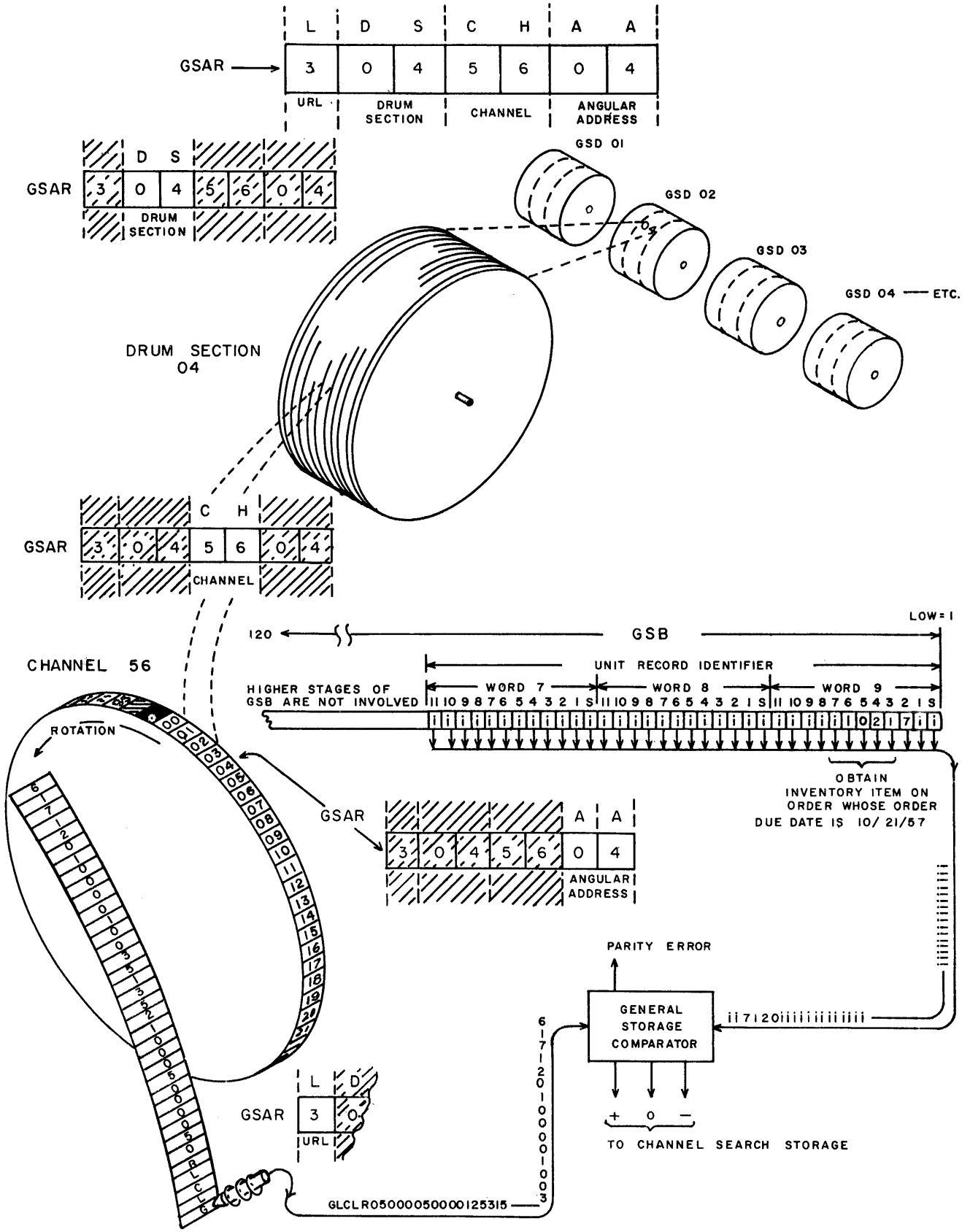


Figure VI-4. Operation of the General Storage Comparator During Channel Search =

Data sent from GSB to a General Storage Drum is sent lowest-order character first. GSB's Word 9, Character S supplies the first character transmitted; GSB's Word 9, Character 1 supplies the next, etc. When Unit Records are recorded, the number of characters transmitted is determined by the Unit Record Length specified for the operation. When this operation is used to load a channel's Search Control Location, exactly six characters are transmitted.

To use the Write Unit Record operation to record a Unit Record (assuming no "lock-out" exists for GSAR and GSB):

a. Load GSAR with that Unit Record's URA Address;

b. Load GSB with the Unit Record to be recorded, being careful to properly position the Unit Record in GSB (i.e., position the Unit Record so that the lowest-order character in the Unit Record is in GSB's Word 9, Character S position; the next-lowest-order character in the Unit Record is in GSB's Word 9, Character 1 position, etc.); and, then

c. If the program is internally defined, cause a Write UR Sub-Instruction to be initiated by coding S = M in the OP of the appropriate Instruction Word; or, if the program is plugboard-defined, by appropriate patchcord wiring cause the WRITE UR hub on the Program Control Plugboard to be pulsed.

When the operation in (c) is initiated, the location of the URA specified by the contents of GSAR is found and the Unit Record held in GSB is recorded in that location, lowest-order character first. The original contents of GSAR and GSB are unaltered by this operation.

Figure VI-5, Page VI-17 illustrates a typical Write Unit Record operation that records a 12-character Unit Record.

The maximum time (T_{max}) required to record a Unit Record of n words by a Write Unit Record operation is:

$$T_{max} = 34 + 0.65\ n \text{ (milliseconds)}$$

The average time (T_{avg}) required to record a Unit Record of n words by a Write Unit Record operation is:

$$T_{avg} = 17 + 0.65\ n \text{ (milliseconds)}$$

Note: The above times do not include the loading of GSB and GSAR.

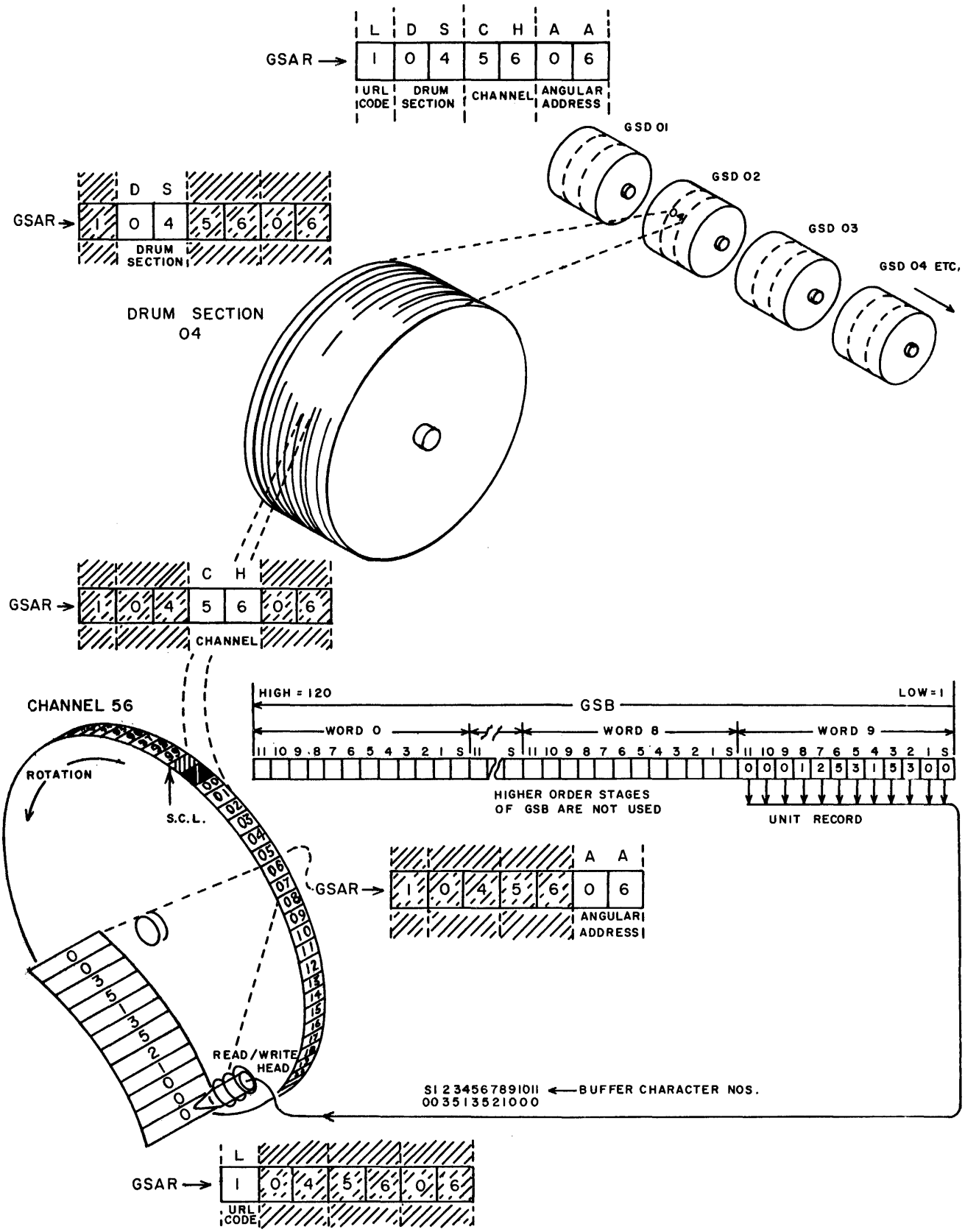


Figure VI-5. Write Unit Record Operation that records a 12-character Unit Record

To use the Write Unit Record operation to record an Overflow Address or End of File code in a channel's Search Control Location (assuming no "lock-out" exists for GSAR or GSB):

a. Load GSAR with the channel's SCL Address:

b. Load the lower-order six character positions of GSB with the appropriate

Overflow Address: DSCHAA (the new address from which the search is to continue when this channel's Search Control Location is read; this address can be the address of any URA on any General Storage Drum); or

End of File (Stop) code: xxxx '0 (where x is any computer character, and xxxx '0 is the code which stops Channel Search operations when this channel's Search Control Location is read.) This code is called an End of File code because the programmer will usually be interested in searching complete files during Channel Search operations. Actually, it can be placed on any Channel's Search Control Location to do "partial" file searches if this is desired. In this sense it is a Stop code; then

c. Cause the program to appropriately execute a Write UR Sub-Instruction or Sub-Step as described above for recording Unit Records.

When the specified channel's Search Control Location is found, the Overflow Address or End of File code is recorded in that location, lowest-order character first. The original contents of GSAR and GSB are unaltered by this operation.

Figure VI-6, Page VI-19 illustrates two typical Write Unit Record operations that record an Overflow Address and an End of File (Stop) code in a channel's Search Control Location.

The maximum time (T_{max}) required to load a channel's Search Control Location by a Write Unit Record operation is:

T_{max} =34.65 milliseconds

The average time (T_{avg}) required to load a channel's Search Control Location by a Write Unit Record operation is:

T_{avg} =17.65 milliseconds

Note: These times do not include the loading of GSB and GSAR.

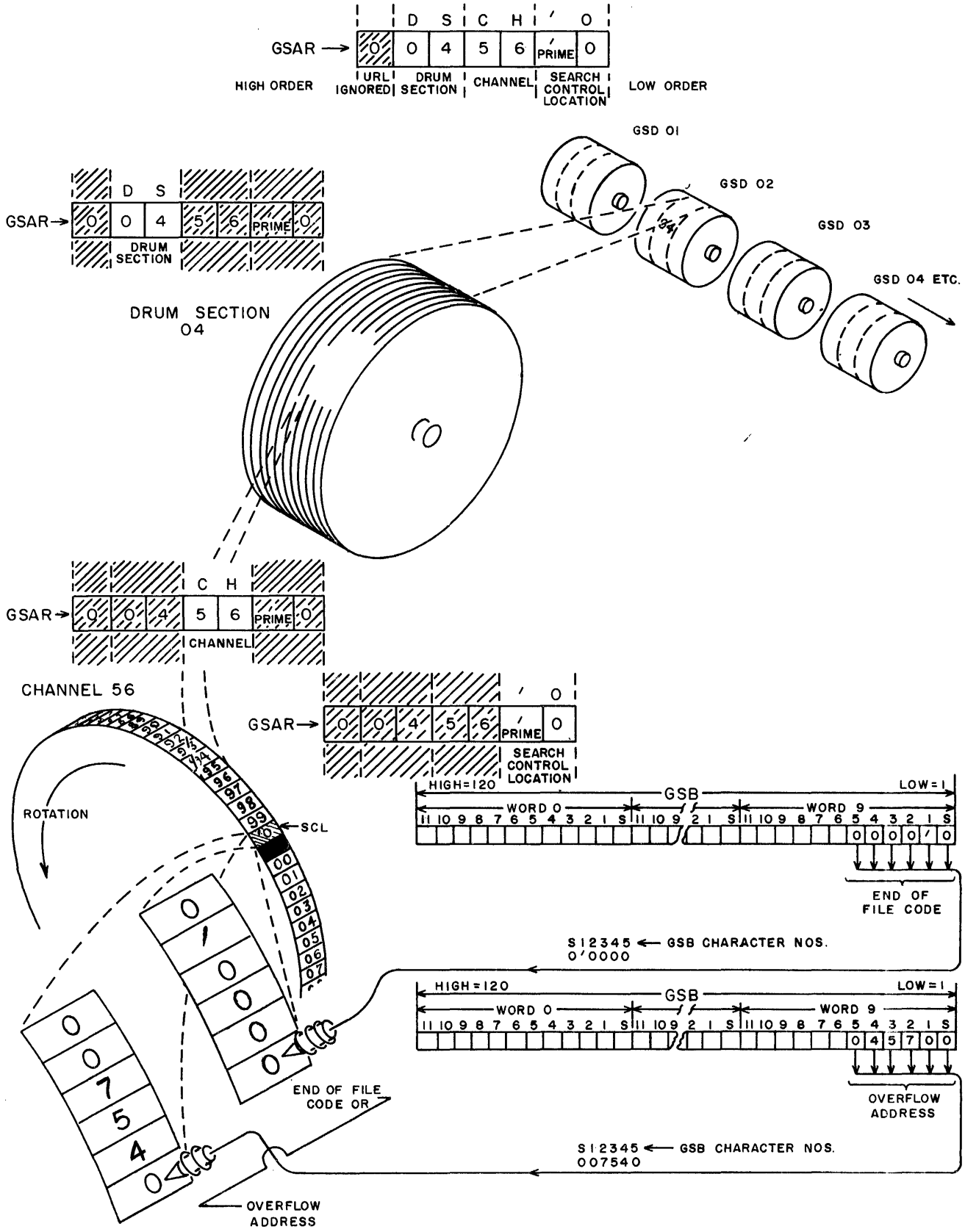


Figure VI-6. Write Unit Record Operations that record an Overflow Address and End of File Code in a channel's Search Control Location

See Notes - VI-9

3. Write Unit Record and Check

This operation can be used for the same purposes as the Write Unit Record operation, and the same procedures as outlined on Pages VI-16 and VI-18 are employed to initiate this operation, except that (c) reads:

> "If the program is internally defined, cause a Write UR & Check Sub-Instruction to be initiated by coding S=P in the OP of the appropriate Instruction Word; or if the program is plugboard-defined, by appropriate patch cord-wiring cause the WRITE UR & CHECK hub on the Program Control Plugboard to be pulsed."

Up to two drum revolutions (maximum) are required:

> during the first revolution, the data to be recorded is written; during the second revolution, that data is read back, and compared with the original data still retained in GSB. (See Figure VI-7, Page VI-21)

If the intended data was recorded correctly, the operation terminates in the normal manner.

If the intended data was not recorded correctly, a WRITE UR AND CHECK error occurs, and the computer hangs up when it begins execution of the next instruction.

The maximum time (T_{max}) required to record a Unit Record of n words by a Write Unit Record and Check operation is:

$$T_{max} = 68 + 0.65\ n \text{ (milliseconds)}$$

The average time (T_{avg}) required to record a Unit Record of n words by a Write Unit Record and Check operation is:

$$T_{avg} = 51 + 0.65\ n \text{ (milliseconds)}$$

Note: These times do not include the loading of GSB and GSAR.

The maximum time (T_{max}) required to record a channel's Search Control Location by a Write Unit Record and Check operation is:

$$T_{max} = 68.65 \text{ milliseconds}$$

The average time is $T_{avg} = 51.65$ milliseconds

Note: These times do not include the loading of GSB and GSAR.

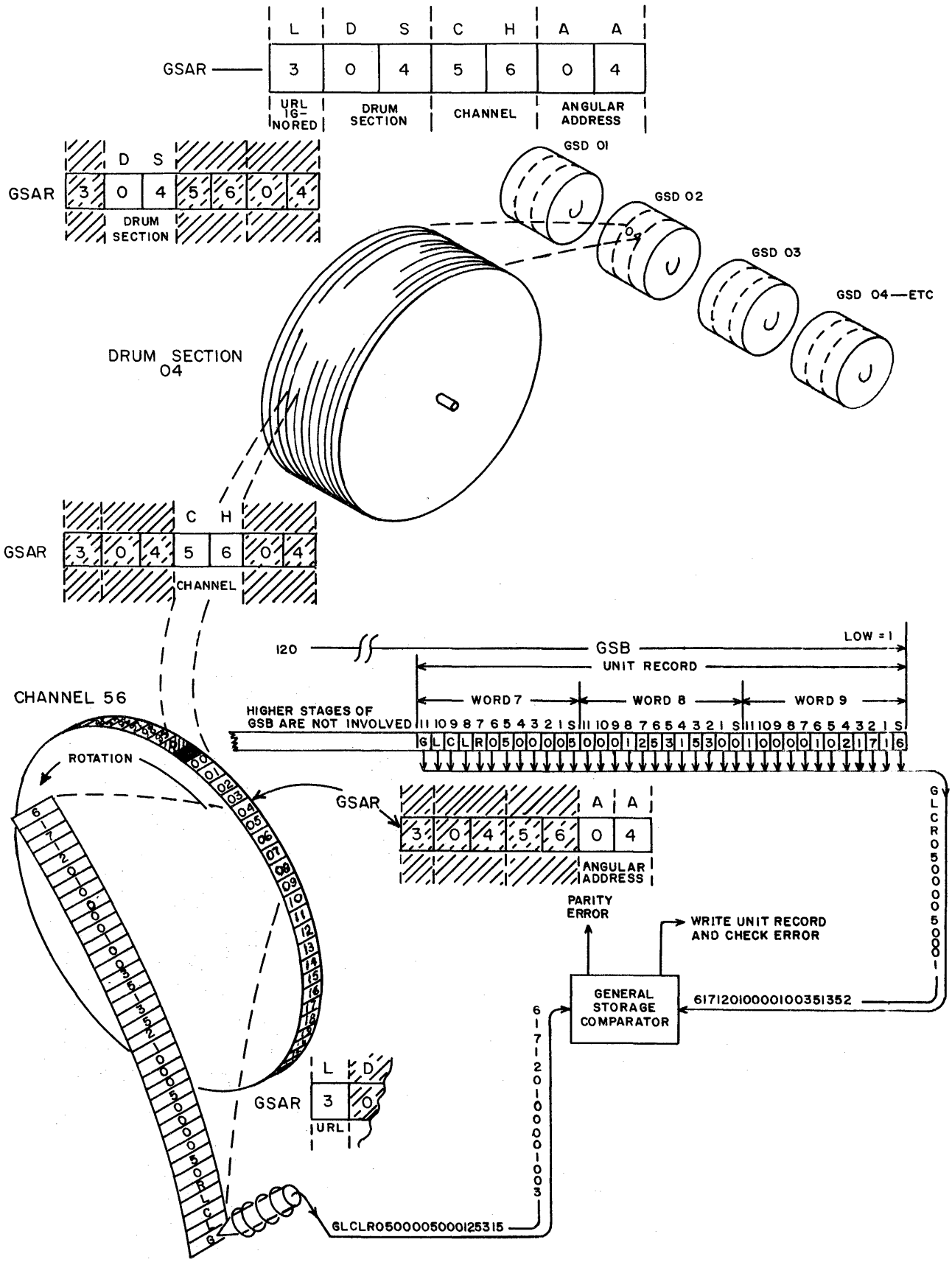


Figure VI-7. Operation of the General Storage Comparator during Write Unit Record and Check

4. Read Unit Record

This operation is used for two purposes:

To locate a specific Unit Record Area (whose URA Address is held in GSAR) and to transfer the Unit Record stored there to GSB; or

To locate a particular channel's Search Control Location (when that channel's SCL Address is held in GSAR), and to transfer the 6-characters stored in that location to GSB.

Data sent to GSB from a General Storage Drum is sent lowest-order character first. The first character received (i.e., the first read from the GSD) is placed in GSB's Word 9, Character S position, etc. When Unit Records are read, the number of characters transmitted is determined by the Unit Record Length specified for the operation. When a channel's Search Control Location is read, exactly six characters are transmitted.

To use the Read Unit Record operation to obtain a Unit Record from a General Storage Drum (assuming no "lock-out" exists for GSAR and GSB):

a. Load GSAR with the Unit Record's URA Address; and then,

b. If the program is internally-defined cause a Read UR Sub-Instruction to be initiated by coding S = L in the OP of the appropriate Instruction Word; or if the program is plugboard-defined, by appropriate patchcord-wiring cause the READ UR hub on the Program Control Plugboard to be pulsed.

When the URA whose location is specified by the address in GSAR is found, a copy of the Unit Record it contains is sent to GSB. Note particularly that the Unit Record is loaded into GSB lowest-order character first, beginning when GSB's Word 9, Character S position, and that the original data in the positions of GSB involved in this operation are destroyed. This operation does not affect the original contents of GSAR or the Unit Record Area read.

Figure VI-8, Page VI-23 illustrates a typical Read Unit Record operation that obtains a 12-character Unit Record from a General Storage Drum.

The maximum time (T_{max}) required to read a Unit Record of n words is:

$$T_{max} = 34 + 0.65n \text{ (milliseconds)}$$

The average time (T_{avg}) required to read a Unit Record of n words is:

$$T_{avg} = 17 + 0.65n \text{ (milliseconds)}$$

Note: These times assume that GSAR has already been loaded; and they do not include the Program Control Storage Reference time required to make the Unit Record available to Program Control from GSB.

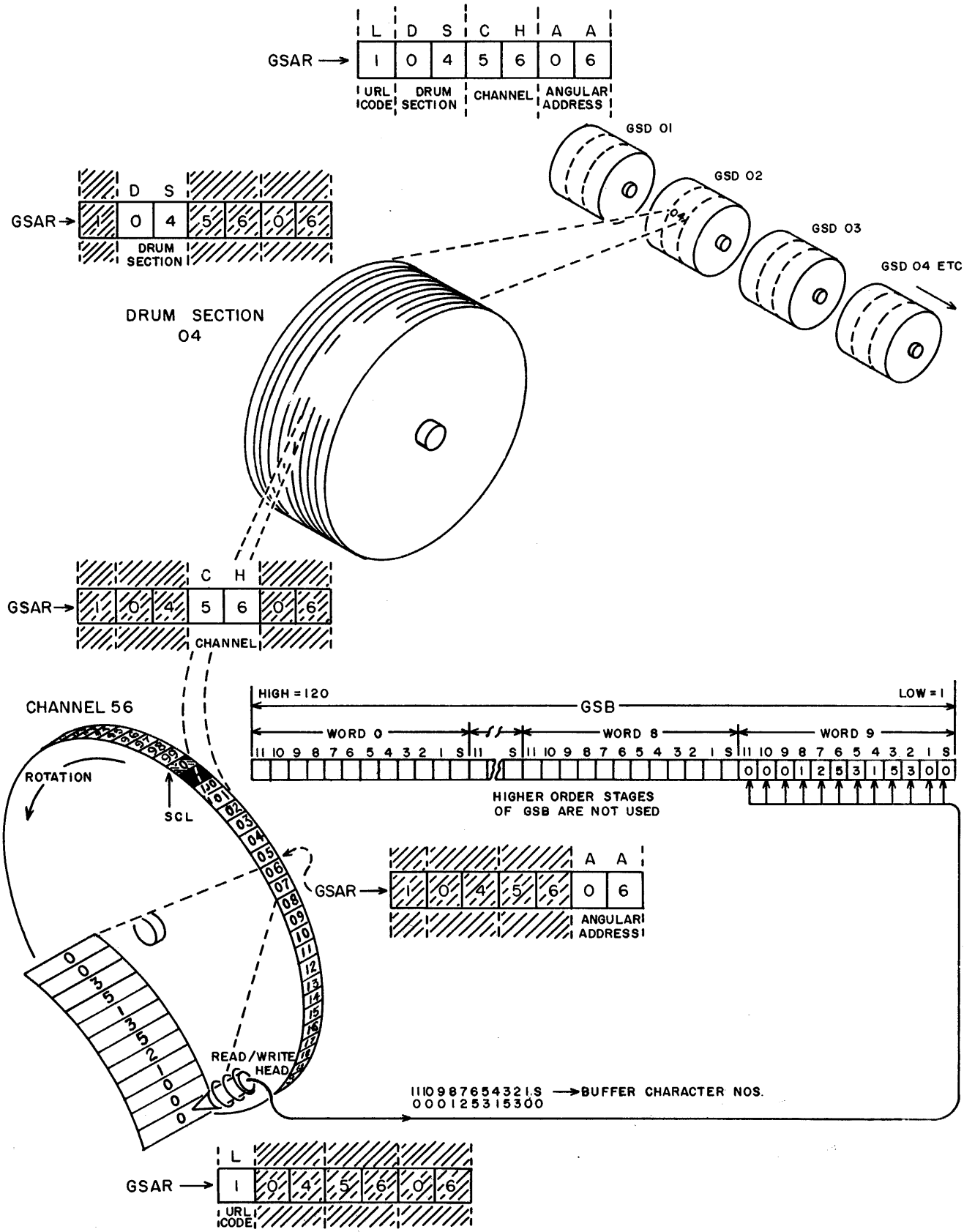


Figure VI-8. Read Unit Record Operation that obtains a 12-character Unit Record

To use the Read Unit Record operation to read a channel's Search Control Location (assuming no "lock-out" exists for GSAR and GSB):

a. Load GSAR with the channel's SCL Address; and, then

b. If the program is internally-defined, cause a Read UR Sub-Instruction to be initiated by coding S = L in the OP of the appropriate Instruction Word; or if the program is plugboard-defined, by appropriate patchcord-wiring cause the READ UR hub on the Program Control Plugboard to be pulsed.

When the specified channel's SCL is found, its contents are transmitted lowest-order character first to GSB, and loaded into GSB, beginning with GSB's Word 9, Character S position. The original data in the lower-order six character positions of GSB is destroyed. This operation does not affect the original contents of GSAR or the Search Control Location read.

Figure VI-9, Page VI-25 illustrates a typical Read Unit Record operation that obtains the contents of a channel's Search Control Location.

The maximum time (T_{max}) required to read a channel's Search Control Location is:

T_{max} =34.65 milliseconds

The average time (T_{avg}) required to read a channel's Search Control Location is:

T_{avg} =17.65 milliseconds

Note: These times assume that GSAR has already been loaded; and they do not include the Program Control Storage Reference time required to make the 6 characters of Search Control data available to Program Control.

5. Channel Search = .

This operation is used to search a file in General Storage for a Unit Record whose address is unknown. To properly initiate this operation (assuming that no "lock-out" exists for GSAR or GSB):

a. Load the Search Control Location of every channel in the file with the appropriate Search (sequence) Control data: Overflow Address or End of File (Stop) code. This can be done by either a Write UR or a Write UR and Check operation, described previously.

b. Load GSB with the set of data, called a Unit Record Identifier, which is to be compared with each Unit Record read. This data must be loaded into GSB in the following manner:

lowest-order character of Unit Record Identifier in GSB's Word 9, Character S position; next-lowest-order character of Unit Record Identifier in GSB's Word 9, Character 1 position, etc.

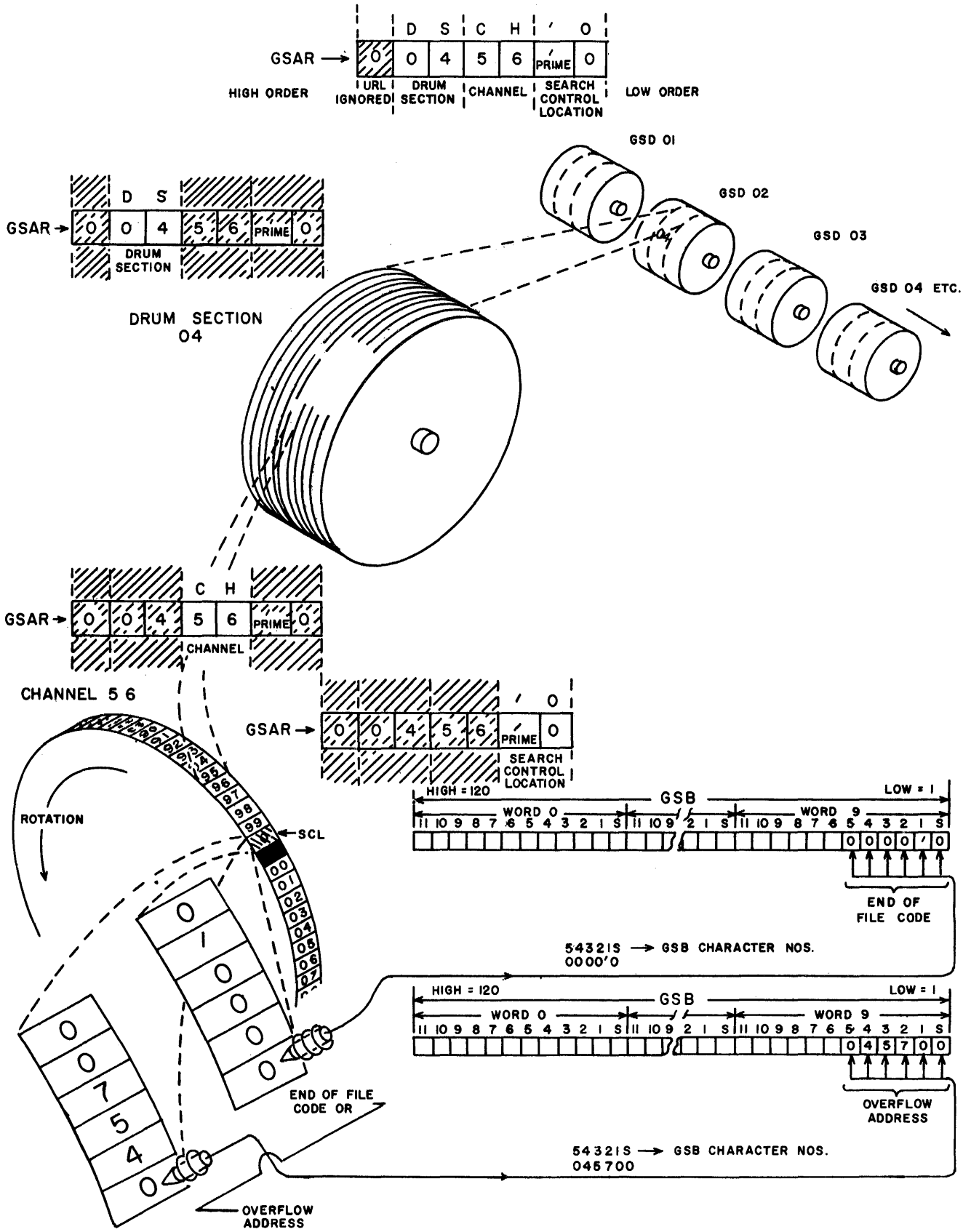


Figure VI-9. Read Unit Record Operations that obtain the contents of a channel's Search Control Location

See Notes VI-9

the length of the Unit Record Identifier must be the same as the Unit Record length employed for the Channel Search; and Ignore codes should be placed in all character positions of the Unit Record Identifier that are not to be compared.

NB: If GSB is found to contain all Ignore codes during a Channel Search, a PROGRAM (Unit Record Identifier all Ignores) error occurs: the Channel Search terminates after comparing one Unit Record, and the computer hangs up at the beginning of the execution of the next instruction.

c. Load GSAR with the URA Address from which the Channel Search is to begin;

NB: The Unit Record Length specified by the L position in GSAR is used throughout the Channel Search. Note that, depending on the setting of the Unit Record Length Selector switch (See page VI-6) the Character in the L position of GSAR can be determined by the switch setting or by the L Character in the address sent to GSAR.

d. If the program is internally defined, cause a Channel Search= Sub-Instruction to be initiated by coding S = N in the OP of the appropriate Instruction Work; or if the program is plugboard-defined, by appropriate patchcord-wiring.

The above procedure is probably not required, as listed, each time a Channel Search = operation is initiated by 5 (d), since (a), (b), and/or (c) may already have been taken care of by previous operations. The complete procedure is listed above to emphasize that (a), (b), and (c) must be taken care of before (d) is attempted.

Each Unit Record in succession, beginning with the starting URA Address, will then be read and each of its characters compared with a correspondingly significant character in the Unit Record Identifier held in GSB. (Ignore codes in the Unit Record Identifier suppress comparison in the character positions in which they occur.)

If a "find", (or exact match of every set of characters compared) occurs:

(1) this result, called PLUS, is set-up in a special-purpose memory (in the Central Computer) called Channel Search Storage;

(2) The Unit Record "found" and its address are captured during the next drum revolution after the find occurs: the Unit Record is sent to GSB, lowest-order character first, and stored in GSB beginning with GSB's Word 9, Character S position; and the URA Address is set-up in GSAR; then

(3) the operation terminates.

If no "find" occurs before the Search Control Location (of the channel on which the search began) is read, and that Search Control location contains a valid Overflow Address:

(1) that Overflow Address is automatically placed in GSAR; and

(2) the search is automatically continued from this new URA Address.

If no "find" occurs before the Search Location of the next channel is read and that channel's Search Control Location also contains an Overflow Address (1) and (2), of this paragraph, are repeated.

If a Unit Record is read which has Ignore Codes in every character position compared (i.e., in every character position for which the Unit Record Identifier has a non-Ignore code):

(1) this result, called ZERO, is set up in Channel Search Storage;

(2) the Unit Record that contains these "empty" areas and its address are captured during the next Drum Revolution after the find occurs: the Unit Record is sent to GSB, lowest-order character first, and stored in GSB beginning with GSB's Word 9, Character S position; and the URA Address of this Unit Record is set-up in GSAR;

(3) the operation then terminates.

If neither type of "find" (exact match or empty area) is accomplished by the time the last channel in the file has been searched, the following occur when the contents of that channel's Search Location (End of File) are read:

(1) the result (no "find"), called MINUS, is set up in Channel Search Storage;

(2) the Unit Record Identifier is left unaltered in GSB;

*(3) GSAR is set to the contents of SCL Address of the channel whose Search Control Location contained the End of File code; and

(4) the operation terminates.

---

* This feature has been incorporated to provide for storage of an additonnal 4 characters of control data in the channel's SCL if this is helpful in more efficiently sequencing Channel Search operations. If these 4 characters are not needed for such special control purposes, DSCH (the channel's address) can be stored in these four character positions when the End of File or Stop code is written in the channel's SCL. Only if this is done will (3) as shown on Page VI-27 be correct. The programmer thus can, when he codes the loading of SCL's used for stops, determine what is placed in GSAR when a Search results in a "no find."

6. Channel Search ≠.

This operation is also used to search a file in General Storage for a Unit Record whose address is unknown. However, the search is made on the basis of inequality; i.e., a Unit Record is desired which will not match a Unit Record Identifier in at least one pair of character positions compared.

The procedure to follow in initiating this operation is the same as that listed in 5(a) through 5(d) above for Channel Search = operations, except that 5(d) reads:

> "If the program is internally-defined, cause a Channel Search ≠ Sub-Instruction to be initiated by coding S = 0 in the OP of the appropriate Instruction Word; or if the program is plugboard-defined, by appropriate patchcord-wiring cause the CS ≠ hub on the Program Control Plugboard to be pulsed."

Each Unit Record, in succession, beginning with the starting URA Address will then be read and each of its characters compared with a correspondingly significant character in the Unit Record Identifier held in GSB. (Ignore codes in either the Unit Record or the Unit Record Identifier suppress comparison in the character positions in which they occur.)

If a "find" (or a mismatch of at least one pair of characters compared) occurs:

(1) this result, called PLUS, is set up in Channel Search storage;

(2) the Unit Record "found" and its address are captured: the Unit Record is sent to GSB lowest-order character first and is stored in GSB, beginning with GSB's Word 9, Character S position; and the URA Address is set up in GSAR;

(3) the operation then terminates.

Overflow Addresses sequence Channel Search ≠ operations in the same manner as in Channel Search = operations.

However, since Ignore codes in the Unit Record also suppress comparison in Channel Search ≠ operations, Channel Search Storage cannot be set to ZERO in these operations.

If no "find" takes place by the time the last channel in the file has been searched, the following occur when the contents of that channel's Search Control Location (End of File) are read:

(1) the result (no "find"), called MINUS, is set up in Channel Search Storage;

(2) the Unit Record Identifier is left unaltered in GSB;

(3) GSAR is set to the SCL Address of the channel whose Search Control Location contained the End of File code; and

(4) the operation terminates.

Testing Channel Search Storage:

In internally-defined programs the Channel Search Probe Instruction Word (27S) can be executed to test whether or not a previously initiated Channel Search operation is completed. If the search is still in progress, the computer program will resume without delay, and the next Instruction Word will be obtained from the location specified by the U-address of the Channel Search Probe Instruction Word. If the search has been completed, the next Instruction Word is taken from the location specified either by the V or W addresses of the Channel Search Probe Instruction Word, or by the contents of PAK as explained below.

If Channel Search Storage is set to:	The address which specifies the location of the next Instruction Word is:
MINUS	the V-section of the Channel Search Probe Instruction Word
ZERO	the W-section of the Channel Search Probe Instruction Word
PLUS	(PAK)

In plugboard-defined programs, two types of test for Channel Search completions can be made:

Channel Search Probe, and
Channel Search Probe and Wait.

Two sets of correspondingly named hubs are provided on the Program Control Plugboard: CS PROBE, CS PROBE AND WAIT.

When one of the four CS PROBE hubs is pulsed, the corresponding ACTIVE hub emits if the Channel Search operation is still in progress. If the Channel Search operation is completed, one of the other hubs, +, -, or 0, emits depending on how Channel Search Storage was set during the operation.

When one of the four CS PROBE AND WAIT hubs is pulsed, the computer waits if the Channel Search operation is still in progress. When the Channel Search has been completed, the associated +, -, or 0 hubs emits, depending on how Channel Search Storage was set during the operation. Patchcord-wiring is employed to determine when these tests are made and what program variance each result (+, -, or 0) is to achieve.

NB: The Channel Search Probe Instruction Word and Sub Steps merely test Channel Search Storage. The programmer must keep track of which kind of search operation set Channel Search Storage.

APPENDIX A

GLOSSARY OF TERMS

FOR THE

UFC-1

A

Term	Symbol	Explanation
access time		The interval between the time storage control initiates a storage reference and the time that storage reference is completed. The timing listed on Pages II-222, 223, 229, and 230 in Section II are access times for the operating memory of the Central Computer. The times listed on Pages VI-18 and VI-22 are the access times for the General Storage System.
ACTIVE		A condition of the General Storage System which is detected when a Channel Search Probe is performed. General Storage ACTIVE means that a previously initiated Search operation has not yet been completed.
addend		A number or quantity to be added to another (called an augend). In the UFC, V_2 in an add operation is the addend.
adder		The circuits which form the algebraic sum of two quantities. The UFC adder is also called an adder-subtractor. It is a serial device; i.e., (algebraic) addition is performed on a character-by-character basis, one set of characters at a time.
address		A label, name or number identifying a memory location at which information can be stored, and from which information can be obtained. Two types are employed in UFC: General Storage Address Program Control Storage Address
algorithm		A step-by-step presentation of the principal parts of a process or sequence of events. Used principally in connection with arithmetic operations, as Multiply Algorithm, etc.

Term	Symbol	Explanation
alpha		In general, a quantity consisting only of letters and/or other non-numeric symbols; also used as abbreviation for alphabetic characters. In add or subtract operations, alpha has a more particular meaning. See Paragraph IID, Section II, Rules for Addition.
alpha-numeric		A quantity containing letters and/or other (non-numeric) symbols in addition to numbers.
Alternate Switch	AS	A manually operated Selector
analog computer		A computer which calculates on continuous functions using physical analogs as variables. Usually a one-to-one correspondence exists between each numerical value occurring in the problem and a varying physical measurement in the analog computer.
angular position		A term used in connection with magnetic drums. It refers to a discrete location around the periphery of a track. (A track number and an angular position number define a unique location on a magnetic drum.)
arithmetic operation		An operation involving addition, subtraction, multiplication or division.
Arithmetic Transfer	AT	A Computer Instruction used for programmed data transfers of 12 characters or less. (Register D of the Arithmetic Section is used as a buffer between the Source and Destination.)
augend		A number or quantity to which another (called an addend) is added. In the UFC, V_1 in an add operation is the augend.

B

Term	Symbol	Explanation
base		Ten in the decimal notation of numbers; two in the binary notation of numbers; eight in octal notation; and in general, the radix in any scale of notation for numbers.
binary		Involving the integer two. For example, the binary number system uses two as its base of notation.
binary-coded decimal notation		One of many systems of writing numbers in which each decimal digit of the number is expressed by a different code written in binary digits. (See "excess-three code").
binary digit		A digit in the binary scale of notation; is always "0" or "1"; and is a numerical image of a bistable ("on-off" "yes-no" etc.) machine component.
bit		A binary digit.
block		In Univac I and II and in the Univac Scientific systems, a block is a formated group of exactly 720 characters, and is a machine unit of input/output operation. A block is not a machine unit in the UFC; when used in UFC, the term block merely indicates a group or ensemble of characters. It is in this sense that the Block Transfer Buffer is named.
blockette		A group of 120 contiguous characters. The blockette is the UFC System's fundamental unit of input/output format (e.g., card unit and magnetic tape unit data transmissions to and from the computer are 120 character or blockette transmissions. In these devices the blockette is the principal operating unit). The blockette is also used to specify the entire contents of a 120-character buffer or a

Term	Symbol	Explanation
		track on the High Speed Drum.
Blockette Address	Z	When Z is placed in the lower-order character position of a Program Control Storage Address the entire contents of a track or buffer (120 characters) is referred to.
Block Transfer Buffer	BTB	A 120-character magnetic core memory used in Buffer Transfer instructions to transmit a "block" of data (1 up to 120 characters) from one Program Control Storage location to another. BTB is itself a Program Control Storage Location (10x) which is Word, Field, and Blockette addressable in all instructions except the Buffer-Transfer instruction. (In Word Addresses x is 0 through 9; in Field Addresses, x is A-V, excluding I and O; in Blockette Addresses, x is Z).
Block Transfer Buffer Pattern	BTP	A 120-bit register which holds the Field Selection Pattern for BTB; addressable (99X) only as a Destination.
borrow		The digit to be taken from the next higher digit position (and subtracted from that digit position) when the subtrahend digit of one digit position exceeds the minuend digit of that digit position.
Branch Storage	BS	A temporary memory in the Central Computer for the result of arithmetic or comparison operations. Branch Storage is set to 0 by every instruction just before process time. If the instruction specifies an arithmetic process (add, subtract, multiply, or divide) or a compare process, Branch Storage is set to + if the result is + (or if $V_1 > V_2$), or to - if the result is - (or $V_1 < V_2$). If the result is 0 (or $V_1 = V_2$), the 0 setting of Branch

B

Term	Symbol	Explanation
		Storage is retained. In plugboard operations, the contents of Branch Storage can be probed, and used for program variance by BRANCH hub patchcord wiring. In internally-defined programs, if S in the Operation Code of an Instruction Word is found to be any of the following values: 2, 3, 4, 9, B, C, D, or I, the setting of Branch Storage at that time (+ , 0, or -) is transmitted to a more permanent type of storage called Conditional Storage. Conditional Storage is then tested by Jump instructions to cause program variance.
Branching		A means of determining the next operation, depending on whether the result of a Program Step is + , -, or 0. It is also used in connection with the Compare Process to determine the next operation depending on whether V_1 is > , < , or = V_2 (See "Compare").
Breakpoint 1,2,3		Any one of the three Breakpoint signals (1,2,3) can be produced on the Program Control Plugboard if (1) an Instruction Word's Special Character specifies that Breakpoint (1,2, or 3); and (2) the BREAKPOINT SELECTOR is appropriately set to allow that Breakpoint to be produced. As its name implies, Breakpoint is generally used to interrupt an internally-defined program (i.e., stop the execution of Instruction Words) to allow a check to be made on the progress of the program or to stop. If the BREAKPOINT hub which emits is patched to a STEP IN hub, a series of Program Steps can be initiated. This series of Program Steps can be designed

Term	Symbol	Explanation
		to make (limited) automatic adjustments in the internal program; to execute an auxiliary sub-routine (related to the program being solved); or, as is generally the case, to cause an automatic type-out of the contents of certain registers and memory locations for checking purposes. This operation can be made completely automatic (in terms of the internal program) by patching the STEP OUT of the last Program Step to a NEXT INSTRUCTION hub. This causes the internally-defined program to resume immediately after the (Breakpoint-initiated) plugboard sequence. If the BREAKPOINT hub pulsed is patched to a STOP hub, Program Control operations cease. A visual check can then be made on the contents of certain registers and memory locations, etc. In this case, the program can be resumed only if the necessary manual controls are operated. If the BREAKPOINT hub which emits is patched to a NEXT INSTRUCTION hub, the internal program is not interrupted, and the execution of Instruction Words continues as if no Breakpoint were initiated. If a Breakpoint is produced, but the corresponding BREAKPOINT hub is not patched, the computer hangs up.
Breakpoint plugboard sequence		A sequence of Program Steps which is initiated when the computer executes a Breakpoint and the appropriate BREAKPOINT hub is patched to a STEP IN hub on the plugboard.
bus		In the computer, also referred to as a transfer bus, and is a line, or trunk over which data transmissions occur from any of several sources to any of several destinations. On the Program Control Plugboard, a tie-point for several similar outs to be patched to

B

Term	Symbol	Explanation
		a common in, and a means of expanding an out that must be patched to several ins.
BUS hub		A hub on a plugboard; generally used either to facilitate wiring multiple outs (not requiring isolation) to a common in or for wiring an out to several ins.
buffer		A temporary storage for data; an isolating device, generally used to transfer data between two storage elements that are not synchronized.
Buffer Transfer	BT	A computer Instruction generally used for data transfers of 12 characters or more. A maximum of 120 characters can be transmitted by this instruction. If Field addressing is employed, any number of characters from 1 up to 120 can be transmitted. (Special rules apply to BTB in connection with this instruction.)

Term	Symbol	Explanation
carry		The digit to be taken to the next-higher order digit-position (and there added) when the sum of the digits in one digit position equals or exceeds the number base.
carrow		A general term referring to those signals from one digit position that effect an addition to or subtraction from the next-higher order digit position.
Central Computer		That section of a Univac File Computer System which controls the operation of the system and executes the program of arithmetic and logical operations necessary to the solution of a problem. The Central Computer is composed of the following principal parts: Program Control, Program Control Storage, an Arithmetic section, and an Input/Output Control system.
channel		A narrow band on the periphery of a General Storage Drum; the area which passes beneath a read/write head as the General Storage Drum revolves. (The equivalent of a track of the High-Speed Drum.) Also a level on punched paper or magnetic tape.
Channel Search Operations	CS = CS ≠	There are two Channel Search operations: Channel Search = Channel Search ≠ Each is a time-shared operation of the General Storage System, and each is initiated when its correspondingly named Sub-Instruction or Sub-Step (CS = or CS ≠) is executed in the Central Computer. In both Channel Search operations the basic objective is to examine a file or portion of a file in General Storage to

C

Term	Symbol	Explanation
		locate a Unit Record whose address is unknown.
Channel Search Storage		A special-purpose memory in the Central Computer used for storing the result of a Channel Search operation. Channel Search Storage is set by General Storage to + , 0, or - as follows:

	+	0	−
CS=	"find"	Unit Record "found" with ignore codes in every character position compared.	"no find"
CS≠	"find"	---	"no find"

Channel Search Probe		In internally-stored programs, an Instruction Word; in plugboard-defined operations a Sub-Step; in either case, a means of testing the contents of Channel Search Storage for program variance.
clear		To replace information in a register by (Univac coded) zeros, ignore codes, or space codes; in the case of certain special-purpose memories to remove any previous setting.
code		(noun) A system of symbols for representing information in a computer and the rules for associating them. (See Univac code.) (verb) To program.
Code Distributor Register	CDR	A 1-character addressable register used to store any one of a group of 40 permissible characters for the purpose of variously routing pulses or d-c enables on the Program Control Plugboard. (Program Control Storage Address is 994.)

Term	Symbol	Explanation
command		A pulse, signal, or set of signals initiating one step in the execution of a computer instruction, sub-instruction, or sub-step.
comparator		A group of circuits which compare two quantities and presents an appropriate indication of the result of the comparison. Comparators are used in a variety of places in the UFC System: the General Storage System, Arithmetic Section, Magnetic Tape Units, etc.
computer character		The basic unit of data for the UFC. A legal character is any letter, number, or symbol that can be expressed in Univac Code. Each computer character consists of seven bits: 6 bits in Univac Code, and 1 parity bit
Computer Ground		An unconditional ground supply for the coil circuit of a Selector. See "Demand Ground".
Computer Instruction		A completely defined operation for the computer; the principal unit of a computer program. Two types of Computer Instructions are employed: Instruction Words for internally-stored programs Program Steps for plugboard-defined programs.
computer language		Univac coded characters.
Computer-To-Input/Output Control Lines	C-I/O	A group of ten lines (labeled A-J) emanating from the Central Computer, over one or more of which the computer program sends a signal (or combination of signals) that defines an I/O Instruction. These lines are switched to a particular I/O Unit when that Unit is placed "on demand".

C

Term	Symbol	Explanation
Conditional Storage		A special-purpose memory in the Central Computer used only in internally-stored programs for more permanently retaining the contents of Branch Storage. Conditional Storage is set during the execution of a Set Conditional Storage Sub-Instruction. It retains that information until another Set Conditional Storage Sub-Instruction is executed. The following Instruction Words use the setting of Conditional Storage for program variance: Jump Plus, Jump on Minus, and Jump on Zero.
conditional jump		A jump instruction wherein a jump is achieved only if a particular condition tested exists. E.g., the Jump on Plus, Jump on Zero, Jump on Minus, and Channel Search Probe Instruction Words.
Condition Compare		A computer Sub-Step. Normally, in the Compare Process, any space codes present in V_1 or V_2 are treated as if they were zeros in the comparison of V_1 and V_2. When a Condition Compare Sub-Step is performed, space codes and zeros are each given their weighted values in Univac Code in the next Compare process, and V_1 and V_2 are compared as they actually are.
contents of	( )	The information stored within.
contiguous		Adjacent.
Current Instruction	CI	The instruction now being executed by the computer; CI is always held in IRVc. (See IRV for explanation of IRVc.)

Term	Symbol	Explanation
debug		To isolate and remove a computer malfunction or the programming mistakes in a computer program.
d-c enable		A term normally applied to a flip-flop output based at -45 volts d-c and rising to +35 volts d-c and staying at +35 volts d-c for at least 3 sec.
Demand In		A sequence, initiated by the computer, that places a particular I/O Unit "on demand" and permits control information to be exchanged between the computer and that particular I/O Unit. This sequence is initiated internally by the Demand In Instruction Word; it is initiated on the plugboard by pulsing the appropriate DEMAND IN (0-9) hub.
Demand Ground		One of ten conditional ground supplies (0-9) each of which is associated with an I/O Unit's Demand Station, and is completed only when its associated I/O Unit is "on demand".
DEMAND OUT		A signal produced during a Demand In sequence by the I/O Unit "on demand" indicating that the I/O Unit "on demand" is READY and no (HS) I/O-C control lines are energized.
Demand Station		Part of the control circuitry of each I/O Unit; specifically that which (a) is the I/O-terminus of the connective cabling between the computer and the I/O unit; and (b) is the actual control and synchronization circuitry used by the Central Computer in monitoring I/O operations. The principal parts of a Demand Station for I/O Unit "b" are:

D

Term	Symbol	Explanation
		C-I/O control lines A-J; (LS)I/O-C control lines a-l; (HS)I/O-C control lines W-Z; Demand Ground "b" supply; *Demand Test in "b" control lines; DEMAND TEST IN "b", READY "b", NOT READY "b"; *Demand Test In control circuits: Demand In "b" control lines: DEMAND IN "b", DEMAND OUT "b", SPECIAL OUT "b"; Demand In sequence control circuits; I/O Track Control circuits: Set SAR "b" enable Read & Write circuits for I/O Track 0"b" Track switching circuits for I/O Track "b" --- *In developing the mnemonic code for the Process portion of Instruction Words, it was convenient to call the Instruction Word which initiates a Demand Test In sequence, a Test Demand In (TD) Instruction Word. Demand Test In and Test Demand In are identical sequences.
Demand Station Position		Ten sets of connectors on Program Control Cabinet #2. Each of these sets links together a group of I/O Control circuits in the computer (to which the set is permanently wired) and an I/O Unit's Demand Station (to which the set is connected when cabling from the I/O Unit's Demand Station is plugged into the set.)

Term	Symbol	Explanation
Destination Address		An address that specifies a particular location (of definite capacity) at which information held in some intermediate storage is to be stored; in plugboard operations, the R ADDRESS; in most Instruction Words, the W-section of the Instruction Word; in Jump Instruction Words, the V-section of the Instruction Word.
difference		The result of a subtract process.
digit		A term for the computer characters 0-9.
digital computer		A computer which calculates using discrete quantities, as numbers expressed as binary-coded digits or "yes" - "no" conditions expressed as bits, to represent all the variables that occur in a problem. Differentiated from analog computers in that it counts rather than measures in solving a problem.
dividend		A number to be divided. In the UFC, V_1 in divide processes.
divisor		The number by which the dividend is divided. In the UFC, V_2 in divide processes.
down-time		Generally refers to the time when a computer is malfunctioning or not operating due to machine failure. Also used in a broader sense to include all times when the machine is idle.

E

Term	Symbol	Explanation
End of Data code		A term generally used in connection with input operations. An End of Data code is control information that is included along with the actual data to indicate that no further data is available. On magnetic tapes, End of Data is designated by at least one complete word, 12 characters, of "%" codes.
End of File code		A special code stored along with actual data; used to separate files. In General Storage, prime zero ('0) is used; on magnetic tape 12 "Z's" in at least one word are employed.
error		A computer malfunction or programming mistake.
erase		A term usually applied to magnetic drums and magnetic tapes to mean "record binary 0's" on the drum or tape. In this connection it effectively "clears" the area on which the 0's are recorded. Since other methods of "clearing" are employed in UFC (e.g., space codes and ignore codes are used by certain devices to "clear" drum areas) the term erase is used in UFC only in connection with revolvers and magnetic tapes, as: "the area ahead of the read/write head is erased prior to recording." Erased areas are parity-bad.
excess-three code		Basically, a binary-coded decimal notation for decimal numbers. In this code each decimal digit is represented by a binary number (four bits in length) whose value is three greater than the value of the decimal digit that is coded. For use in Univac I and II and in UFC, the code is extended to six bits in length and includes a weighted value for 64 characters, ten of which are the digits 0-9.

Term	Symbol	Explanation
external memory		Any memory device in the system which can permanently store (and subsequently supply) data, but which is not directly accessible to Program Control Storage. Examples: General Storage System: an addressed external memory in which random-access storage references can be initiated under computer control, but are accomplished by the General Storage System rather than by Program Control Storage. Magnetic Tape, Punched Cards { data available sequentially in input/output operations.

F

Term	Symbol	Explanation
Factor Storage Tracks	FS-Tracks	Tracks 11x and 12x on the High Speed Drum. For Word Addresses, x is 0-9; for Field Addresses, x is A-V (excluding I and O); for Blockette Addresses, x is Z.
field		A unit of data on the HSD, in GSB, or in BTB which is a collection of contiguous characters; the number of characters vary from 1 up to 119 as defined by a Field Selection Pattern.
Field Address		Addresses which can be used in storage references involving the HSD, BTB or GSB to obtain or store fields. Each is composed of two numeric characters and a letter A-V (excluding I and O).
Field Selection Pattern (or Pattern)		A collection of 120 bits used to define the fields in a buffer or on a HSD track. The bits involved are parity bits. Each field begins with "0" and ends with "1".
File		The programming unit used for overall organization of data. On magnetic tape, a collection of contiguous Items each of which consists of a fixed master portion comprising subjects common to all (or practically all) Items, and to which are added or appended a variable number of other "trailer" subjects which may range in number from none to several hundred. In General Storage a similar concept applies except (a) Unit Records are involved, and (b) in some applications, the General Storage Address for a Unit Record can be used to specify a portion of the information that must be coded in the master portion of Items on magnetic tape.

Term	Symbol	Explanation
flow-chart		A graphical representation of a sequence of programming operations using symbols to represent operations such as compute, substitute, compare, jump, etc.
flow diagram		A schematic-type representation of a sequence of sub-routines designed to solve a problem. It is less detailed and less symbolic than a flow-chart and frequently includes descriptive text.
Function Delay	FD	A circuit available via the Program Control Plugboard which delays the start of a dependent operation until another operation has been completed. Four Function Delay circuits are provided in the UFC, Model 1. Each has two IN hubs and an OUT hub on the Program Control Plugboard.
Function Sequence	FS	A group of circuits which produce an output when two inputs (the first to a SET hub, the second to a PROBE hub) are received in proper order. It is thus a means of insuring that two ordered events occurred in the proper sequence. Four Function Sequence circuits are provided in the UFC Model 1. Each has a SET, PROBE and OUT hub on the Program Control Plugboard.

G

Term	Symbol	Explanation
General Storage Address		A 7-digit address which specifies either a Unit Record Area or a Search Control Location on a General Storage Drum; used for storage references in the General Storage System:

format: Unit Record Areas (LDSCHAA)
Search Control Locations (-DSCH'0)

where L Unit Record Length: number of computer words in a Unit Record (1, 2, 3, ... 9, 0 where 0 means ten words)

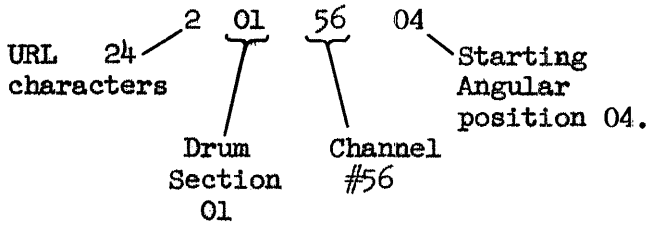
DS Drum Section (3 per drum): 00 - (3n-1) where n is the number of drums in an installation.

CH Channel in that section (100/section): 00-99.

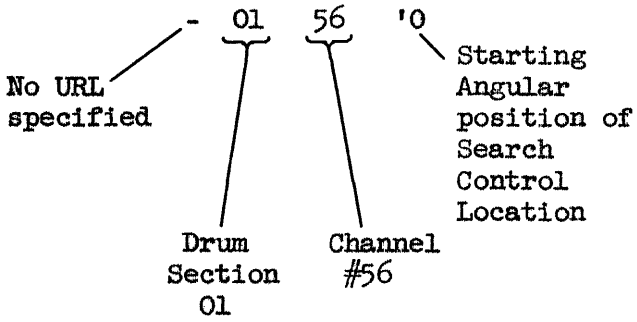
AA Angular Address: starting angular position of a Unit Record Area.

'0' the starting angular position of the Search Control memory location:

TYPICAL UNIT RECORD AREA ADDRESS



SAME CHANNEL'S
SEARCH CONTROL LOCATION ADDRESS



Term	Symbol	Explanation
General Storage Address Register	GSAR	A 7-digit register that holds a General Storage Address in General Storage References. Three types of General Storage Addresses are placed in GSAR: (a) The address of a Unit Record Area at which a Unit Record is to be read or stored; (b) The address of a Unit Record Area at which a Channel Search operation is to begin; or (c) A search Control Location address which specifies a particular Search Control Location at which special data (6 characters) is to be read or stored. GSAR is also a Program Control Storage location (995). It functions in the General Storage System during operations of that system, and at all other times is part of the Central Computer.
General Storage Buffer	GSB	A 120-character magnetic core register which functions as a buffer for data transmissions to and from the General Storage Drums. In General Storage operations, GSB is part of the General Storage System. GSB holds the Unit Record to be written in Write UR and Write UR and Check operations. It receives the Unit Record read in Read UR operations. In Channel Search operations: GSB initially holds the UR Identifier; at the conclusion of a Channel Search, GSB holds the UR "found" if a "find" occurred, or the UR Identifier if no "find" occurred. Except during those times in which General Storage operations are taking place, GSB can be used as a Program Control Storage Location (98x) which is Word, Field, and Blockette addressable. (in Word Address x is 0 through 9; in Field Addresses x is A-V, excluding I and O; in Blockette addresses x is Z.) During General Storage operations data transmissions to and from GSB always begin with GSB's Word 9, Character S position.

G

Term	Symbol	Explanation
General Storage Drum	GSD	A large diameter magnetic drum used as the storage medium for the General Storage System. Each GSD is divided into 3 sections, 100 channels per section. Each channel in turn is divided into 100 6-character areas (in which Unit Records can be stored); and a Search Control Location in which 6 characters of control data for sequencing search operations can be stored.
General Storage Buffer Pattern	GSP	A 120-bit register which holds the Field Selection Pattern for GSB; addressable (99Y) only as a Destination.

Term	Symbol	Explanation
hardware		The mechanical, magnetic, electrical, and electronic components from which a computer is constructed.
higher-order		The significance given data in the computer is the same as that associated with reading or hand-writing data. Left-most character is the most-significant or highest-order digit; right-most character is the least-significant or lowest-order digit. (Exception: in an operand the sign is always the lowest-order character in the computer.)
High-Speed Drum	HSD	A small diameter magnetic drum used as (the principal) part of the operating memory of the Central Computer. Three types of tracks on this drum are Word, Field, and Blockette addressable as the source or destination of data: I/O-Tracks (20, two for each track address), Factor Storage Tracks (2), Intermediate Storage Tracks (85). The HSD also contains the Instruction Revolver, the Shift Revolver, the High-Speed Drum Pattern (ISP), and timing tracks.
High-Speed Drum Pattern	ISP	A special track on the High Speed Drum that contains the Field Selection Pattern for the I/O, FS, and IS Tracks; address-able (99W) only as a destination.
High Speed Input/Output-to-Computer control lines	(HS) I/O-C	A group of four lines (W, X, Y, and Z) in each I/O Unit, over which the I/O Unit can send program variance information to the computer's High Speed I/O-C Control Line Storage. An I/O Unit's (HS) I/O-C lines are switched to the computer when the I/O Unit is "on demand" and ready. If an I/O Unit has produced a W, X, Y, or Z (or combination of these) condition(s), it will send the corresponding W, X, Y, Z or combinations of these to the computer when it is (a) placed "on demand" and (b) goes READY. (HS) I/O-C are sent to

Term	Symbol	Explanation
		the computer only during Demand In sequences: they set High Speed I/O-C Control Line Storage in the computer; and they are always accompanied by a SPECIAL OUT.
High Speed Input/Output-to Computer Control Line Storage		A special purpose memory in the computer for the (HS) I/O-C control line signals: W, X, Y, and Z. This is a temporary storage which is cleared each time an I/O Unit is placed "on demand". Accordingly if a SPECIAL OUT is detected in a Demand In sequence, this memory should be immediately interrogated: in internally-stored programs by Test Incoming Control Instruction Words, in plugboard-defined operations by SPECIAL OUT-to-W, X, Y, Z IN hub patching.
hub		A receptacle for patchcord wiring on a plugboard; the break-off point between patchcord wiring (variable) and internal wiring (fixed).

Term	Symbol	Explanation
INACTIVE		When a Channel Search operation is initiated, General Storage becomes ACTIVE, and remains ACTIVE until the search operation is completed. General Storage then becomes INACTIVE. When a Channel Search Probe is made and General Storage has become INACTIVE as described above, the results of the channel search (+, 0, or -) are made available, and program variance can be achieved if desired.
information		Information is commonly used as a general term for both actual data and for control data.
Ignore code	i	A computer character (1000000) used primarily to suppress comparisons.
Input/Output Instruction	I/O Instruction	An instruction or an operating condition for an I/O Unit; the control data sent by the computer to an I/O Unit via the C-I/O control lines A-J.
Input/Output Storage Tracks	I/O Tracks	A collection of twenty tracks on the HSD which operate as 10 pairs of tracks. At any given time one track of the pair is connected to the computer; and the other track of the pair is connected to an I/O Unit. Program control Storage Addresses: 00x through 09x; for Word Addresses x is 0 through 9; for Field Address x is A through V (excluding I and O); for Blockette Address x is Z.
Input/Output Units	I/O Units	Devices which either supply data to the computer (Input) or receive data from the computer (Output).
Instruction Revolver	IRV IRVc IRVn	Although it has but a single address (996), IRV is actually two 12-character revolvers on the High-Speed Drum: IRVc and IRVn. IRVc is used to store the Instruction word currently being executed:

I

Term	Symbol	Explanation
		IRVn is used to store the next Instruction Word to be executed. The next Instruction Word is stored in IRVn as the current Instruction Word is executed. When the execution of current Instruction Word is completed, control switches revolvers: IRVn becomes IRVc and supplies the data for the new current Instruction Word; and the next Instruction Word is loaded into what was IRVc but which is now IRVn.
Instruction Word	IW	A 12-character computer word that defines a computer instruction; is stored in the operating memory of the computer, usually in sequence with other Instruction Words on the HSD. Format: U V W OP xxx xxx xxx xxx (x = 1 character) where U, V, and W are usually storage addresses for V_1, V_2, and R, respectively; and OP (operation code) = PRS or TCS PR = PROCESS code TC = TRANSCOP code S = Special Character which extends or modifies the operation specified by PR or TC.
intermediate storage		A temporary storage location into which data is automatically transmitted from a programmed Source or from which data is automatically obtained for storage in a programmed Destination.

Term	Symbol	Explanation
Intermediate Storage Tracks	IS Tracks	A group of 85 tracks on High Speed Drum; their addresses are 13x through 97x; used primarily to store Instruction Words. For Word Addresses, x is 0-9; for Field Addresses, x is A-V (excluding I & O); for Blockette Addresses, x is Z.
Internal memory (or operating memory, or Program Control Storage Locations.)		Addressed storage locations in the Central Computer which are directly-accessible via their Program Control Storage addresses.
Item		A programming unit of format for magnetic tape data. A collection of characters which is a unique major file-entry. Each Item contains a prime set of identifying data (or master portion) which distinguishes that Item from all other items in a File; is recorded in the same pattern as the prime set of identifying data in every other Item; and contains all the data including, in general, the appropriate Search Validation Character required for file classifications and searches (i.e., contains all the data elements which are ever referred to in classifying that Item or in finding that Item). In addition, an Item may contain one or more sub-Items as "trailer" information, (i.e., additional information which is not used to classify or locate that Item)
Item Identifier		An Item Identifier is a collection of from 1 up to 120 alphanumeric, non-ignore characters, not necessarily contiguous, which are either a copy of the prime set of data (or master portion) in an Item, or a copy of a set of data derived from

Term	Symbol	Explanation
		the prime set. For equipment reasons it is necessary to program Item (and Sub-Item) Identifiers as 120-character quantities, with Ignore codes in every character position that is not to be compared with tape data in a tape search operation. Item Identifiers are used principally to locate full Items; they can also be used to locate any set of data in an Item's prime set of identifying data. (In the foregoing it is assumed that the appropriate Search Validation character is included in the prime set of identifying data, and hence in the Item Identifier.)

Term	Symbol	Explanation
logical design		Design that deals with the logical and mathematical inter-relationships that must be implemented by the hardware.
logical operation		The operations of masking, comparing, normalizing, etc. where in essence characters or bits constitute the elements being operated upon. In contrast to arithmetic operations wherein the elements of the operation are numerical values.
lower-order		Pertaining to the right-most digit of a typewritten or handwritten number or message. (See higher-order. In UFC words, the Sign is the lowest-order character of each operand.)
Low-Speed Input/Output-to-Computer control lines	(LS) I/O-C	A group of twelve relay-operated lines (a-l) in each I/O Unit over which the I/O Unit can send program variance information to the Program Control Plugboard. An I/O Unit's (LS) I/O-C lines are switched to the computer when the I/O Unit is placed "on demand". If an I/O Unit is placed "on demand" and has energized any (one or more) of its (LS) I/O-C lines, B+ is immediately applied to the correspondingly named (LS) I/O-C CONTROL LINE hubs, a-l, on the Program Plugboard.

M

Term	Symbol	Explanation
machine word (or computer word)		Twelve characters; two general types; operands and Instruction Words
magnetic core		A form of rapid-access storage wherein information is represented as the polarization of a wire-wound magnetically permeable core, which may be straight, doughnut-shaped, etc.
magnetic drum		A rapidly rotating cyclinder, the surface of which is coated with a magnetic material on which information can be stored as small polarized spots. See High-Speed Drum and General Storage Drum.
magnetic tape		Tape made of a metal or a plastic base that is coated with a magnetic material on which polarized spots representing information can be recorded.
malfunction		Equipment failure.
Mask Transfer		A Computer Instruction which permits characters to be screened out of one operand on the basis of the characters stored in another operand called a mask.
memory		Information storage; any device into which information can be introduced and then extracted at a later time.
memory capacity		The amount of information which a memory system can store. Quoted by number of words, decimal digits, characters or bits that can be stored.
millisecond	ms	A thousandth of a second
minuend		The number from which another (called a subtrahend) is to be subtracted. In the UFC, V_1 in a subtract process.

Term	Symbol	Explanation
mnemonic code		The Process codes of Instruction Words have a mnemonic as well as numeric listing. The (two) letters employed in each mnemonic code suggest the name of the instruction and have the same lower-order four bits in Univac code as the numbers they represent.
microsecond	μsec	A millionth of a second
multiplicand		The number that is to be multiplied by another number, called a multiplier. In the UFC, V_1 in multiply processes.
multiplier		The number by which another number is multiplied. In the UFC, V_2 in multiply processes.

N

Term	Symbol	Explanation
Next Instruction		A term employed to indicate the instruction whose execution is to follow that of the instruction currently being executed.
NEXT INSTRUCTION hub	NI	In plugboard programs when the NEXT INSTRUCTION hub is pulsed, a Next Instruction Sub-Step is initiated which interrupts plugboard operations and reverts control to the internal program. (The next instruction executed is an Instruction Word rather than a Program Step.)
non-volatile storage		Storage media which retain information in the absence of power, such as magnetic drums, magnetic cores, magnetic tapes, etc.
(left) normalize		An operation in the UFC Arithmetic Section wherein an operand is shifted left until its most-significant character is in a register's most-significant character position.
Normalized Operand		An operand, so positioned in a register, that its most-significant character is located in the register's most-significant character position.
Normalizing Count		A count of the number of places an operand is shifted left in a normalize operation. (In UFC this count is stored in Register B at the conclusion of a Left Normalize instruction.)
notation		A manner of representing numbers. If quantities are written in the scale of notation "n" then the successive positions of the digits report the powers of "n".
NOT READY		A condition of an I/O Unit which means that the I/O Unit is either inoperable or still engaged in a previously initiated operation.

Term	Symbol	Explanation
"on demand"		An I/O Unit is placed "on demand" when a Demand In sequence is executed. In general, an I/O Unit stays "on demand" until it receives an I/O Instruction or until another I/O Unit is placed "on demand". When an I/O Unit is placed "on demand": (a) the following control lines from that and only that I/O Unit are connected to the computer: C-I/O (A-J) (LS) I/O-C (a-l) (Information on the (LS) I/O-C lines is available as soon as the relays involved energize; the information on the (HS) I/O-C lines does not become available until after an I/O Unit "demanded" becomes READY: C-I/O information is generally sent only after the DEMAND or SPECIAL OUT signals are produced, i.e., after unit "on demand" goes READY.) (b) The set SAR enable is available for plugboard addressing of the I/O Track associated with the I/O Unit "on demand". (c) The Demand Ground circuit associated with that I/O Unit is completed.
"off demand"		An I/O Unit is taken "off demand" when another I/O Unit is placed "on demand". No exchange of information over the (HS) I/O-C, (LS) I/O-C or C-I/O lines can occur when an I/O Unit is "off demand". However, Demand Test In sequences and Track switching can be performed, and Demand In sequences can be initiated.

O

Term	Symbol	Explanation
operand		A computer word used in arithmetic and logical operations. Numeric operands have the following format: High-Order Low-Order X X X X X X X X X X X X 11 10 9 8 7 6 5 4 3 2 1 Sign
Operation Code	OP	The right-most (or lowest-order) three characters of an Instruction Word. These three characters have the following significance: two left-most characters: PROCESS code or TRANSCOP code right-most character: Special Character, S Operation Code defines what the computer is to do in executing the instruction.
Operation Pulse/Enable Distributor	OED	A group of circuits which time-sequences and controls the execution of instructions in the computer.
Out Expander		A combination of three hubs on a plug-board: one IN hub and two OUT hubs. When the IN hub is pulsed, the two OUT hubs emit. The OUT hubs are diode-protected to prevent back circuits. Out Expanders are used to multiply and amplify pulse outs that are to be patched to several ins.
overflow		In a counter or register, the production of a number which is beyond the capacity of the counter or register.
Overflow Address		The new address in the General Storage System from which a channel search operation is to continue if

Term	Symbol	Explanation
		(a) a terminal condition for the search does not occur in a channel by the time that channel's Search Control Location is read; and (b) other Unit Record Areas still need to be searched in the operation. In short, one of the two possible kinds of search-sequence control data that can be stored in a channel's Search Control Location.

P

Term	Symbol	Explanation
packing		To combine several different brief fields of information into one or more machine units of storage (as word locations on a track) for the purpose of conserving memory space.
pad		The filling out of a word or field with (Univac coded) zeros, space codes, or ignore codes.
parity bit		A redundant bit stored with each 6-bit Univac coded character. If the number of "1's" in the 6 bits of the Univac Code is even, the parity bit is a "1"; if the number of "1's" in the 6 bits of the Univac Code is odd, the parity bit is "0". An odd-even check on each character can thus be made during data transmissions. The parity bit is also used in defining the Field Selection Patterns used in Field addressing.
Parity Check		A check made on each character, generally during data transmissions, to determine if the number of "1's" in each 7-bit computer character is odd or even. If odd, operation continues; if even, a parity error is indicated.
patchcord wiring		Manually-plugged wiring used on a plugboard or pinboard to connect various circuits within a unit for the purpose of defining what operations the unit is to perform.
Pattern Address		A Program Control Storage Address which refers to the ISP, BTP, or GSP (98W, 98X, and 98Y, respectively).
Pinboard		A non-detachable connection panel whereon patchcord wiring can be employed to define various logical or control conditions.

Term	Symbol	Explanation
plugboard		A removable connection panel whereon patchcord wiring can be employed to define a program. The Central Computer's plugboard is called the Program Control Plugboard. Several I/O Units also have plugboards.
Plugboard Addressing System		The collection of hubs listed below which are located on the Program Control Plugboard, and to which the V_1 ADDRESS, V_2 ADDRESS, and R ADDRESS hubs can be patched to specify Program Control Storage Addresses; used in plugboard operations to define programmed storage references in Program Steps. RA BTB WORD (0-9) RB GSB WORD (0-9) RC FS #1 WORD (0-9) RD FS #2 WORD (0-9) GSAR I/O WORD (0-9) CDR BTB FIELD (A-V) PAK GSB FIELD (A-V) U ADR FS #1 FIELD (A-V) V ADR FS #2 FIELD (A-V) W ADR I/O FIELD (A-V) BTP BTB-Z GSP GSB-Z ISP FS #1-Z IRV FS #2-Z SRV I/O-Z
Plugboard Step		One of 48 locations (numbered 51-98) on the Program Control Plugboard where Program Steps can be defined by patchcord wiring.
prime zero	'0	A pair of symbols which: (a) If placed in the two lowest-order positions of a General Storage Address, specify a channel's Search Control location; or

Term	Symbol	Explanation
		(b) if placed in the two lowest-order character positions of the Search Control Location, itself, form an End of File (or Stop) code which terminates Channel Search operations.
probe		To interrogate; usually used in connection with pulses; however, in this publication both pulse and d-c probes are referred to.
process		The basic operation to be performed during a computer instruction.
PROCESS Code	PR	The higher-order two digits of the Operation Code of an Instruction Word if a number <50 is specified by these digits. The following numbers (and corresponding mnemonic codes) are used to define UFC Model 1 Instruction Word Processes: 14 (AD) Add 22 (SB) Subtract 43 (ML) Multiply, Store Lower 44 (MU) Multiply, Store Upper 48 (DQ) Divide, Store Quotient 35 (LN) Left Normalize 32 (LS) Load Shift 37 (CP) Compare 19 (JZ) Jump on Zero 17 (JP) Jump on Plus 15 (JN) Jump on Negative 41 (UJ) Unconditional Jump 33 (CC) Channel Clear 31 (LA) Load GSAR 34 (TD) Test Demand In 45 (DE) Demand In 39 (TI) Test Incoming Control 49 (DR) Divide, Store Remainder 13 (AT) Arithmetic Transfer 23 (BT) Buffer Transfer 42 (MK) Mask Transfer 29 (SZ) Suppress Left Zeros 27 (SP) Channel Search Probe 24 (SU) Substitute U

Term	Symbol	Explanation
		25 (SV) Substitute V 26 (SW) Substitute W
Process Register		A two digit register which holds the left-most two digits of an Operation Code during the execution of an Instruction Word, and a Plugboard Step number during plugboard operations.
product digits		The result digits in a multiplication operation.
Program		(noun) A sequence of computer instructions so arranged and coded that its execution by the computer solves some problem or effects a series of data transmissions, as entry or exit of data from General Storage, or reproduction of data on one medium, say cards, on another medium, as magnetic tape. (verb) To design a program.
Program Address Counter	PAK	An addressable three-character shift register used to sequence internally stored programs. When Instruction Words are obtained they are always taken from the address specified by the contents of PAK. PAK is also a Program Control Storage Location (997).
Program Control		That section of the Central Computer which interprets, executes and sequences computer instructions.
Program Control Plugboard		The Central Computer's plugboard.
Program Control Storage		All those storage locations which form the operating memory of the Central Computer, as well as all the circuitry required for finding a specified loca-

Term	Symbol	Explanation
		tion and achieving the desired storage reference. Principal functions are: procurement of operands, storage of results, and acquisition of the next Instruction Word. Also involved in all data transmissions that take place automatically as part of the execution of an instruction.
Program Control Storage Address		A three-character address, the left-most two characters of which are always numeric, and the right-most character of which is numeric (0-9) if the address is a Word or Register Address; or alpha (A-V, excluding I and O) if the address is a Field Address; or Z if the address is a Blockette Address.
Program Select		A group of circuits used to "pick-up" Selectors by a pulse at a specific time in a program and exercise control over the length of time the Selectors remain picked up. Sixteen Program Selects are provided. Each has the following hubs on the Program Control Plugboard: IN - This hub is pulsed to energize the Program Select Circuit. OUT- When the IN hub is pulsed, the OUT hub emits B+ (this hub is wired to the appropriate Selector(s)' SELECTOR PICK-UP hubs, 1-48) DELAYED OUT hub - Can be patched to continue the program if the source of the pulse sent to the IN hub is not split wired. (The DELAYED-OUT pulse is not issued in Program Select #1 until 15 ms after the IN hub is pulsed, and until 10 ms after the IN hub is pulsed in all other Program Selects; this allows time for the contacts to close in the Selectors "picked up".)

Term	Symbol	Explanation
		DROP OUT - This hub is pulsed to stop the OUT hub from emitting; i.e., to "drop out" the Selector(s) to whose SELECTOR PICK-UP hub(s) the OUT was patched.
		When any of the CLEAR hubs associated with the Program Selects is pulsed, the effect is the same as dropping out all Program Selects which are emitting.
Program Step		A Plugboard-defined computer instruction. In general, nine hubs are patched to other hubs to define a Program Step: STEP IN, PROCESS, V_1 ADDRESS, V_1 SHIFT, V_2 ADDRESS, V_2 SHIFT, R ADDRESS, R SHIFT, and STEP OUT.
pulse		A normally less-than-three microsecond burst of electrical energy based at −20 volts and rising to +50 volts (no load).
punch card		A card of constant size and shape that can be handled mechanically, and is suitable for punching a pattern of holes that has a meaning. The punched holes are sensed electrically by wire brushes, mechanically by metal fingers, or photoelectrically. 80 or 90 column punched cards are used in UFC.
punched paper tape		Paper tape in which holes are punched in some pattern to store data in a coded form. The holes are sensed photoelectrically or mechanically to transcribe the data. Depending on the number of holes that can be punched in a row across the width of the tape, paper tapes are called 5, 6, 7, or 8-level tapes. (5, 6, or 7-level paper tape input/output devices are available in the UFC Model 1.)

Q

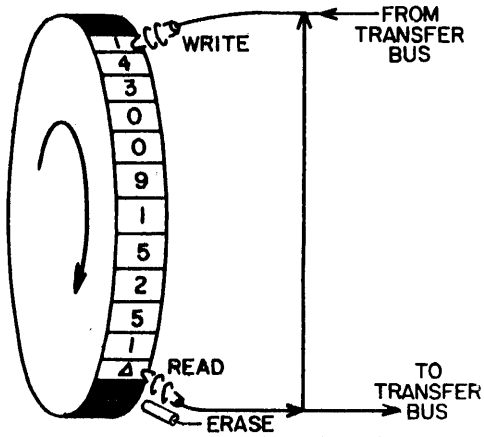
Term	Symbol	Explanation
quotient		The number resulting from the division of one number by another.

Term	Symbol	Explanation
random-access		A term applying to computer memories meaning non-sequential, direct-access to any memory location in a storage reference; as opposed to having to refer to memory locations one after the other to get to the one desired.
read		Refers to the action whereby stored data is sensed and transmitted elsewhere, or is shifted out of a location and sent elsewhere. In general, a read operation does not affect the final contents of the location read.
read/write head		A dual-purpose device containing a permeable core and one or more sets of windings. Used to read and record data on a magnetic media.
ready (and not ready) condition (s)		An I/O Unit is said to be in a ready condition if it is capable of receiving an I/O Instruction when placed "on demand". An I/O Unit is said to be not ready if any of the following conditions exist: the I/O Unit is inoperable (it has no power or some other error condition exists in the unit); or if, initially, the manual operations required to prepare the I/O Unit (internally) for operation have not been performed; or if, engaged in an I/O operation, it has not reached that point in its cycle where it would be capable of receiving another I/O Instruction were it placed "on demand".
READY(and NOT READY) signals		Signal (s) produced by an I/O Unit during a Demand Test In sequence indicating the I/O Unit is in a ready (or not ready) condition.

R

Term	Symbol	Explanation
Redundant Check		See Parity Check.
register		The hardware for storing a computer word or computer address.
Register A	RA	A 12-character addressable shift register which receives the operand V_1 from storage during the execution of certain computer instructions. (Program Control Storage Address is 990)
Register B	RB	A 12-character addressable shift register which receives the operand V_2 from storage during the execution of certain computer instructions. (Program Control Storage Address is 991.)
Register C	RC	A 12-character addressable shift register which is used in arithmetic operations either to form the result which is to be stored, or to form a result which is to be used for checking purposes. Register C is also used for temporary storage for U in the Jump Instruction Words. (Program Control Storage Address is 992.)
Register D	RD	A 12-character addressable shift register in which the result of arithmetic and logical operations is usually formed, and from which the result is usually stored. Register D also functions as a buffer in the Arithmetic Transfer Instruction. (Program Control Storage Address is 993.)
remainder		A number less than the divisor which if added to the product of the divisor and quotient will produce the dividend.
Result	R	The sole (or, in some cases, the principal) quantity produced in an arithmetic or logical process.

Term	Symbol	Explanation
round-off		To change a more precise quantity to a less precise one; usually choosing the nearest less precise one.
reverse process		Interchanging the relative function of operands in an arithmetic sequence to check a previous (normal) arithmetic operation.
revolver		A device which makes special use of a track on a magnetic drum to store and recirculate a fixed number of characters for the purpose of making the recirculated data readily accessible to the computer program. A 12-character revolver is illustrated below:



Data from the Transfer bus is recorded on the track by the Write head; when the recorded data reaches the Read head it is recirculated and written again; it can also be sent out over the Transfer Bus. The Erase Head then destroys what was recorded.

rewind		Reposition tape by moving the tape in a backward direction.

S

Term	Symbol	Explanation
Search Control Location		Every channel on each General Storage Drum has a special location in which 6 characters of search-sequence control data: Overflow Address, or End of File (Stop) code can be stored. A channel's Search Control Location is its 101st 6-character group, and is the last location read in each drum revolution.
Search Validation Character		A character of control data that is stored in the first (highest-order) or last (lowest order) character position of each blockette (of multiblockette Items) on magnetic tape. As its name implies, this character is used to validate comparisons in tape search operations. By devising an ordered system of these characters, corresponding blockettes within Items in a file can be similarly "addressed." In tape search equal operations, the actual data compared in the blockette and the blockette's Search Validation Character must match the Item or Sub-Item Identifier to produce a "find." In tape search unequal operations, the actual data compared in the blockette must mismatch the corresponding data in the Item or Sub-Item Identifier, and the blockette's Search Validation Character must match that in the Item or Sub-Item Identifier. In this way specific blockettes in multiblockette Items (and only those blockettes) can produce results in the search. If the Search Validation Character in an Item or Sub-Item Identifier is an Ignore code, the search operation becomes valid for all blockettes on tape. In effect the Search Validation Character in each blockette is ignored. Depending on the setting of a two-position switch, the Tape Unit will detect either the first or the last character of a blockette on tape as the Search Validation Character. (Forward direction assumed: first character means lowest-order character.) Appropriate programming required, therefore, dependent on switch setting to be used in reading the tape.
Selector		A relay that is used as a two-way, electrically operated switch; it allows a programmer to route machine functions

Term	Symbol	Explanation
		in one of two directions, based upon the presence or absence of control conditions, negative results, or any other controllable machine function.
serial		To handle one character (or bit) after another using the same device; as opposed to parallel where all characters (or bits) are handled simultaneously, each having a separate device.
serializer		Devices which accept a character in parallel by bits and expel it serial by bits.
Shift Revolver	SRV	An addressable 12-character revolver on the High Speed Drum used to store Shift Words; can be used by either a Program Step or an Instruction Word. Addressable (998) in Program Control Storage References, but only as a Destination. (SRV is automatically referred to by Program Control when each section of the Shift Word stored in SRV is needed.)
Shift Word		U V W uxx vyy wzz u, v, and w indicate type of shift: 0 = no shift 1 = right end-around 2 = left end-off 3 = right end-off and xx, yy, and zz are the number of places to be shifted. In internally-stored programs uxx is used for V_1, vyy for V_2 and wzz for R. In plugboard-defined programs patchcord wiring determines how each section is used.
Sign position		The low-order character position in any Arithmetic Register or in any location specified by a Word Address.

S

Term	Symbol	Explanation
Source Address		An address specifying the location from which data can be obtained.
Space code		A computer character (0000001) used to represent a space or blank area, as on drum or in buffer.
Special Character	S	The lowest-order or right-most character of an Operation Code. Used to extend or modify the basic operation specified by Instruction Words.
Special Character Out		When the Special Character in an Instruction Word is Q-Y, a pulse is emitted from a correspondingly labelled SPECIAL CHARACTER OUT hub (Q-Y) on the Program Control Plugboard, just prior to the initiation of the next Instruction Word; can be used, for example, to initiate Condition Compare or Clear BTB to Ignore Sub-Steps. (Since this Sub-Instruction does not interrupt the internal program, the SPECIAL CHARACTER OUT hub should not be wired to STEP IN hubs on the plugboard.)
Special Character Register	SR	A one-character register which receives the Special Character during the execution of Instruction Words. SR is referred to during addition and subtraction at the time the Process is executed; it is referred to in multiplication and division after the result is stored. In all other cases, SR is referred to just before beginning the next Instruction Word.
SPECIAL OUT		A signal produced by an I/O Unit that has been placed "on demand" and is READY. It signifies that at least one of the (HS) I/O-C lines of that I/O Unit are energized.

Term	Symbol	Explanation
staticizer		A device which accepts a character in serial by bits and expels it in parallel by bits.
Storage Address Register	SAR	A three-character register which is not addressable by a computer program, but which holds a Program Control Storage Address while the location that address specifies is being found and the storage reference completed. Also used in I/O Control Instruction Words.
storage reference		Usually, the action whereby a memory location specified by a computer address is found, and data is placed in or received from that location. Each storage reference actually involves two locations: in general one is programmed, and the other (the intermediate or buffer location) is automatically determined by the instruction being executed.
stored program or (internally-stored program)		A sequence of Instruction Words which defines a computer program.
Sub-Instruction		An operation specified by one of the values of the Special Character S.
Sub-Item		A term used principally in connection with magnetic tapes: a collection of from 1 up to 120 alphanumeric non-ignore characters, not necessarily contiguous that represents a minor file-entry; a collection of data in an Item other than the prime set of identifying data. Sub-Items thus contain "trailer" information (i.e., information in addition to that specified by the prime set of data.) In Tape Search operations, the actual data in a Sub-Item and the Search Validation Character for the blockette containing the Sub-Item are regarded as the Sub-Item.

S

Item	Symbol	Explanation
Sub-Item Identifier		A collection of from 1 up to 120 alphanumeric, non-ignore characters, not necessarily contiguous, which are a copy of a set of data contained in a Sub-Item, or a copy of the Sub-Item itself; should be programmed as a 120-character alphanumeric quantity; is used to locate data in a particular blockette: contains (a) the actual data to be compared in locating the Sub-Item, (ignore cores in every other character position), and (b) a Search Validation Character corresponding to the one in the blockette containing the Sub-Item.
Sub-Step		A computer, General Storage or I/O Unit Control operation which is specified by a Program Step in addition to the basic operation (Process) defined by the Program Step. The plugboard counterpart of Sub-Instruction.
subtrahend		The quantity to be subtracted. In the UFC, V_2 in subtract processes.
sum		The result of an addition operation.

Term	Symbol	Explanation
Test Demand In or (Demand Test In)		A sequence wherein a particular I/O Unit (0-9) is tested to see whether it is READY or NOT READY for subsequent use. The I/O Unit is not put "on demand" and the result of the test (READY or NOT READY) is immediately produced. This operation is initiated in internally-stored programs via the Test Demand In Instruction Word; it is initiated in Plugboard operations when the appropriate DEMAND TEST IN hub (0-9) is pulsed.
Test Incoming Control		An Instruction Word that enables the computer program to test High Speed Input/Output-to-Computer Control Line Storage for a particular W, X, Y, or Z content. (In plugboard operations, a patchcord-wiring network is used to test (HS) I/O-C Control Line Storage.)
temporary storage		Intermediate or buffer storage; also used to mean volatile storage.
three-address		Having the property that each computer instruction not only includes a basic operation to perform, but in general, also specifies three storage locations. In UFC, two sources (one for V_1 and V_2), and one destination (for R) can be specified.
track		A narrow band on the periphery of the High-Speed Drum; the area which passes beneath a read/write head as the High-Speed Drum revolves; the equivalent of a channel on a General Storage Drum.
Track Switching		In the Model I in-out system each I/O Track is actually a pair of tracks. At any given time the computer is connected to one track of the pair and the I/O Unit is (independently) connected to the other track of the pair. Track switching reverses this assignment. In internally stored programs, a control

T

Term	Symbol	Explanation
		bit ("a") in the U-section of the 39S, and 45S Instruction Word is employed for track switching. In plugboard operations, track switching is programmed by patching a pulse out hub to the appropriate TRACK SWITCH (0-9) hub.
transfer		Move a copy of data from one location to another.
TRANSCOP	TC	The higher-order digits of an Operation Code of an Instruction Word are called a TRANSCOP code if they are a number 51-98. When this code is interpreted, the execution of Instruction Words stops and Program Control goes to the plugboard to continue the program. Specifically, Program Control goes to the Program Step whose Plugboard Step number is the same as the TRANSCOP code.
Transcop plugboard sequence		A series of Program Steps initiated when the computer executes a Transcop Instruction Word.
translate		To change information from one code to another without affecting the value. In the UFC, all I/O Units are equipped with translators so that Univac coded characters are delivered to and can be received from the Central Computer without any translation or conversion time being required in the Central Computer.

Term	Symbol	Explanation
U-Address	U	The three highest-order characters of an Instruction Word; used in most instructions to specify the location of V_1.
U-section		The three highest-order characters in a Word Location or Computer Word.
Unconditional Jump		A computer instruction which causes a jump to be performed whenever it is executed.
UNI-BUS hubs		A combination of five hubs tied to a bus: 4 IN hubs and an OUT hub. When a signal is applied to any IN hub, the OUT hub emits. The IN hubs are diode-protected to prevent back circuits. A Uni-bus is an isolation device.
Unit Record	UR	The operational (or machine) unit of the General Storage System. A collection of contiguous alphanumeric characters of the following programmable lengths: 12, 24, 36, 48, 60, 72, 84, 96, 108, or 120 characters. From a programming standpoint, a Unit Record can be a major file-entry or merely a portion thereof. If a major file-entry, it is similar in concept to an Item on magnetic tape. If only a portion of a major file entry, the Unit Record is like the identically named business concept, unit record, which represents a business transaction. Unit Records are transmitted in and out of GSB (from and to the General Storage Drums) lowest-order character first beginning from GSB's Word 9, Character S position and extending up to the length of the Unit Record.
Unit Record Area	URA	A location of a Unit Record on a General Storage Drum. The General Storage Address of a Unit Record Area specifies the Drum Section (DS) in which the URA is found

Term	Symbol	Explanation
		the specific track (CH) in that section the starting angular position of the URA (AA), and the number of angular positions around the track the URA includes (L).
Unit Record Identifier	URI	A collection of from 1 up to 120 contiguous characters, some, but not all of which, can be Ignore Codes; used in Channel Search operations to locate a Unit Record whose address is unknown. The length of a Unit Record Identifier is the same as the Unit Record Length employed in the Channel Search operation.
Unit Record Length	URL	The number of computer words that compose a Unit Record for a particular General Storage operation. Defined by a setting (1,2,3,...9,0) of the UNIT RECORD LENGTH SELECTOR; or, if the UNIT RECORD SELECTOR is set to GSAR, by "L" of a General Storage Address.
Univac		Universal Automatic Computer
Univac Code		A system of notation in which (commonly) 64 letters, numbers, and symbols are given values in a binary scale of notation. The notation is based on the (four-bit) excess-three code employed for the digits 0-9. In this code, the decimal digit "0" is represented as 0011 (binary "three"); the decimal digit "1" is represented as 0010 (binary "four"); and in general each of the decimal digits 0-9 is represented by a binary code whose value is three greater than the actual value of the digit. To the four-bits required to represent the decimal digits 0-9 in excess-three binary code, two higher-order bits, called zone bits, were added to formulate the rest of the Univac code. That is, the digit-portion of the

Term	Symbol	Explanation
		code was expanded, and letters and symbols assigned to different values of the six-bit combinations.
Unpacking		To separate packed fields of information.

V

Term	Symbol	Explanation
V-section, or V Address	V	The 4th, 5th, and 6th most-significant characters of an Instruction Word; used in most instructions to specify the location of the operand V_2.
	V_1	The first value operated on in an instruction.
	V_2	The second value operated on in an instruction.
volatile storage		Memories whose contents are destroyed when power is lost or shut-off.

Term	Symbol	Explanation
W-section, or W Address	W	The 7th, 8th, and 9th most-significant character positions in an Instruction Word; generally used to specify the Address at which the result of the instruction is stored.
word		12 characters; the operational (or machine) unit of data in arithmetic and logical operations.
Word Address		A three-digit (i.e., numeric) Program Control Storage Address which specifies the location of a (12-character) computer word.
write		To record; i.e., to store on a magnetic medium.

Z

Term	Symbol	Explanation
Z Address	Z	A blockette Address; a Program Control Storage Address which always specifies 120 characters.
zero	0	A computer character; in excess-three notation: 1000011
zone bits		In Univac Code, the left-most 2 bits. XX 0000 Zone Bits — Excess-three Bits (numeric bits)

APPENDIX B

LIST OF ABBREVIATIONS

AA	Angular Address
AD	Add
AS	Alternate Switch
AT	Arithmetic Transfer
BTB	Block Transfer Buffer
BTP	Block Transfer Buffer Pattern
BS	Branch Storage
BT	Buffer Transfer
C - I/O	Computer to Input/Output Control Lines
CC	Channel Clear
CDR	Code Distributor Register
CDR A/N	Code Distributor Alphanumeric
CDR GP	Code Distributor Group
CH	Channel
CI	Current Instruction
CP	Compare
CS =	Channel Search Equal
CS ≠	Channel Search Unequal
DE	Demand In
DQ	Divide Store Quotient
DR	Divide Store Remainder
DS	Drum Section
FD	Function Delay
FS	Function Sequence
FS Tracks	Factor Storage Tracks
GS	General Storage
GSAR	General Storage Address Register
GSB	General Storage Buffer
GSD	General Storage Drum
GSP	General Storage Buffer Pattern
HSD	High Speed Drum
(HS) I/O - C	High Speed Input/Output to Computer Control Lines
i	Ignore Code
I/O Instruction	Input/Output Instruction
I/O Tracks	Input/Output Storage Track
I/O Units	Input/Output Units
IRV	Instruction Revolver
IRV_c	Current Instruction Revolver
IRV_n	Next Instruction Revolver
IS	Intermediate Storage
IS Tracks	Intermediate Storage Tracks
ISP	High Speed Drum Pattern
IW	Instruction Word
JN	Jump on Negative
JP	Jump on Plus
JZ	Jump on Zero
LA	Load GSAR

LN	Left Normalize
LS	Load Shift
(LS) I/O - C	Low-Speed Input/Output-to-Computer Control Lines
MK	Mask Transfer
ML	Multiply Store Lower
MS	Millisecond
MU	Multiply Store Upper
NI	Next Instruction
OP	Operation Code
OED	Operation Pulse/Enable Distributor
PR	Process Code
PAK	Program Address Counter
PS	Program Select
R	Result
RA	Register A
RB	Register B
RC	Register C
RD	Register D
RV	Revolver
S	Special Character
SAR	Storage Address Register
SB	Subtract
SK	Shift Counter
SP	Search Probe
SR	Special Character Register
SRV	Shift Revolver
SU	Substitute U
SV	Substitute V
SW	Substitute W
SZ	Suppress Left Zeros
TC	Transcop
TD	Test Demand
TI	Test Incoming Control
TS	Track Switch
U	The Location of Value 1
UJ	Unconditional Jump
UR	Unit Record
URA	Unit Record Area
URI	Unit Record Identifier
URL	Unit Record Length
V	The Location of Value 2
V_1	Value 1
V_2	Value 2
W	The Location of R
Z	Blockette Address
Δ	Space Code
μ	Microsecond
'0	Prime Zero

APPENDIX C

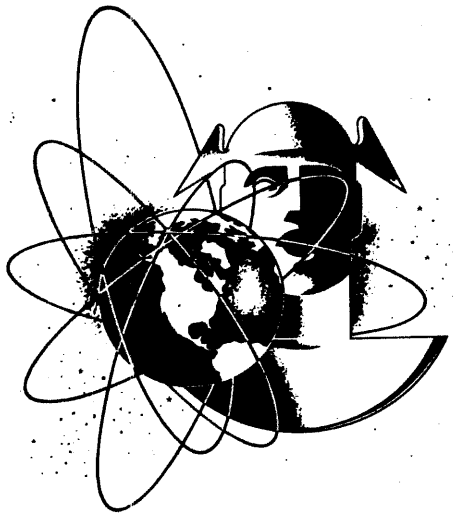
INDEX

APPENDIX D

PROGRAM CONTROL PLUGBOARD

Table D-1. Locator for Program Control Plugboard

Figure D-1. Program Control Plugboard



UNIVAC®—The FIRST Name in Electronic Computing Systems

U 1608 (EF309E) PRINTED IN U.S.A.