NAVAL TACTICAL DATA SYSTEM

PROGRAMMERS' GUIDE for COMPUTER AND EQUIPMENT

PX 1493

VOLUME

TWO

Remington Rand Univac
DIVISION OF SPERRY RAND CORPORATION
UNIVAC PARK, ST. PAUL 16, MINNESOTA

NAVY DEPARTMENT
CONTRACT: NObsr 72769

BUREAU OF SHIPS
NTDS NO. U-6464

ELECTRONICS DIVISIONS
1 MARCH 1961

SECTION D

PERIPHERAL EQUIPMENT

SECTION D1

FUNCTIONAL AND PROGRAMMING SPECIFICATIONS

FOR THE

INTERIM TAPE-SYSTEM ADAPTER

CONTENTS

CONTENTS (Continued)

ILLUSTRATIONS

TABLES

FUNCTIONAL AND PROGRAMMING SPECIFICATIONS

FOR THE INTERIM

TAPE SYSTEM ADAPTER (ITSA)

1. BASIC INFORMATION

The Interim Tape System Adapter (ITSA) is an interconnecting device which permits an AN/USQ-17 Unit Computer to operate with as many as seven Remington Rand Univac File Computer Magnetic Tape Units (MTU), Type 4950. The principal function of ITSA is to perform data conversion between the 30-bit parallel input/output of the Unit Computer and the serial-bit input/output of MTU.

Tape-system operation differs from the normal operation of most AN/USQ-17 Unit Computer peripheral equipment in the following respects:

1) The C^3 register is not used for function control; instead, the C^2 register assumes this duty since it is desirable to have a *Resume* line on the function channel to ITSA.

2) A dual level of control is involved; control of ITSA operations and control of MTU operations.

These differences require special programming techniques. This section specifies ground rules and restrictions which must be observed when programming this interim tape system.

2. MAGNETIC TAPE UNIT CHARACTERISTICS

Each Type 4950 tape unit consists of a tape handler, a 120-character buffer memory, and associated control circuitry. It uses one-half-inch Mylar-base magnetic tape. Data are recorded or read a line at a time, with one 7-bit character stored on each line. In *Read* and *Write* operations, MTU processes 120 consecutive lines or characters before the tape stops. The basic character grouping on magnetic tape is a *blockette* consisting of 120 characters.

Other MTU specifications and characteristics include:

1) *Tape Speed:* 75 inches per second.

2) *Recording Density:* 139 lines per inch.

3) *Blockette Length:* 120 characters, 0.86 inch.

4) *Interblockette Spacing:* Univac File Computer (UFC) Format - 0.5 inch; High-Speed Printer (Univac) Format - 1.0 inch with 2.4 inches after each sixth blockette.

5) *Reading Density:* 50 to 160 lines per inch.

6) *Bad-Spot Detection:* Photoelectric cells, transparent markers.

7) *Number of Blockettes Per Reel:* Maximum 20,300 blockettes on 2400-foot reel with 0.5-inch blockette spacing. Automatic counter limit is 20,000 blockettes.

8) *Automatic Terminal Condition Detection:* The MTU can detect the following logical conditions:

   a) *Beginning and End of Tape* - Photoelectric detection of transparent tape.

   b) *End of File and End of Data* - Detection of sentinels written on the tape.

   c) *End of Blockette Count* - Automatic blockette counter compared against preset limit.

   d) *Tape Search Find* - Compares tape blockette with identifier blockette for equality.

MTU is described in detail in PX 738, *Service Manual for Magnetic Tape Unit, Type 4950.*

3. EQUIPMENT INTERCONNECTIONS

ITSA is connected to each tape unit by a pair of signal cables. These cables convey timing pulses, control signals, and serial data. A complete description of this interface is not required for programming purposes.

Communications between ITSA and the Unit Computer require four cables (channels) and two C registers of the computer.

A. *CONTROL AND STATUS COMMUNICATION* (C^2 Register)

1) C^2 Register Output Channel - Transmits function commands from the computer to ITSA. It consists of 15 information lines, one *Output Ready* line, and one *Output Resume* line.

2) C^2 Register Input Channel - Transmits tape-system status information from ITSA

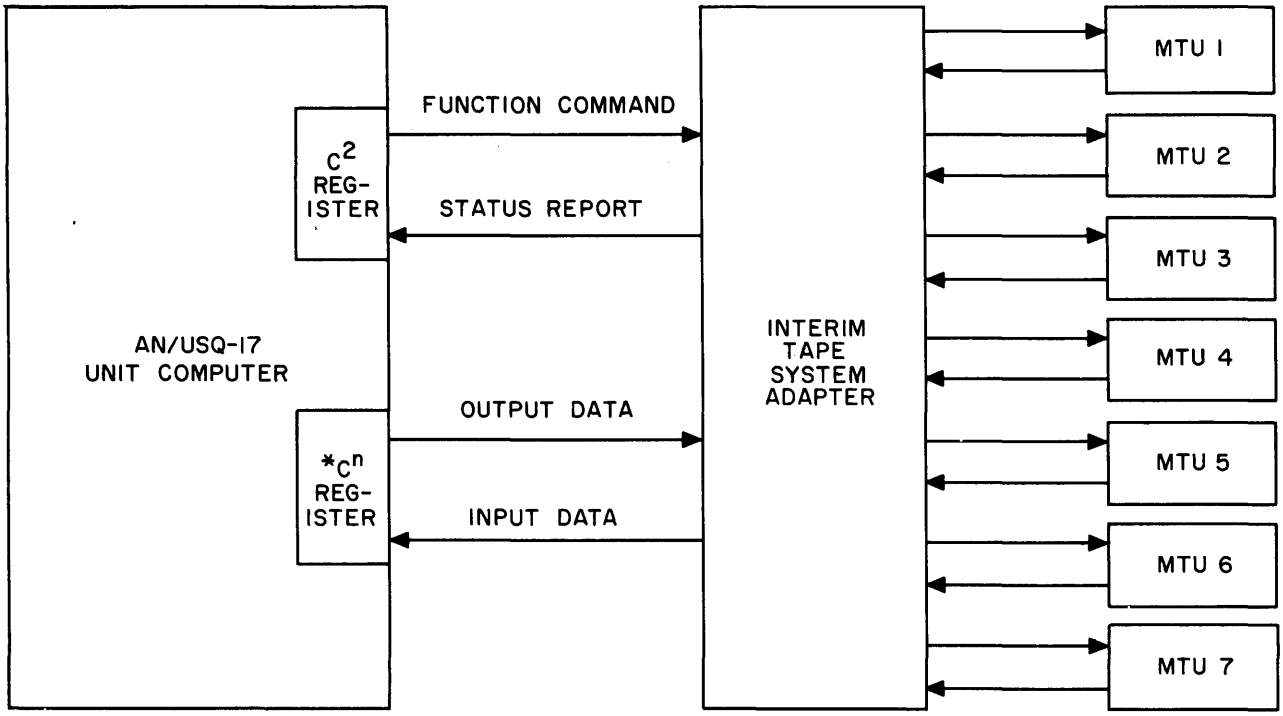
to the computer. It consists of 15 information lines, one *Input Ready* line, and one *Input Resume* line.

B. *DATA COMMUNICATION* (C^n Register, n = 4, 5, 6, or 7)

1) C^n Register Output Channel - Transmits output data from the computer to ITSA. It consists of 30 information lines, one *Output Ready* line, and one *Output Resume* line.

2) C^n Register Input Channel - Transmits input data from ITSA to the computer. It consists of 30 information lines, one *Input Ready* line, and one *Input Resume* line.

Figure D1-1 is an equipment interconnection diagram.



$*C^n = C^4, C^5, C^6,$ OR C^7

Figure D1-1. Equipment Interconnections

4. COMPUTER CONTROL OF TAPE SYSTEM

A. *FUNCTION COMMAND*

The computer sends function commands on C^2 output channel to ITSA to control tape-system operation. In general, each function word includes an instruction addressed to one or more MTU's and an instruction to ITSA. An MTU can execute the following instructions:

1) Write
2) Read Forward
3) Read Backward
4) Search Forward Equal
5) Search Backward Equal
6) Search Forward Equal or Greater Than
7) Search Backward Equal or Less Than
8) Transfer Blockette to Computer
9) Transfer Blockette to MTU
10) Wind Forward
11) Wind Forward with Interlock
12) Rewind (Wind Backward)
13) Rewind with Interlock
14) Write Check (not used in ITSA operation)

ITSA can execute two major sequences:

1) Transfer Data In (to computer)
2) Transfer Data Out (from computer)

In addition, ITSA can direct a Special Status Request to a particular MTU.

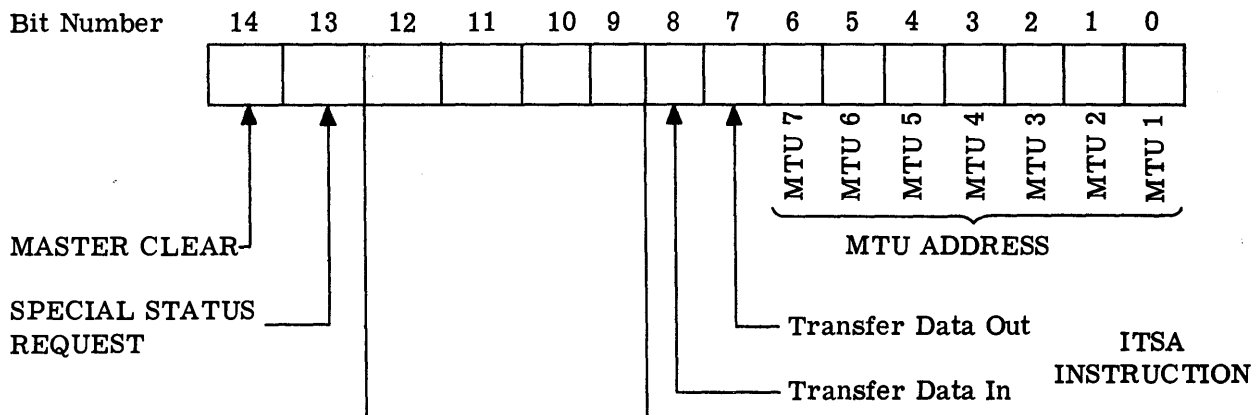
A computer to ITSA function command word has the following configuration (see Figure D1-2):

Bit 0	Address, MTU 1
Bit 1	Address, MTU 2
Bit 2	Address, MTU 3
Bit 3	Address, MTU 4
Bit 4	Address, MTU 5
Bit 5	Address, MTU 6
Bit 6	Address, MTU 7
Bit 7	Transfer Data Out
Bit 8	Transfer Data In
Bit 9	MTU Instruction (Line A)
Bit 10	MTU Instruction (Line B)
Bit 11	MTU Instruction (Line C)
Bit 12	MTU Instruction (Line D)
Bit 13	Special Status Request
Bit 14	Master Clear

The MTU instruction (bits 9 through 12) and the ITSA instruction (bits 7 and 8) *must* be matched. That is, an MTU instruction requiring data from the computer (Write, Transfer Blockette to MTU, all Search operations) must be accompanied by the ITSA instruction, *Transfer Data Out.* Similarly, the MTU instruction, *Transfer Blockette to Computer,* must be accompanied by the ITSA instruction, *Transfer Data In.* The remaining MTU instructions do not involve data transfer via ITSA.

The Master Clear command (bit 14) clears ITSA registers (as does the MANUAL CLEAR switch on ITSA). If this command is used, it should be strictly an initiatory procedure. Once tape system operations have begun, Master Clear need not be used.

The computer program may send a function command to ITSA only if ITSA is *Available* and the desired MTU is *Ready.* ITSA is considered *Available* when it is prepared to accept an instruction. It is considered *Not Available* when it is engaged in a data transfer sequence or in the exchange of control information with an MTU. After completing an instruction, ITSA indicates its



D	C	B	A	MTU INSTRUCTION
0	1	1	0	Write
0	0	0	1	Read Forward
1	0	0	1	Read Backward
0	0	1	0	Search Forward Equal
1	0	1	0	Search Backward Equal
0	0	1	1	Search Forward Equal or Greater Than
1	0	1	1	Search Backward Equal or Less Than
1	0	0	0	Transfer Blockette to Computer
1	1	1	1	Transfer Blockette to MTU
0	1	0	0	Wind Forward
0	1	0	1	Wind Forward with Interlock
1	1	0	0	Rewind
1	1	0	1	Rewind with Interlock

Figure D1-2. Function Command Word - Unit Computer to ITSA

availability by providing an Output Resume on C^2. From the time that the computer program enters C^2 with the function command until an Output Resume is received, the output channel selection and the content of C^2 must not be disturbed.

B. *STATUS REPORT*

ITSA responds to an Input Ready signal on C^2 by sending a status report and an Input Resume to the computer on C^2 input channel. The status report contains two types of information:

1) MTU Ready Indicators (bits 0 through 6) - There is one Ready Indicator bit for each MTU. If the *Ready* bit of a particular MTU is *one,* that unit has completed its previous instruction and is prepared to accept a new instruction. The computer program must establish that a unit is *Ready* before it may send an instruction to that unit. These bits are included in *each* status report to the computer.

2) MTU Special Status (bits 7 through 10) - Each MTU is able to detect the following terminal conditions:

   a) Beginning of Tape

   b) End of Tape

   c) End of File

   d) End of Data

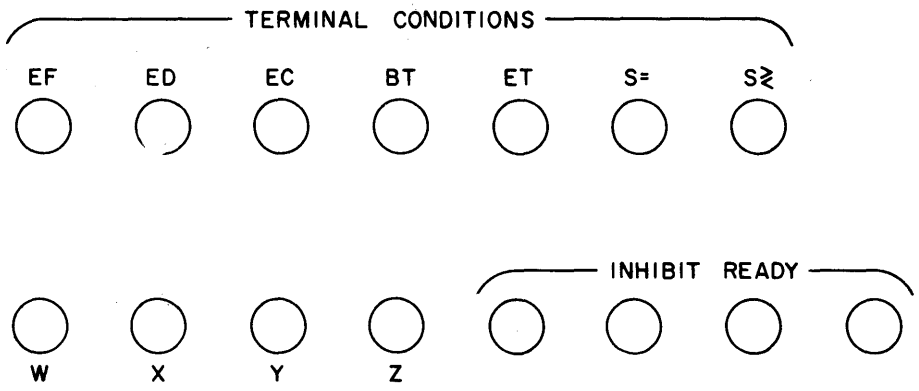
   e) End of Blockette Count

   f) Search Equal Satisfied

   g) Search Unequal Satisfied

Any four of these conditions may be reported to ITSA by making connections to the W, X, Y, or Z jacks (see Figure D1-3) on the MTU control panel.

Assigning a particular terminal condition to a particular line (W, X, Y, or Z) is the choice of the programmer. A terminal condition line may be connected to either a special status line (W, X, Y, or Z) or an Inhibit-Ready jack on the MTU panel. (In the latter case, presence of the terminal condition prevents MTU from becoming *Ready.*) Special status conditions will be included in the status report to the computer only if MTU received a Special Status Request on the previous command. In this case, bit 14 will also be set.



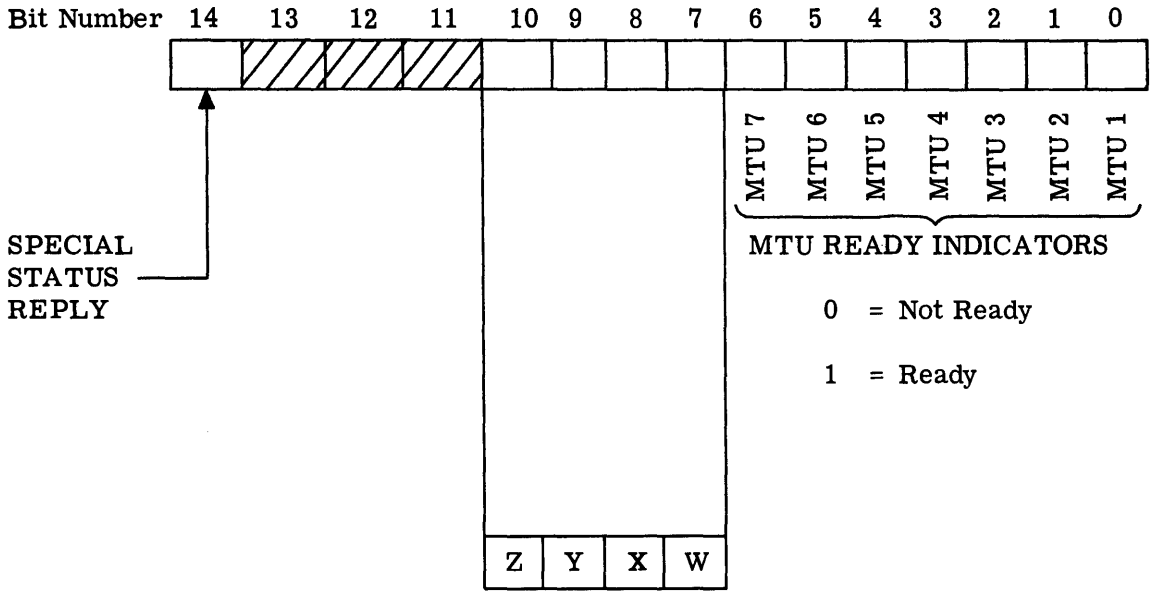
A TERMINAL CONDITION CAN BE PATCHED EITHER TO ONE OF THE W, X, Y, OR Z JACKS, OR TO AN INHIBIT READY JACK

EF	END OF FILE	ET	END OF TAPE
ED	END OF DATA	S=	SEARCH EQUAL SATISFIED
EC	END OF BLOCKETTE COUNT	S≷	SEARCH UNEQUAL SATISFIED
BT	BEGINNING OF TAPE		

Figure D1-3. MTU Control Panel - Plugboard

The Status word sent to the computer has the following configuration (see Figure D1-4):

Bit 0	Ready Indicator, MTU 1
Bit 1	Ready Indicator, MTU 2
Bit 2	Ready Indicator, MTU 3
Bit 3	Ready Indicator, MTU 4
Bit 4	Ready Indicator, MTU 5
Bit 5	Ready Indicator, MTU 6
Bit 6	Ready Indicator, MTU 7
Bit 7	Special Status, W line
Bit 8	Special Status, X line
Bit 9	Special Status, Y line



Any 4 of 7 MTU-detected terminal conditions may be plugged at the MTU control panel into the W, X, Y, Z lines. The conditions are:

1) Beginning of Tape
2) End of Tape
3) End of File
4) End of Data
5) End of Blockette Count
6) Search Equal Satisfied
7) Search Unequal Satisfied

Figure D1-4. Status Report Word - ITSA to Unit Computer

Bit 10	Special Status, Z line
Bits 11-13	Not Used
Bit 14	Special Status Reply

C. *PROGRAM REQUIREMENTS*

When sending function commands to or requesting status reports from ITSA (via the C^2 register), the computer program must follow these rules:

1) C^2 must be empty (the d^2-designator = 0) before C^2 input or output channel may be selected. There are two reasons for this rule:

   a) It is desired to produce an Input Ready signal when C^2 input channel is selected; this will occur only if $d^2 = 0$.

   b) It is desired to suppress an Output Ready signal when C^2 output channel is selected; this will occur only if $d^2 = 0$. Subsequent entry of the function command into C^2 will generate Output Ready.

   The d^2-designator may be set to zero by executing a Store-C^2 instruction.

2) After a function command is placed in C^2, it must remain until an Output Resume is sent from ITSA indicating that ITSA is again *Available*. The program may test for receipt of an Output Resume (test for $d^2 = 0$) by executing an Input Jump or an Output Jump instruction. The content of C^2 should not be altered nor should the output channel be disabled until after the Output Resume has been received.

3) ITSA must always be *Available* before the program may send another function command. If ITSA has not replied with an Output Resume to the previous command, a new command should not be sent. If a new command is illegally entered into C^2 when ITSA is *Not Available*, it will be ignored.

4) The computer program must determine that a particular MTU is *Ready* before that MTU may be instructed to operate. This is determined by examining bits 0 through 6 of the status-report word from C^2 input. An MTU whose *Ready* bit is *one* is eligible to accept an instruction.

D. *CONTROL PROGRAMMING EXAMPLES*

This part of this section describes the program steps necessary to 1) determine MTU readi-

ness, 2) send a function command, and 3) request MTU special status.

The following procedure is used to determine MTU readiness:

1) ITSA is known to be *Available* from previous programming.
2) Store C^2 (this sets $d^2 = 0$).
3) Select C^2 input channel (this also sends Input Ready).
4) Wait until C^2 becomes full (Input Resume sets $d^2 = 1$).
5) Disable C^2 input channel.
6) Store C^2 to obtain status report.
7) Examine READY indicator of desired MTU.
8) If MTU is *Not Ready,* it may be desirable to repeat this check (return to step 3) until MTU becomes *Ready.*

The following procedure is used to send a function command to ITSA:

1) ITSA is known to be *Available*, and MTU readiness has been determined by previous programming.
2) Store C^2 (this sets $d^2 = 0$).
3) Select C^2 output channel.
4) Enter function-command word into C^2 (this sets $d^2 = 1$ and sends Output Ready).
5) Wait until C^2 becomes empty (Output Resume sets $d^2 = 0$) or, if preferred, other programs may be performed, provided they do not disturb C^2 or the output-channel selection.
6) When C^2 is empty ($d^2 = 0$), disable C^2 output channel.
7) ITSA is now *Available*, and C^2 may be used for another function command or for a status report.

The following procedure is used to obtain a status report on a specific MTU:

1) ITSA is known to be *Available*, and MTU readiness has been determined by previous programming.
2) Store C^2 (this sets $d^2 = 0$).

3) Select C^2 output channel.

4) Enter C^2 with function word containing Special Status Request (bit 13) and proper MTU address bit.

5) Wait until C^2 becomes empty ($d^2 = 0$).

6) Disable C^2 output.

7) Store C^2.

8) Select C^2 input channel.

9) Wait until C^2 becomes full ($d^2 = 1$).

10) Disable C^2 input channel.

11) Store C^2 to obtain status report (Special Status is in bits 7 through 10).

12) Examine MTU Special Status and proceed with program.

5. DATA FORMAT AND SENTINEL DETECTION

A. *FORMAT*

A blockette (120 characters) is the unit of data transfer between the computer and MTU. In MTU, a character is represented by six information bits and one parity bit.

The parity bit is adjusted to yield an odd number of *ones* in a character. In the computer, however, a character has no parity bit and is thus represented by six bits. ITSA inserts a parity bit into each output data character and deletes a parity bit from each input data character.

MTU is designed to process data consisting of Univac-coded (excess-three) characters; however, other codes or uncoded binary data may be processed if the End-of-File and End-of-Data detection features are used with caution. (These detection features are discussed in a subsequent section.)

A blockette data transfer between computer and ITSA consists of 24 thirty-bit words. Each word contains five characters which are identified as follows:

Character 0	Bits 0 through 5
Character 1	Bits 6 through 11

Character 2	Bits 12 through 17
Character 3	Bits 18 through 23
Character 4	Bits 24 through 29

The computer program initiates a 24-word buffer (120 characters) for all data transfers to or from ITSA; that is, a buffer is initiated only with operations that include a Transfer Data In or Transfer Data Out instruction to ITSA. The relationship between character positions in computer memory and line positions of the written blockette on magnetic tape is specified below. Word 0 is the first word of the computer buffer list and word 23 is the last (see Figure D1-5).

1st line of tape blockette = Word 23, Character 4

2nd line of tape blockette = Word 23, Character 3

3rd line of tape blockette = Word 23, Character 2

. . .

. . .

. . .

118th line of tape blockette = Word 0, Character 2

119th line of tape blockette = Word 0, Character 1

120th line of tape blockette = Word 0, Character 0

The relative position of characters on magnetic tape is significant if a printout is to be made on a High-Speed Printer.

B. *END-OF-FILE AND END-OF-DATA DETECTION*

A file is a programming unit used for over-all data organization and may consist of any number of blockettes. The tape unit detects the End-of-File condition by examining each blockette for the presence of a sentinel which accompanies magnetic tape data. The End-of-File sentinel consists of a minimum of 12 consecutive *Z* characters (111100 in Univac code) in a blockette. When the End-of-File condition is detected, it is presented to ITSA on one of the W, X, Y, or Z lines (if plug-patched on the MTU panel) and is reported to the computer if a Status-Request command is addressed to the MTU.

The End-of-Data sentinel is detected and reported in exactly the same manner. It consists of

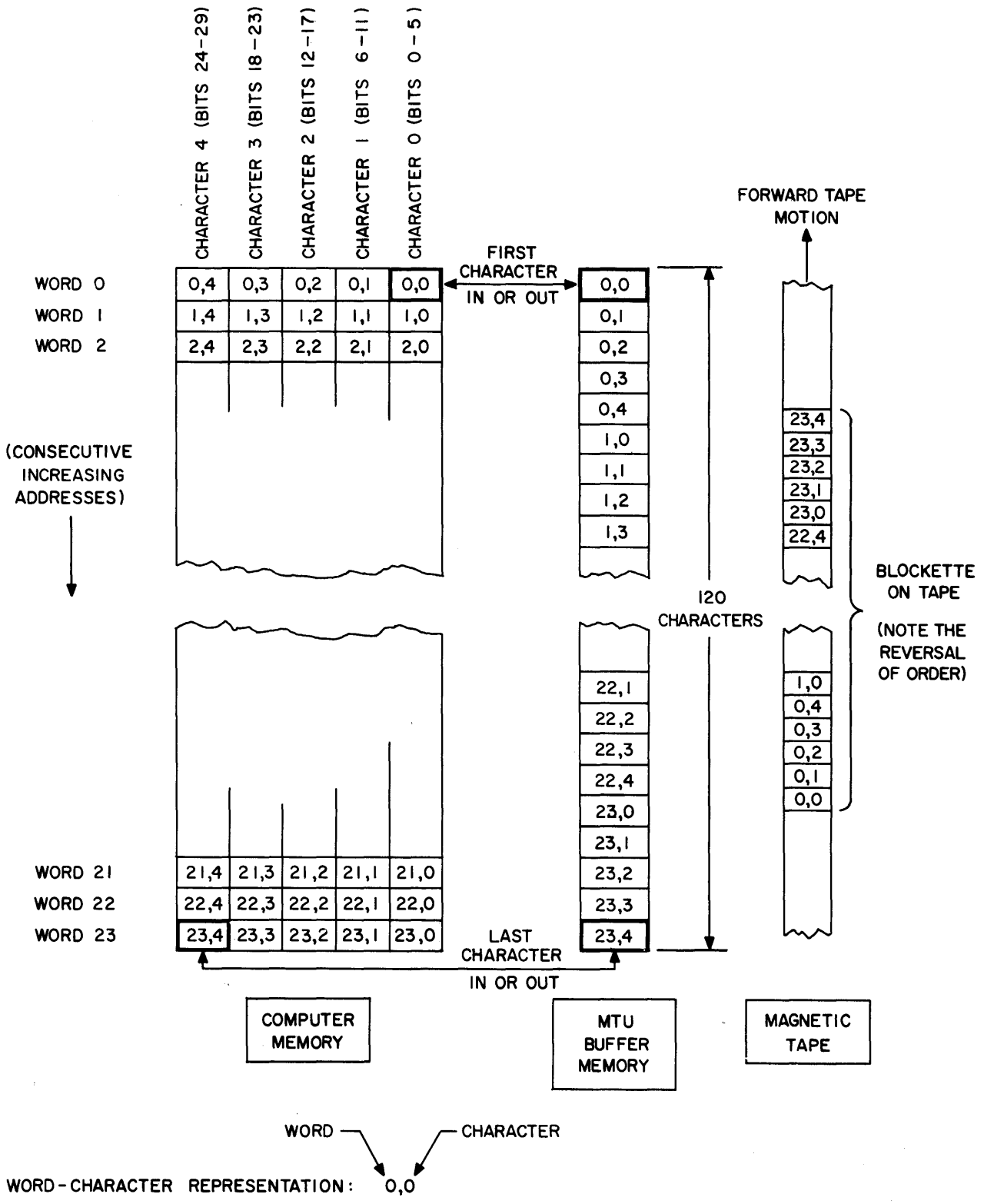


Figure D1-5. Corresponding Character Positions in Computer and MTU

a minimum of 12 consecutive % characters (111101 in Univac code) in a blockette.

If Univac coding is not used for tape data, it is possible for these sentinels to occur in a random combination of data characters and, therefore, be falsely detected by the tape unit. A suggested procedure, which will obviate false detection is described below:

1) Write each End-of-File and End-of-Data sentinel as one *entire blockette* of sentinel characters.

2) Whenever End of File or End of Data is reported in reply to a Status Request, read in the last blockette and verify that it consists entirely of sentinel characters. If not, regard it as a false detection.

6. TAPE SYSTEM OPERATIONS

This subsection describes, from a programming viewpoint, the 13 tape-system operations.

Before the computer program may command an operation, it must make preliminary checks to determine availability of ITSA and readiness of the desired tape unit(s). In addition, a Status Request should be made for MTU-detected terminal (Special Status) conditions.

A. *WRITE*

In the *Write* operation, one blockette of data is transferred from the computer, via ITSA, to MTU buffer memory. Upon completion of the blockette transfer, ITSA sends an Output Resume and becomes *Available*, but MTU does not become *Ready* until it has recorded the contents of buffer memory on magnetic tape.

A Write instruction to MTU and a Transfer Data Out instruction to ITSA comprise the function command that initiates this operation. The Write instruction may be addressed to one or more tape units; however, if multiple units are simultaneously addressed, the computer must check the readiness and special status of each MTU before initiating the operation.

A Write operation should *not* be attempted if MTU is positioned at End of Tape. In addition, it is not possible to write a blockette in the middle of a previously written tape without destroying information in the blockettes that follow; writing must be done sequentially from beginning of tape.

The following procedure is suggested for a Write operation:

1) Determine that MTU is *Ready*.

2) Be certain that MTU special status is not End of Tape.

3) Select C^n output channel.

4) Initiate 24-word output buffer on C^n.

5) Send function command to ITSA as explained in the Control Programming Examples of Subsection 4.

B. *READ FORWARD*

In the *Read Forward* operation, the next blockette of data is read from the tape (as tape moves in a forward direction) and stored in MTU buffer memory. A subsequent Transfer Blockette to Computer instruction is required to obtain the data from MTU buffer memory. Tape movement is stopped at the end of the blockette which has been read. ITSA becomes *Available* almost immediately, but MTU does not become *Ready* until its internal Read sequence is completed.

A Read Forward instruction to MTU and no-data-transfer instruction to ITSA comprise the function command that initiates this operation. If desired, the Read instruction may be addressed simultaneously to several tape units since no input data transfer takes place.

A *Read Forward* operation should not be attempted if MTU is positioned at End of Data or End of Tape since this results in an MTU error condition.

The following procedure is suggested for a *Read Forward* operation:

1) Determine that MTU is *Ready*.

2) Be certain that MTU special status is not End of Data or End of Tape.

3) Send function command on C^2. (Refer to Control Programming Examples of Subsection 4.)

4) When MTU again becomes *Ready*, the computer program may command Transfer Blockette to Computer. The Transfer Blockette to Computer operation is discussed in a subsequent paragraph.

During each *Read Forward* operation, MTU scrutinizes the blockette for End-of-File or End-of-Data sentinels. If a sentinel is detected, it is reported to the computer in response to a Status-Request command; however, if no Status Request is received before the next MTU instruction which requires tape motion, the special status information is lost.

C. *READ BACKWARD*

In most respects, the *Read Backward* operation is similar to the Read Forward operation; however, tape is initially positioned at the end of the blockette. Tape movement is in the reverse direction, and after the read is completed, the tape is positioned at the beginning of the blockette.

A Read Backward instruction to MTU and no-data-transfer instruction to ITSA comprise the function command that initiates this operation.

A Read Backward operation should not be attempted if MTU is positioned at Beginning of Tape since this results in an MTU error condition.

The following procedure is suggested for a Read Backward operation:

1) Determine that MTU is Ready.
2) Be certain that MTU special status is not Beginning of Tape.
3) Send function command on C^2.
4) When MTU again becomes *Ready*, the computer program may command a Transfer Blockette to Computer operation.

The Read Backward operation *does not* alter the order in which the characters of a blockette are received by the computer. The blockette is always presented to the computer in the same format, whether it is read forward or backward.

The automatic sentinel detection feature is the same as in Read Forward.

D. *SEARCH FORWARD EQUAL*

Search operations require that the computer prepare an identifier blockette which is transferred to MTU during the operation and compared with blockettes written on tape.

In *Search Forward* the computer identifier blockette is first loaded into MTU buffer memory; ITSA then reverts to the *Available* state and MTU is released to perform the following sequence of events:

1) The magnetic tape begins moving in a forward direction, and each tape blockette is compared character-by-character against the identifier stored in MTU buffer memory.
2) When the comparator finds 120 characters within a tape blockette equal to the 120 characters stored in MTU buffer, a Search Equal Satisfied signal is energized. This

signal is available to the computer, via ITSA, if this condition has been plugged into one of the W, X, Y, or Z special status lines at the MTU control panel.

3) When a *find* occurs, as described above, the tape blockette is read into MTU buffer memory but is not transferred to the computer. A subsequent Transfer Blockette to Computer operation is required to transfer data from MTU buffer memory to the computer.

4) If End of File, End of Data, or End of Tape is detected before a *find* occurs, the search ends. This signals the program that *no find* has occurred.

5) After a *find*, the tape remains positioned immediately following the blockette that satisfied the search.

Ignore codes (000000 character) may be included in the identifier blockette. An ignore code suppresses the comparison for that character, that is, any tape character is equal to an ignore code. This ability of the ignore code to suppress comparison allows the identifying field to be from 1 to 120 characters in length and to occupy any position within the blockette.

A Search Forward Equal instruction to MTU and a Transfer Data Out instruction to ITSA comprise the function command that initiates this operation. If desired, a Search instruction may be simultaneously addressed to several tape units. Search Forward Equal should not be attempted if MTU is positioned at End of Data or End of Tape.

The following procedure is suggested for a Search Forward Equal operation:

1) Determine that MTU is *Ready*.

2) Be certain that MTU special status is not End of Data or End of Tape.

3) Select C^n output channel.

4) Initiate 24-word output buffer on C^n. (This buffer area must contain a previously prepared identifier blockette.)

5) Send function command on C^2.

6) When C^2 Output Resume indicates ITSA *Available*, wait for MTU to again become *Ready*.

7) Request special status from MTU to determine if a *find* occurred.

8) If a *find* occurred, the program may command a Transfer Blockette to Computer operation.

E. *SEARCH BACKWARD EQUAL*

The *Search Backward Equal* operation is identical to Search Forward Equal except that the direction of tape movement is reversed. In Search Backward Equal, the Beginning-of-Tape condition becomes the *no find* indication.

A Search Backward Equal instruction to MTU and a Transfer Data Out instruction to ITSA comprise the function command that initiates this operation. Search Backward Equal should not be attempted if the MTU is positioned at Beginning of Tape.

F. *SEARCH FORWARD EQUAL OR GREATER THAN*

In *Search Forward Equal or Greater Than* operations, the sequence of events is identical to Search Forward Equal except that the comparator is looking for a blockette on tape which is either 1) greater than the identifier or 2) equal to the identifier. In addition, the first character position of the tape blockette (Character 4 of Word 23) requires special processing. This position contains the Search Validation Character which must be *equal* to the Search Validation Character in the identifier in order for a *find* to occur. If no comparison to the Search Validation Character is desired, an ignore code is placed in this position of the identifier.

A second difference exists in that after a *find*, the tape remains positioned at the beginning of the blockette just found. If a *find* occurs, the Search Equal Satisfied or the Search Unequal-Satisfied signal is sent to the computer in response to a request for special status.

A Search Forward Equal or Greater Than instruction to MTU and a Transfer Data Out instruction to ITSA comprise the function command that initiates this operation.

G. *SEARCH BACKWARD EQUAL OR LESS THAN*

The Search Backward Equal or Less Than operation is identical to Search Forward Equal or Greater Than except that the direction of tape movement is reversed and the comparator checks for a tape blockette equal to or less than the identifier value in the buffer. The function of the Search Validation Character is identical.

A Search Backward Equal or Less Than instruction to MTU and a Transfer Data Out instruction to ITSA comprise the function command that initiates this operation.

H. *TRANSFER BLOCKETTE TO COMPUTER*

In the *Transfer Blockette to Computer* operation, information in the MTU buffer memory

is transmitted, via ITSA, to the computer. No tape movement is involved in this operation. ITSA becomes *Available* and MTU becomes *Ready* as soon as the blockette transfer has been completed. The Transfer Blockette to Computer operation is normally used to obtain MTU data following a Read or a Search operation.

A Transfer Blockette to Computer instruction to MTU and a Transfer Data In instruction to ITSA comprise the function command that initiates this operation. Only *one* MTU may be addressed in this function command; if more than one unit is addressed, the data received by ITSA will be meaningless.

The following procedure is suggested for a Transfer Blockette to Computer operation:

1) Determine that MTU is *Ready.*
2) Initiate 24-word input buffer on C^n.
3) Select C^n input channel.
4) Send function command on C^2.
5) Wait until ITSA becomes *Available.*
6) Input data blockette may now be processed.

I. *TRANSFER BLOCKETTE TO MTU*

In *Transfer Blockette to MTU* operation, data in the computer output buffer list are transmitted, via ITSA, to MTU buffer memory. No tape movement is involved in this operation. ITSA becomes *Available* and MTU becomes *Ready* after the blockette transfer is completed. This operation has limited utility except in maintenance programming.

A Transfer Blockette to MTU instruction to MTU and a Transfer Data Out command to ITSA comprise the function command that initiates this operation. This instruction may be simultaneously addressed to several units.

The following procedure is suggested for a Transfer Blockette to MTU operation:

1) Determine that MTU is *Ready.*
2) Select C^n output channel.
3) Initiate 24-word output buffer on C^n.
4) Send function command on C^2.

J. *WIND FORWARD*

In the *Wind Forward* operation, tape movement begins in a forward direction and continues until End of Tape is detected. ITSA becomes *Available* immediately after it relays the instruction to MTU, but MTU does not become *Ready* until tape movement stops.

A Wind Forward instruction to MTU and no-data-transfer instruction to ITSA comprise the function command that initiates this operation. This instruction may be simultaneously addressed to several units. If Wind Forward is attempted when MTU is at End of Tape, MTU will "hang up" and remain *Not Ready*.

The following procedure is suggested for a Wind Forward operation:

1) Determine that MTU is *Ready*.
2) Send function command on C^2.

K. *WIND FORWARD WITH INTERLOCK*

Wind Forward with Interlock is identical to Wind Forward except that, when the tape stops, an interlock is set which prevents further use of MTU until manually released at the tape unit's maintenance panel.

L. *REWIND*

In the *Rewind* operation, tape movement begins in a backward direction and continues until Beginning of Tape is detected. ITSA becomes *Available* immediately after it relays the instruction to MTU, but MTU does not become *Ready* until tape movement stops. During Rewind, each blockette is read to check parity and character count. If an error is detected, the MTU ERROR light is turned on, operation is suspended, and MTU remains *Not Ready*.

A Rewind instruction to MTU and no-data-transfer instruction to ITSA comprise the function command that initiates this operation. This instruction may be simultaneously addressed to several units. Rewind should not be attempted if MTU is at Beginning of Tape since this will cause MTU to "hang up" and remain *Not Ready*.

M. *REWIND WITH INTERLOCK*

Rewind with Interlock is identical to Rewind except that, when the tape stops, an interlock is set which prevents further use of MTU until manually released at the tape unit's maintenance panel.

A simplified sequence summary of the five basic tape operations appears in Figure D1-6.

7. TIMING

To facilitate efficient tape system programming, the programmer must be aware of the approximate duration of an operation. This permits program planning that takes advantage of time intervals during which the tape system is busy. From a programming standpoint, the two most significant time intervals are:

1) The interval that ITSA remains *Not Available.*

2) The interval that MTU remains *Not Ready.*

These intervals are variable depending primarily upon 1) the time the function command is issued with respect to the basic timing cycle of the system (this can cause a delay of up to five milliseconds) and 2) the possibility of a tape motion lockout delay in MTU (this can cause a delay of up to 10 milliseconds).

Table D1-1 lists the approximate minimum and maximum times required for each operation measured from the time that the function command is sent. It is assumed that no error conditions are present, tape is not at beginning or end, and no re-reads are necessary.

TABLE D1-1. OPERATION TIMES

Operation	ITSA Becomes Available		MTU Becomes Ready*	
	Min.	Max.	Min.	Max.
Write	7	12	21	36
Read	Less Than 3		26	36
Transfer	7	12	7	12
Search	7	12	* *	
Wind/Rewind	Less Than 3		* *	

All times in milliseconds, measured from time that function command is sent.

* MTU times for 0.5-inch blockette spacing. If 1.0-inch spacing is used, add approximately six milliseconds to specified read and write times.

**Tape speed is 75 inches per second.

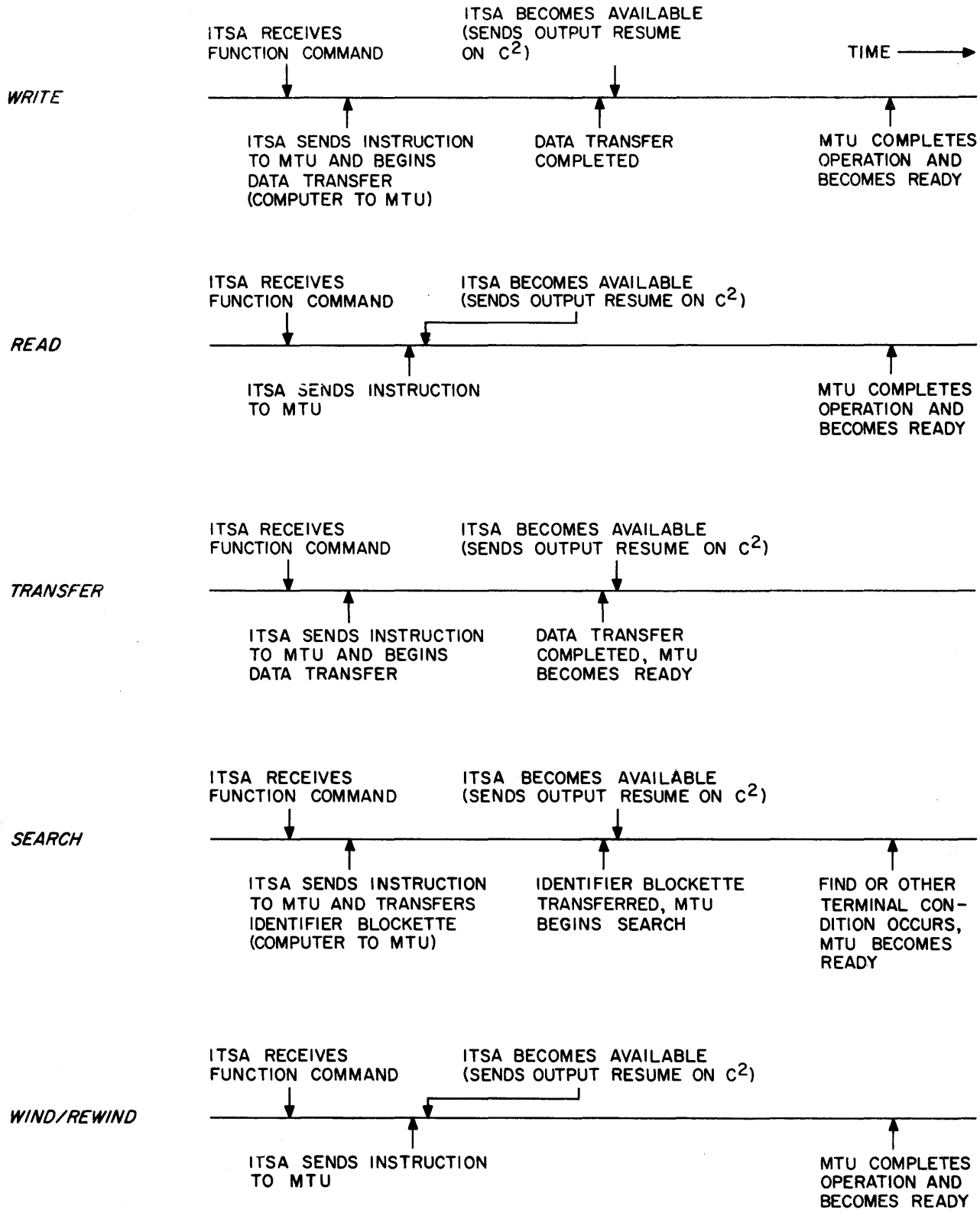


Figure D1-6. Tape System Operations

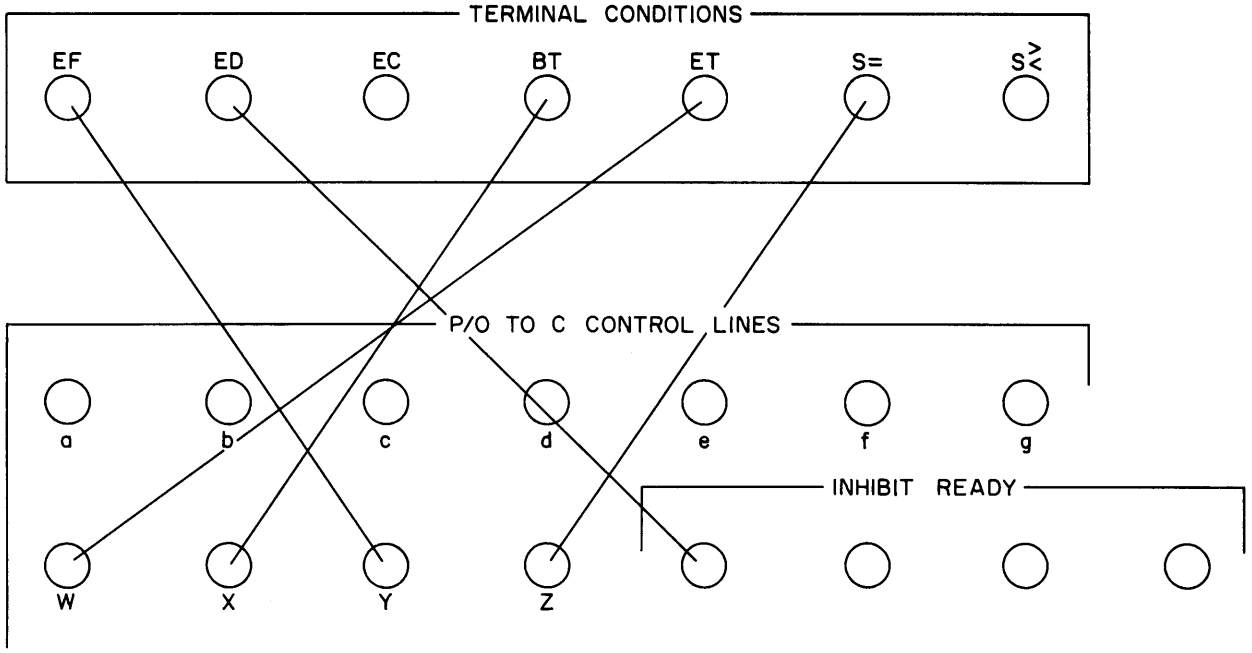
8. TAPE REVERSAL DELAY

The computer program must include a 400-millisecond delay between successive operations which result in a tape direction reversal. This delay is measured from the time that MTU becomes *Ready* after the first operation until the time the computer issues the second function command.

If this delay is not included, there is a danger of improper MTU operation due to tension-arm throwout. If MTU has been modified to eliminate this problem, program delay is not required.

9. CONVENTIONAL PLUGBOARD WIRING

Figure D1-7 depicts the conventional plugboard wiring of File Computer Magnetic Tape Units. It is essential that all programmers follow this wiring convention to assure proper *status report* interrogation on established NTDS Service Routines.



FROM	TO
END OF TAPE (ET)	CONTROL LINE W
BEGINNING OF TAPE (BT)	CONTROL LINE X
END OF FILE (EF)	CONTROL LINE Y
SEARCH EQUALS (S=)	CONTROL LINE Z
END OF DATA (ED)	ANY INHIBIT READY LINE

Figure D1-7. Conventional MTU Plugboard Wiring

For further information, refer to Section D10 of this document; Sections A1 and A4 of NTDS *Service Routines;* and NTDS Technical Note No. 246, *Univac File Computer Magnetic Tape System Control Program.*

SECTION D2

NTDS POTTER MAGNETIC TAPE SYSTEM

FOR AN/USQ-17

NTDS POTTER MAGNETIC TAPE SYSTEM
FOR AN/USQ-17

1. BASIC INFORMATION

The Potter Magnetic Tape Unit for the NTDS Unit Computer, AN/USQ-17, in conjunction with a communication register (C^1, C^2, C^4, C^5, C^6, or C^7) and the *Function* register (C^3) serves a dual purpose in the computer system. The Potter tape unit (hereafter called tape unit) may be used for 1) storage of NTDS operational programs and 2) storage of input and output data - data may be read in by an operating program or the program may dump data onto tape.

2. DESCRIPTION

Data as they appear physically on the magnetic tape are shown in Figure D2-1. There are six information bits, a parity bit, and a timing bit in each frame on magnetic tape. Thus, in a *write* operation, 30 bits of information are taken from computer storage and are written on magnetic tape as five sequential six-bit words. The tape unit adds the parity and timing bits. In a *read* operation, five sequential six-bit words are assembled by the tape unit and are sent to the computer as one 30-bit word.

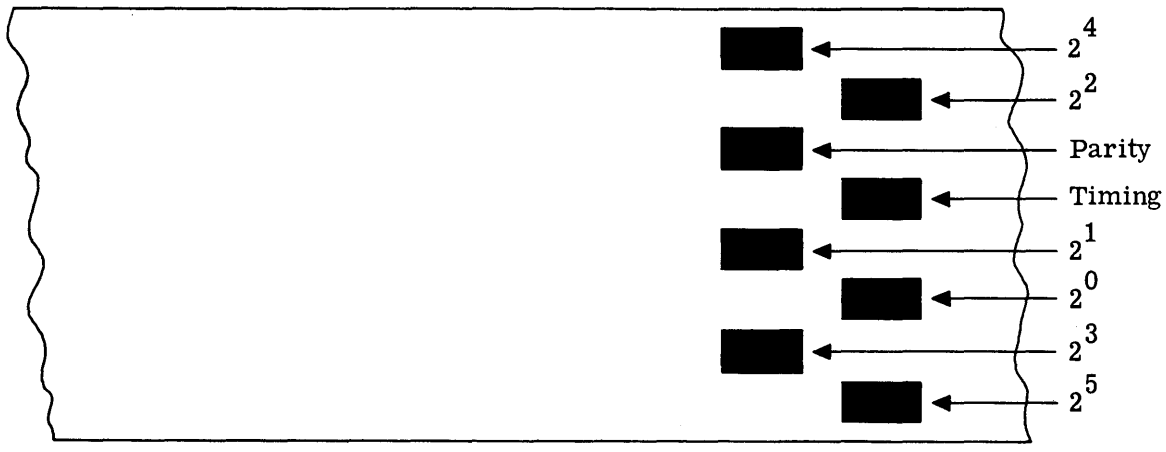


Figure D2-1. Bit Structure

Potter Magnetic Tape Unit for NTDS has the following physical and functional characteristics:

1) Data words to or from the tape unit consist of 30 bits.

2) The tape moves past the read/write head at a rate of 75 inches per second.

3) Standard tape lengths are 400 and 2400 feet.

4) Tape width is one-half inch.

5) Writing is done in Univac format with a Univac read/write head.

6) Density of writing is 128 lines per inch.

7) Word transfer rate is 1920 words per second.

8) A space of 1.2 inches is placed between blocks (interblock space) as they are written.

9) The tape unit records in variable length blocks with a minimum block length of one word and a maximum length generally dictated by core size.

10) Except for *Wind Forward* and *Wind Backward,* a command must be sent to the tape unit for each block to be written or read.

11) An erase head is placed approximately 6-1/2 inches in front of the read/write head.

12) Provision for disabling the write circuitry and erase head (by means of a switch) is made.

13) The tape unit requires one input and one output data cable and one *Function* cable.

14) If several tape units are connected in parallel on one C register, information can be processed on only one unit at a time. *Wind* can be initiated on any or all units sequentially.

15) Indication of the end of the useable portion of a tape is provided. Results of this condition made by end-of-tape *write* circuitry include

    a) Completion of operation being performed followed by winding of tape in the forward direction until *Out-of-Tape* condition is reached;

    b) all information to the end of the tape is erased as tape winds.

16) Silver conductive strips approximately 25 inches in length are placed on the front and end of the tape to indicate beginning and end of tape.

3. SPECIAL FEATURES

A fold-down maintenance panel is provided on the tape unit to perform operations manually. Commands can be entered into the control register and executed at the tape unit. C-register 3 of the computer must be cleared before the manual operation is initiated.

Several indicator lights on the maintenance panel indicate errors that have occurred and conditions existing in the tape unit.

The tape unit should be *master cleared* manually at the tape-unit cabinet before a computer program which uses the tape unit is started.

If for any reason the computer is stopped and manual operations are performed at the tape-unit cabinet, the tape unit should be *master cleared* before restarting the computer.

4. DETAILED PROGRAMMING REQUIREMENTS

The tape unit is controlled by control words sent to it through C-register 3. Control words are placed in the lower 6 bits of C-register 3 and consist of three parts: tape-unit selection, Status-Request selection, and specific instruction. The output function channel must be specified in the upper three bits of C-register 3. The word as it appears in C-register 3 is arranged as shown in Figure D2-2.

Control words are of two major types: 1) request unit status and 2) request tape movement. Bit-position 3 of the control word indicates which of the two types a control word is. If bit 3 is a *one*, the control word is a status request. If bit 3 is a *zero*, the control word is a tape movement request.

A. *REQUEST UNIT STATUS*

Present status of the tape unit, if desired, can be found by placing a *one* in the Status-Request bit position (bit-position 3) of the control word. The unit number must be specified in bit-positions 4 and 5. The lower three bits of the control word, normally used for the specific instruction, are completely ignored by the tape unit when the Status-Request bit is a *one*. A status request may normally be made at any time because it does not affect operation of the tape unit in any way*.

---

*The request must *not* be made until at least 265 microseconds have elapsed since the last tape-movement-producing command was sent to the tape unit.

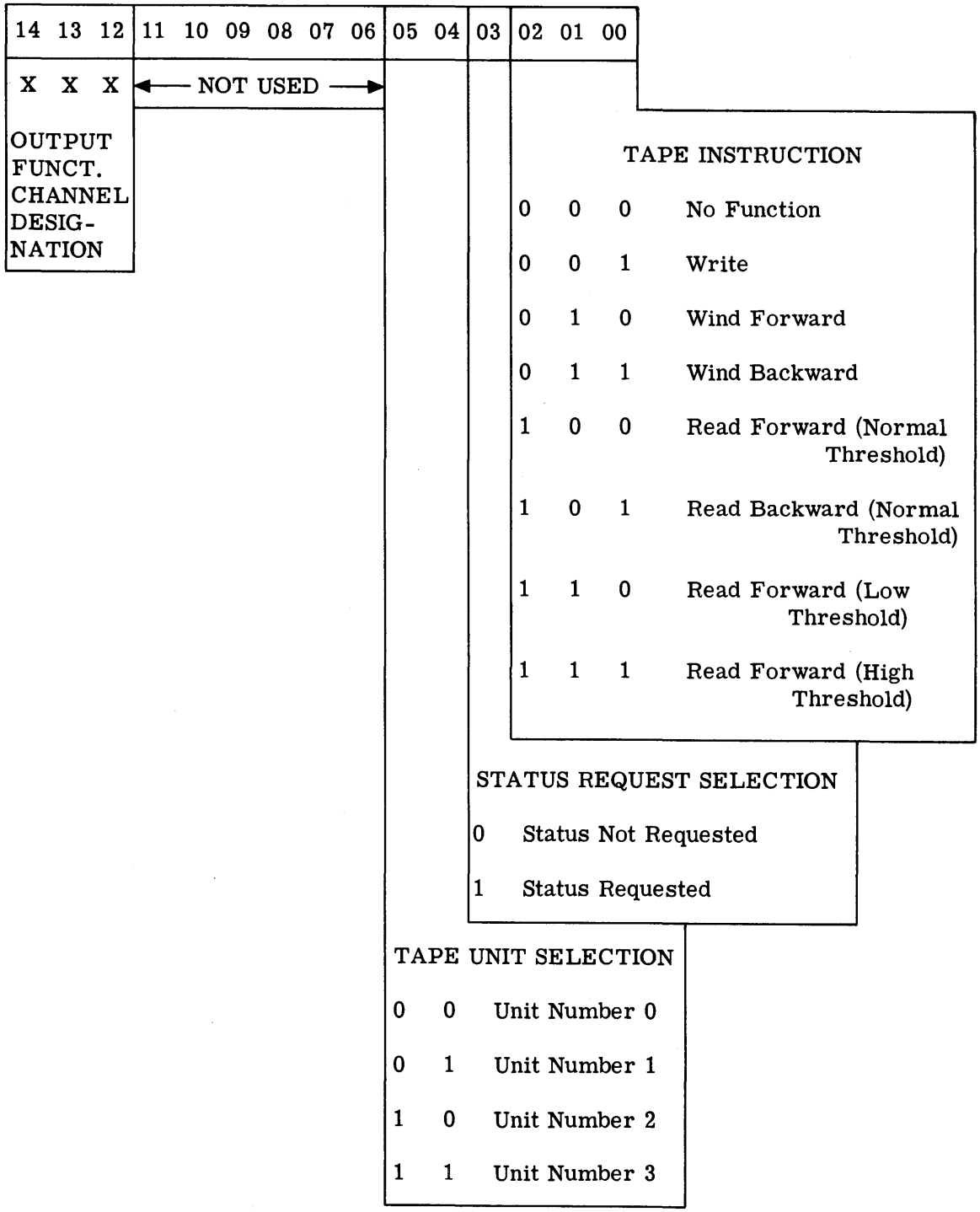


Figure D2-2. Output Function Word - Computer to Tape Unit

In order to receive the status report from the tape unit, an input function channel must be enabled. It should be enabled only after sufficient time has been allowed for the tape unit to receive the status request. A 2-instruction delay is sufficient. The status report will then come into C-register 3 and appear as shown in Figure D2-3.

The following is a description of various parts of the status reply word.

1) Bit-positions 4 and 5 indicate to which tape unit the status report applies.

2) Bit-position 3 indicates operational status of the tape unit. If bit-position 3 contains a *one,* the tape unit is ready for operation or is in operation. If it contains a *zero,* the tape unit cannot be operated until the trouble in the tape unit is corrected. Typical troubles are broken tape, no tape, tape load switch not closed, power failure, unit in test, or unit in local control. The exact trouble can be found by examining the tape unit.

3) Bit-positions 0, 1, and 2 indicate the present condition of the tape unit as follows:

   a) *Ready for Operation* (000) - indicates that the tape unit is ready to accept the next command.

   b) *Winding* (001) - indicates that the tape is winding forward or backward.

   c) *End of Block* (010) - is a condition that exists for a short time at the end of a block in a *read* or a *write* operation. It can be received as a status report only during the time the tape moves from the last word of a block to the middle of the interblock space. The tape unit is ready to accept a new command.

   d) *First Block* (011) - is a condition that exists when the tape is positioned so that the silver conductive strip on the front of the tape contacts two of the loop-arm terminals. It may indicate that the tape is stopped or is moving. (See "Additional Programming Restraints.")

   e) *End of Tape* (100) - If the indicating arm on the side of the tape reel closes the end-of-tape switch during a *Read Forward* or *Write* operation, the operation will be completed and the tape wound to the end automatically. It is during this time of winding, following the completion of the *read* or *write* and until the out-of-tape condition is reached, that the end-of-tape condition exists and will be received as a status report.

   f) *Parity Error* (101) - indicates that a parity error has occurred in a *Read* operation. The block may still be reading into the computer. When a parity-error

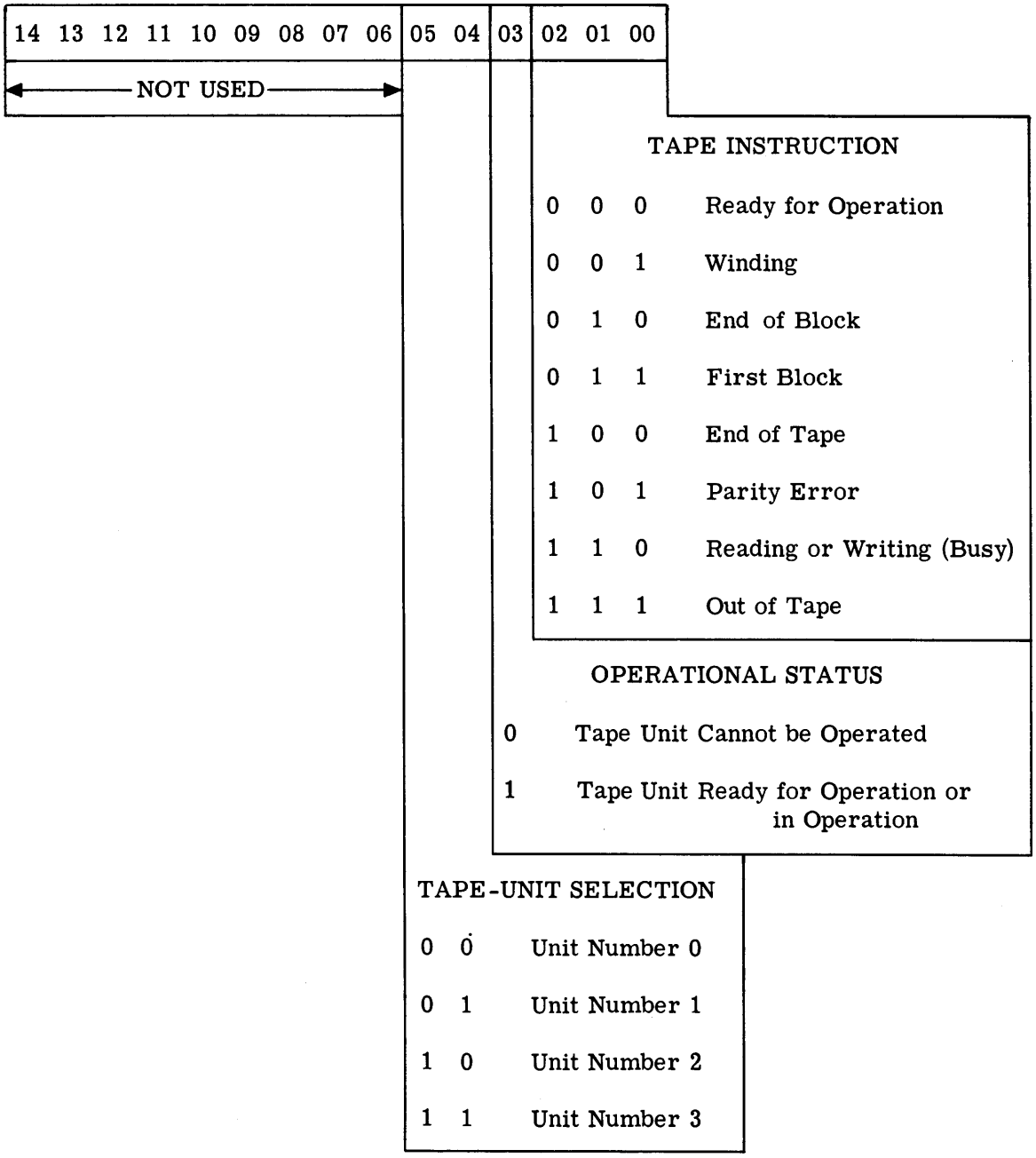


Figure D2-3. Input Function Word - Tape Unit to Computer
(Received Only After a Status Request)

status report is received by the computer, it is dropped from the status register of the tape unit. (See "Additional Programming Restraints.")

g) *Reading* or *Writing* (110) - indicates that the tape unit is engaged in reading or writing a block.

h) *Out of Tape* (111) - is a condition that exists when the tape is positioned so that the silver conductive strip on the end of the tape contacts two of the loop-arm terminals. It may indicate that the tape is stopped or is moving. (See "Additional Programming Restraints.")

Suggested procedure for requesting status is as follows:

1) Store C^3 to disable any existing *Ready*.

2) Set C^3 to output.

3) Enter C^3 with control word, the upper three bits designate the output function channel, bit-positions 4 and 5 designate the tape-unit number, and bit-position 3 contains a *one**.

4) Delay 32 microseconds (minimum).

5) Set C^3 to input.

6) Delay 32 microseconds (minimum).

7) Store C^3 as the status report.

8) Examine the status word to determine the status of the tape unit.

B. *REQUEST TAPE MOVEMENT*

If one of the tape-movement operations is desired, bit-position 3 of the control word must contain a *zero*. The unit upon which tape movement is desired must be specified in bit-positions 4 and 5, and bit-positions 0, 1, and 2 must have some nonzero code to specify the desired tape motion. In the following discussion, C^n refers to the 30-bit register used for data transfer. Included in the suggested programming procedures are sample *Function* words for the particular action being described. For purposes of illustration, these *Function* words are considered to be addressed to tape-unit number 1 with function-channel 4 being used. Codes in parentheses are in octal notation.

*The EXT-FCT poly-instruction of the CS-1 compiler cannot be used here.

(1) *Write* (40021)

This instruction initiates forward motion of the tape and prepares the tape unit to receive data from the computer. Recommended procedure for programming a *Write* operation follows:

1) Check status of the tape unit to determine if it is ready to accept another command and to ascertain that tape position does not prohibit use of the *Write* command.
2) Store C^n to drop any existing *Ready*.
3) Store C^3 to drop any existing *Ready*.
4) Set C^3 and C^n to output.
5) Enter C^3 with control word designating *Write*.
6) Initiate Output Buffer on C^n or enter each word in C^n and perform an I/O jump.
7) Continue with next operation.

(2) *Wind Forward* (40022)

This instruction initiates forward motion of tape. The tape unit cannot act on another instruction until winding is complete; that is, until Out-of-Tape condition exists.

(3) *Wind Backward* (40023)

This instruction initiates backward motion of tape. The tape unit cannot act on another instruction until winding is complete; that is, until *First Block* condition exists. Recommended procedure for programming a *Wind* operation follows:

1) Check status of the tape unit to determine if it is ready to accept another command.
2) Store C^3 to drop any existing *Ready*.
3) Set C^3 to output.
4) Enter C^3 with control word designating desired *Wind*.
5) Continue with next operation.

(4) *Read Forward - Normal Threshold* (40024).

This instruction initiates forward motion of tape and prepares the tape unit to send data to the computer. Read amplifiers are set at Normal Threshold by this instruction.

(5) *Read Backward - Normal Threshold* (40025).

This instruction initiates backward motion of tape and prepares the tape unit to send data to the computer. Read amplifiers are set at Normal Threshold by this instruction.

(6) *Read Forward - Low Threshold* (40026).

This instruction is identical to *Read Forward - Normal Threshold* except that the read amplifiers are set at a low threshold; that is, low signal levels will be read as well as the normal and high signal levels.

(7) *Read Forward - High Threshold* (40027).

This instruction is identical to *Read Forward - Normal Threshold* except that the read amplifiers are set at a high threshold; that is, only high signal levels will be read. Recommended procedure for programming a *Read* operation follows:

1) Check status of the tape unit to determine if it is ready to accept another command and to ascertain that tape position does not prohibit use of the desired *Read.*
2) Store C^3 to drop any existing *Ready.*
3) Enter C^n with a dummy word to set *Ready.*
4) Set C^3 to output and C^n to input.
5) Enter C^3 with control word designating desired *Read* operation.
6) Store C^n to set input ready (not necessary if Buffer mode is used).
7) Initiate Input Buffer on C^n or perform I/O jump on C^n and store each word of the block.
8) Continue with next operation.

5. ADDITIONAL PROGRAMMING RESTRAINTS

A. *FIRST BLOCK AND OUT-OF-TAPE CONDITIONS*

The *First Block* or *Out-of-Tape* condition remains in the status register for a short time after a command has initiated tape motion from either of these conditions; therefore, a *First Block* or *Out-of-Tape* status reply may be received under two conditions: a) the tape is stopped and there is no command in the control register, or b) the tape is moving and there is a command in the control register. When a *First Block* or *Out-of-Tape* status report is received from the tape unit, a program must ascertain whether a command was sent to the tape

unit which has started to move the tape from the *First Block* or *Out-of-Tape* condition. If this is not done and ***First Block*** or *Out-of-Tape* is assumed to mean that the tape is stopped with no instruction in the control register, commands to the tape unit could be piled on top of each other and thus cause a program error.

B. *PARITY ERROR*

Several restrictions exist on detection of a parity error of the tape unit. If a parity error occurs in a *Read* operation, it will stay in the status register until a status request is made or until end-of-block or end-of-tape condition exists; therefore, a parity error must be detected by the computer while the block is being read.

If detection of parity errors is desired, one of the following procedures must be used:

1) The interrupt request feature of the tape unit is used. In this case, information can be read either by buffer or by successive I/O jumps.

2) The interrupt request feature of the tape unit is not used. In this case, information must be read by buffer and the status checked continuously while the block is being read.

C. *TIMING RESTRICTIONS*

1) If the *First-Block* condition exists and a command which will initiate forward tape motion is sent to the tape unit, the computer must be ready to send or receive data within 1.5 seconds.

2) If other than a *First-Block* condition exists and a command which will initiate tape motion in either direction is sent to the tape unit, the computer must be ready to send or receive data within 10 milliseconds.

3) Words of information must be available within 400 microseconds of each other once information flow has begun.

4) When 700 microseconds have elapsed since the receipt of a word, the tape unit assumes that the end of the block has been reached and prepares to stop. No additional information can be recorded on the tape or read from it until a new instruction enters the control register.

5) Once a tape-movement instruction has been sent to the tape unit, a delay of 250 microseconds is required before another instruction is issued or a status request made. If an instruction is not held by the tape unit for 250 microseconds it may not

be recognized. This delay is not necessary *following* a status request.

6. INTERRUPT REQUEST

The status control section of the tape unit contains provisions for sending an interrupt request to the tape unit when certain conditions exist in the tape unit.

Conditions which result in an interrupt request are

1) Tape Break - the tape has been broken or the loop arms have activated the micro-switch stops. This is indicated if bit-position 3 of the status report contains a *zero.*
2) End-of-Block - the end-of-block space has been reached.
3) Parity Error - a parity error has occurred in reading a block.

If an interrupt request is made, there must be a subroutine to handle it. The interrupt request should be handled by using addresses 00011, 00012, and 00013 in the manner recommended for interrupt requests. The subroutine to handle the interrupt request must include a status request to the tape unit involved so that the interrupt will be cleared. The status request should determine which of the three conditions has caused the interrupt request.

If it is desired that no interrupt requests be allowed to originate from the tape unit, this tape unit function can be disconnected by means of the Interrupt OFF/ON switch mounted on the tape-unit cabinet.

7. SAMPLE PROGRAMS FOR POTTER MAGNETIC TAPE SYSTEM FOR AN/USQ-17

The following assumptions are made for the sample programs:

1) $C^n = C^4$
2) Input Channel 2 of C^4 is used
3) Channel 4 of C^3 is used
4) $Core^0$ = a core address
5) N = number of words to be written or read
6) Tape unit number 1 is used

A. *STATUS REQUEST*

$STAT^0 \longrightarrow$ ENTRY

$\longrightarrow$ $STR \bullet C^3 \bullet A \longrightarrow$ Drop C^3 Ready

	→	SEL-CHAN•C^3 OUT		
	→	ENT•C^3•40030	→	Status Request Code
	→	SEL-CHAN•C^3IN4	→	Delay and Select Channels
	→	ENT•B^0• 0	→	Delay
	→	ENT•B^0• 0	→	Delay
	→	STR•C^3•Q	→	Store Status Report
	→	ENT•LP•10•ANOT	→	Check Ready Bit
	→	JP•NOTOPER	→	Unit Cannot Be Operated
	→	ENT•LP•7		
	→	ADD•A•STAT 1	→	Jump Address
	→	JP•A		
$STAT^1$	→	JP•READY	→	Ready For Operation
	→	JP•WINDING	→	Winding
	→	JP•END BLK	→	End of Block
	→	JP•FIRST BLK	→	First Block
	→	JP•END TAPE	→	End of Tape
	→	JP•PARERR	→	Parity Error
	→	JP•REWRIT	→	Reading or Writing
	→	JP•TAPEOUT	→	Out of Tape

B. *WRITE*

$WRITE^0$	→	ENTRY		
	→	RJP•$STAT^0$	→	Check Status
	→	STR•C^4•A	→	Drop C^4 Ready
	→	STR•C^3•A	→	Drop C^3 Ready
	→	SEL-CHAN•C^3OUT•C^4OUT		
	→	ENT•C^3•40021	→	Write Code

WRITE^1 ⟶ ENT•C^4•W(CORE^0+B^6) ⟶ Enter Words

WRITE^2 ⟶ JP•WRITE^2•C^4 FULL

⟶ BSK•B^6•[N-1] ⟶ Word Count

⟶ JP•WRITE^1

⟶ EXIT

C. *WIND*

WIND^0 ⟶ ENTRY

⟶ RJP•STAT^0 ⟶ Check Status

⟶ STR•C^3•A ⟶ Drop C^3 Ready

⟶ SEL-CHAN•C^3 OUT

⟶ ENT•C^3•40023 ⟶ Wind Backward Code

⟶ EXIT

D. *READ*

READ^0 ⟶ ENTRY

⟶ RJP•STAT^0 ⟶ Check Status

⟶ STR•C^3•A ⟶ Drop C^3 Ready

⟶ ENT•C^4•A ⟶ Set C^4 Ready

⟶ SEL-CHAN•C^{3}OUT•C^{4}IN^2

⟶ ENT•C^3•40024 ⟶ Read Forward Code

⟶ IN•C^4•W(READ^2) ⟶ Initiate Buffer

⟶ EXIT

READ^2 ⟶ U-TAG•CORE^0+ [N]•CORE^0 ⟶ LIMITS

SECTION D3

HIGH-SPEED PRINTER

HIGH-SPEED PRINTER

1. BASIC INFORMATION

In order to produce printed output consisting of numerical or alphabetical information, the NTDS Unit Computer (AN/USQ-17) may be programmed to utilize the Univac High-Speed Printer *on-line.*

The Univac High-Speed Printer prints data in units called blockettes (a blockette being one line of a printed page consisting of from 1 to 120 characters). The programmer arranges data in a core-memory area called a buffer. Each buffer consists of a multiple of 24 computer words containing data in *excess-three* code. Each 24-word increment equals one High-Speed Printer blockette. Table D3-1 shows the excess-three codes for the High-Speed Printer.

The computer sends information from the buffer area to the High-Speed Printer one word at a time via an adapter which transforms the word into a format acceptable to the High-Speed Printer. Computer-to-printer transmission takes place in the following sequence.

1) Signals from the computer prepare the printer and adapter to receive information.

2) A 30-bit word is sent to the adapter by the computer.

3) The high-order six bits of the 30-bit word, together with an appropriate parity bit are sent by the adapter to the printer. The six information bits will be stored at Position Zero of the printer's memory. Each of the remaining four groups of six bits is sent sequentially to the printer from the adapter in the same manner, filling Positions One through Four of the printer's memory.

4) Twenty-three more words are sent from the computer to the printer via the adapter in a similar manner. After these words have been sent, Positions 0 through 119 of the printer's memory are full (120 characters).

5) At this time the printer stops accepting information from the adapter and prints one line representing the contents of its 120 memory locations interpreted as *excess-three* characters. (The format of the representation depends on printer-plugboard wiring.)

6) After the line has been printed, the printer will again accept information from the adapter as described in steps 2) through 5).

TABLE D3-1. EXCESS-THREE CODE FOR THE HIGH-SPEED PRINTER

CHARACTER	PARITY BIT	CODE ZONE	CODE BODY	COMP DIGIT	CHARACTER	PARITY BIT	CODE ZONE	CODE BODY	COMP DIGIT
(IGNORE)	1	00	0000	5	(IGNORE)	0	10	0000	N
(IGNORE)	0	00	0001	6	(IGNORE)	1	10	0001	O
-	0	00	0010		(FF II)	1	10	0010	P
0	1	00	0011		)	0	10	0011	
1	0	00	0100		J	1	10	0100	
2	1	00	0101		K	0	10	0101	
3	1	00	0110		L	0	10	0110	
4	0	00	0111		M	1	10	0111	
5	0	00	1000		N	1	10	1000	
6	1	00	1001		O	0	10	1001	
7	1	00	1010		P	0	10	1010	
8	0	00	1011		Q	1	10	1011	
9	1	00	1100		R	0	10	1100	
'	0	00	1101		$	1	10	1101	
&	0	00	1110		*	1	10	1110	
(	1	00	1111		(FF III)	0	10	1111	M
(ML)	0	01	0000	E	(STOP)	1	11	0000	V
,	1	01	0001		(BP)	0	11	0001	W
.	1	01	0010		:	0	11	0010	
;	0	01	0011		+	1	11	0011	
A	1	01	0100		/	0	11	0100	
B	0	01	0101		S	1	11	0101	
C	0	01	0110		T	1	11	0110	
D	1	01	0111		U	0	11	0111	
E	1	01	1000		V	0	11	1000	
F	0	01	1001		W	1	11	1001	
G	0	01	1010		X	1	11	1010	
H	1	01	1011		Y	0	11	1011	
I	0	01	1100		Z	1	11	1100	
#	1	01	1101		%	0	11	1101	
(IGNORE)	1	01	1110	C	(FF IV)	0	11	1110	T
(FF I)	0	01	1111	D	UNUSED	1	11	1111	

NOTE: Non-printable symbols are indicated in the CHARACTER column by parentheses. The alternate symbols which will be printed in the Computer Digit mode of operation are listed in the COMP DIGIT column.

2. PROGRAMMING PROCEDURE

The procedure used in programming the computer to produce output on the printer is as follows:

1) Store the C register through which information is to be sent to the adapter. This clears the **d** designator and insures the data *Ready* signal will be "down".

2) Select the output channels for the function and for the information.

3) Send the *CLEAR* External Function code to the adapter*. This results in a *Master Clear* of the adapter and a General Clear of the printer so that printer-memory positions will be filled starting with Position 0.

4) Store C-register 3. This removes the C^3 *Ready* signal in preparation for the next External Function code to C^3.

5) Send the *Print* External Function code. This results in an Initial Start being sent from the adapter to the printer. The printer responds with a Printer *Read Start.* This signifies that the printer is ready to accept data and enables data-transfer circuits in the adapter.

6) Store C-register 3. This removes C^3 *Ready* signal in preparation for the next External Function code to C^3.

7) Send information to be printed. Note that each 30-bit word must contain five *excess-three* characters and that 24 words produce one printed line on the printer.

3. PROGRAMMING REQUIREMENTS

The External Function codes used to address the High-Speed-Printer adapter are as illustrated in Figure D3-1.

The External Function word contained in C-register 3 may be considered made up of three parts: the function-code portion which includes bit-positions 0, 1, and 2, the unit selector composed of bit-positions 3 and 4, and the output-function channel is designated in bit-positions 12, 13, and 14.

Bit-position 0 of the External Function word holds the *Clear* command, bit-position 2 the *Print* command, and bit-position 1 must be *zero* for recognition of External Function codes by the adapter.

*It is assumed the C^3 Ready is "down" as a result of the execution of a previous routine.

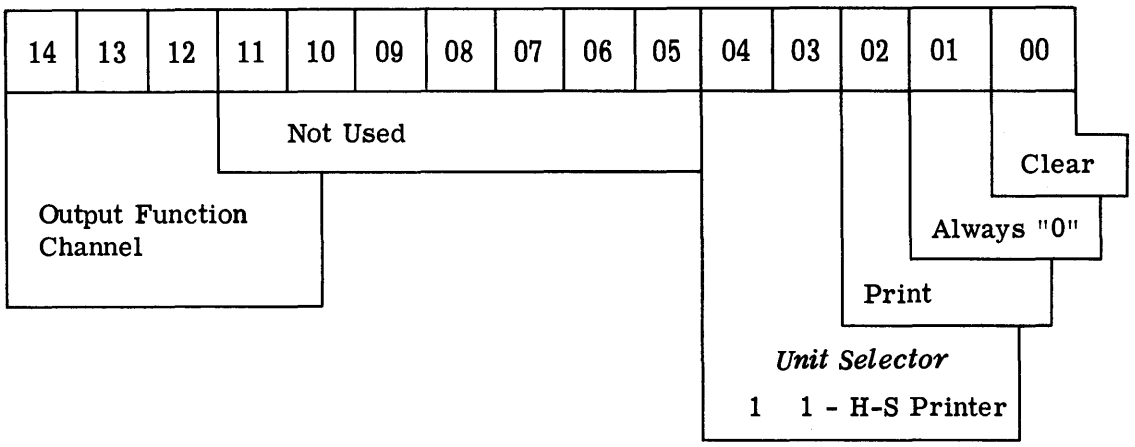


Figure D3-1. C^3 External Function Word

Bit-positions 3 and 4, the unit selector, must both be *ones* to allow the adapter function decoder to differentiate unit selectors of the High-Speed-Printer adapter from unit selectors of other output devices.

The output-function-channel designators select the output function channel.

4. PROGRAMMING AND OPERATING RESTRICTIONS

The High-Speed Printer has a normal print cycle of 100 milliseconds; however, in the event a fast feed is programmed, a print cycle can take as long as three seconds. Because the program has no way of knowing when the printer has completed a print cycle, some programming and operating rules have been established.

1) The *Clear* and *Print* External Function codes must not have a repetition rate of less than three seconds.

2) If an illegitimate Stop occurs, such as a computer Forced Stop, and the computer console is *Master Cleared,* the printer and adapter must be cleared either manually or by appropriate subroutine.

3) If printer stops due to recognizing a Stop code in the message, the *Clear* and *Print* External Function codes must be programmed to restart.

4) If another equipment is to be operated in parallel on the same channel as the printer, a Stop code must be included in the last blockette before the exit from the Print Routine. This disables the printer from accepting data.

5) There should be a 1 to 1 correspondence between the *Enable* Printer codes (*Clear* and *Print*) and the Stop codes.

6) To enable the printer, the *Clear* and *Print* External Function codes must be programmed in close proximity to each other (within 100 milliseconds).

5. PRINTER CONTROL PROGRAMS

A. *ENABLE PRINTER*

Table D3-2 shows a program that enables the High-Speed Printer. Both the CS-1 mnemonics and their generated machine-code instructions are shown.

The program is written with the following assumptions:

1) Information will be sent to the printer on channel 6;

2) Function-channel 2 is used;

3) The program is assembled at address 01000.

TABLE D3-2. PROGRAM TO ENABLE THE HIGH-SPEED PRINTER

MNEMONIC CODE	ADDRESS	ABSOLUTE CODE
EPRINT ⟶ JP • 0	01000	61000 00000
STR • B0 • W(6)	01001	16030 00006
STR • C6 • A	01002	17640 00000
SEL-CHAN • C3OUT • C6OUT	01003	11000 14160
	01004	10000 63617
	01005	57030 00003
	01006	13030 00003
EX-FCT • 2 • 31	01007	13300 20031
	01010	12000 00000
	01011	12000 00000
	01012	17310 01011
EX-FCT • 2 • 34	01013	13300 20034
	01014	12000 00000
	01015	12000 00000
	01016	17310 01015
JP • L(EPRINT)	01017	61010 01000

B. *DATA TRANSMISSION FROM UNIT COMPUTER TO ON-LINE HIGH-SPEED PRINTER*

Table D3-3 shows a program to send data to the *on-line* High-Speed Printer. Both CS-1 mnemonics and their generated machine-code instructions are shown. The program buffers 30_8 (24 decimal) words in *excess-three* code to the printer and assumes the following conditions:

1) B^7 contains the address of the first word to be buffered;

2) The program is assembled at address 01020.

TABLE D3-3. PROGRAM TO TRANSMIT DATA TO HIGH-SPEED PRINTER

MNEMONIC CODE		ADDRESS	ABSOLUTE CODE	
PRINTBU	JP • 0	01020	61000	00000
	STR • B6 • L(RESB6)	01021	16610	01032
START	RJP • CKBUFFER	01022	65000	01034
	ENT • B6 • 0	01023	12600	00000
MOVE	ENT • A • W(B7)	01024	11037	00000
	STR • A • W(BUFFER+B6)	01025	15036	01043
	ENT • B7 • B7 + 1	01026	12707	00001
	BSK • B6 • 27	01027	71600	00027
	JP • MOVE	01030	61000	01024
	OUT • C6 • W(ROGER)	01031	76630	01042
RESB6	ENT • B6 • 0	01032	12600	00000
	JP • L(PRINTBU)	01033	61010	01020
CKBUFFER	JP • 0	01034	61000	00000
CHECK	ENT • A • U(6)	01035	11010	00006
	SUB • A • L(6)AZERO	01036	21420	00006
	JP • CHECK	01037	61000	01035
SELF	JP • SELF • C6FULL	01040	62600	01040
	JP • L(CKBUFFER)	01041	61010	01034
ROGER	U-TAG •BUFFER+30•BUFFER	01042	01073	01043
BUFFER	RESERVE • 30			

Both of the preceding programs (Tables D3-2 and D3-3) are required to send data from the computer to the printer. A CS-1 **PRINT** or **PRINT-BUF** operation will cause one buffer load of data to be printed. The programmer is responsible for enabling the printer once whenever he uses either of these two operations.

SECTION D4

FUNCTIONAL SPECIFICATIONS

FOR THE

MONITORING TYPEWRITER (USQ-17)

CONTENTS

FUNCTIONAL SPECIFICATIONS FOR THE MONITORING TYPEWRITER (AN/USQ-17)

1. BASIC INFORMATION

The Monitoring Typewriter described herein is a low-speed input/output device to be used with the NTDS AN/USQ-17 Unit Computer. Inputs may be either perforated paper tape or manual keyboard entry. Output is presented as typewritten copy and, if desired, can be simultaneously punched on paper tape. The Monitoring Typewriter can be disconnected from the Unit Computer by depressing a switch and used independently as a standard electric typewriter.

The typewriter is a Flexowriter, Model FL, manufactured by Commercial Controls Corporation. Attached to the Flexowriter are a paper-tape reader and punch and switches that actuate them.

A rather standard typewriter keyboard arrangement is shown in Figure D4-1. Some keys, however, were assigned special characters or functions (including upper-case superscript numerals). A typewriter character or operation is represented on perforated tape and on communication channels by a special 6-bit code. Codes for alphabetic, numeric, and symbolic characters and for various typewriter operations are listed in Table D4-1. Special controls are incorporated for automatic tabulation and automatic carriage return. These initiate tabular or return motion of the carriage automatically after the carriage has been spaced past an adjustable, predetermined position.

The Typewriter (mounted on a cabinet which contains the typewriter, control circuitry, and a manual control panel) communicates with the Unit Computer via C-register 1.

The following sections describe functional control, output operation, and input operation of the Monitoring Typewriter. The terms *input* and *output* always imply input to the computer and output from the computer.

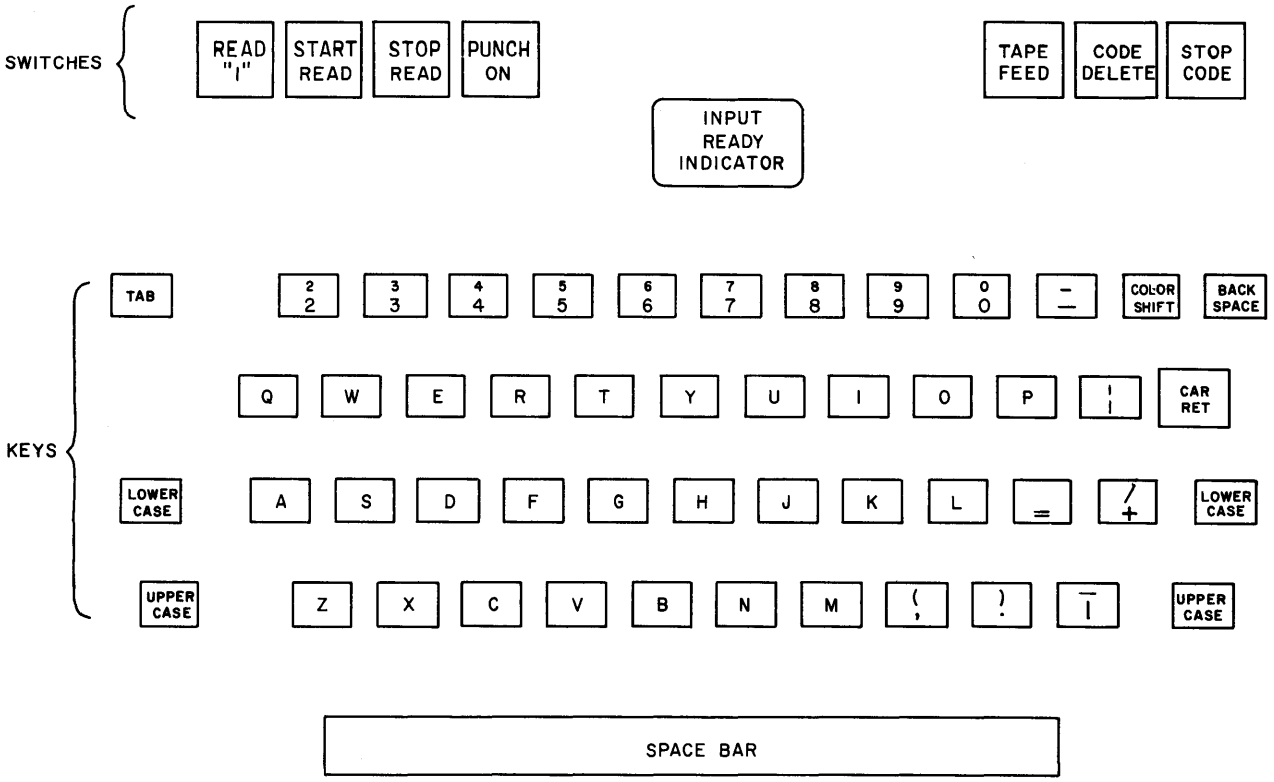


Figure D4-1. Keyboard Layout of Monitoring Typewriter

2. FUNCTIONAL CONTROL

Functional control of the Monitoring Typewriter is performed manually from the control panel or by External Function commands sent via the computer output channel.

A. *CONTROL PANEL*

The control panel of the Monitoring Typewriter cabinet includes the following switches indicators:

1) COMPUTER/MANUAL TWR toggle switch - When the switch is in the COMPUTER position the typewriter can perform input/output operations with the computer. The MANUAL TWR position disconnects the typewriter from computer control, and permits standard typewriter operation.

2) RESUME/TWR STOP SWITCH - If this switch is set to the TWR STOP position, the typewriter will withhold the resume signal on an illegal code. Moving the switch to the resume position will allow operations to continue. The IMPOSSIBLE PRINT

TABLE D4-1. TYPEWRITER CODES*

TYPE LETTER UC	TYPE LETTER LC	OCTAL
A	a	30
B	b	23
C	c	16
D	d	22
E	e	20
F	f	26
G	g	13
H	h	05
I	i	14
J	j	32
K	k	36
L	l	11
M	m	07
N	n	06
O	o	03
P	p	15
Q	q	35
R	r	12
S	s	24
T	t	01
U	u	34
V	v	17
W	w	31
X	x	27
Y	y	25
Z	z	21

TYPE LETTER UC	TYPE LETTER LC	OCTAL
₁	1	52
₂	2	74
₃	3	70
₄	4	64
₅	5	62
₆	6	66
₇	7	72
₈	8	60
₉	9	33
₀	0	37

PERFORM TYPEWRITER OPERATION	OCTAL
Space	04
Shift up	47
Shift down	57
Back space	61
Carriage return	45
Tabulator	51
Color shift	02
Code delete	77*
Stop	43*

The upper case, UC, or lower case, LC, character is typed according to the position of the type bars.

TYPE SYMBOL UC	TYPE SYMBOL LC	OCTAL
- (Superscript Minus)	- (Hyphen or Minus)	56
• (Multiply)	= (Equals)	44
/ (Virgule)	+ (Plus)	54
((Open Parens)	, (Comma)	46
) (Close Parens)	. (Period)	42
_ (Underline)	| (Absolute)	50

* Codes not used (illegal) are 00, 10, 40, 41, 53, 55, 63, 65, 67, 71, 73, 75, 76. Codes 43 and 77 are also considered illegal codes in operations with the computer.

light will come on.

B. *COMPUTER CONTROL*

The Monitoring Typewriter function control logic recognizes five External Function codes from the Unit Computer. The External Function commands are as follows:

1) ENABLE OUTPUT - This command enables Output mode, and illuminates OUTPUT indicator on the control panel. If the Input mode was active prior to receipt of this command, it is disabled by this code. It is represented by a code of 10012.

2) ENABLE INPUT - This command enables Input mode, and illuminates the INPUT indicator on the control panel. If the Output mode was active prior to receipt of this command, it is disabled by this code. It is represented by a code of 10011.

3) DISABLE MODE - This command disables the mode. It is represented by a code of 10016.

4) START MOTOR - This command turns on the motor. It is represented by a code of 10013.

5) STOP MOTOR - This command turns off the motor. It is represented by a code of 10014.

For any of the foregoing External Function commands to be recognized, the COMPUTER/MANUAL TWR switch on the control panel of the Monitoring Typewriter must be in the COMPUTER position.

3. PROGRAMMING RESTRICTIONS

1) It is necessary to disable the High-Speed Punch before sending data to the Flexowriter because of the difference in rate of acceptance of data and the fact that they are on the same output channel.

2) It is necessary to program a delay of approximately 5 milliseconds before sending a START MOTOR function code, and a delay of approximately 40 milliseconds before and after sending the appropriate function code to ENABLE OUTPUT or ENABLE INPUT.

3) It is also necessary to drop the resume on C-register 3 by storing the register before sending another function code.

4. PROGRAMMING

Tables D4-2 through D4-4 show programming codes for the Flexowriter.

TABLE D4-2. OPERATIONS TO ENABLE FLEXOWRITER FOR INPUT

MNEMONIC CODE	NOTES	ABSOLUTE CODE	
EFLEXIN JP • 0		α 61000	00000
SEL-CHAN • C^3OUT • C^1IN^2		11000	00162
		10000	77614
		51030	00003
		13030	00003
RPT • 1000	DELAY	70000	01000
ENT • B^0 • 0		12000	00000
ENT • C^3 • 10013	START MOTOR	13300	10013
RPT • 10000	DELAY	70000	10000
ENT • B^0 • 0		12000	00000
STR • C^3 • A		17340	00000
ENT • C^3 • 10011	ENABLE FLEX IN	13300	10011
RPT • 10000	DELAY	70000	10000
ENT • B^0 • 0		12000	00000
STR • C^3 • A		17340	00000
JP • EFLEXIN		61000	α

TABLE D4-3. OPERATIONS TO ENABLE FLEXOWRITER FOR CUTPUT

MNEMONIC CODE	NOTES	ABSOLUTE CODE	
EFLEXOUT JP • 0		α 61000	00000
SEL-CHAN • C^3OUT • C^1OUT		11000	00163
		10000	77614
		57030	00003
		13030	00003
RPT • 1000	DELAY	70000	01000
ENT • B^0 • 0		12000	00000
ENT • C^3 • 10013	START MOTOR	13300	10013
RPT • 10000	DELAY	70000	10000
ENT • B^0 • 0		12000	00000
STR • C^3 • A		17340	00000
ENT • C^3 • 10026	DISABLE HSP	13300	10026
RPT • 10000	DELAY	70000	10000
ENT • B^0 • 0		12000	00000
STR • C^3 • A		17340	00000
ENT • C^3 • 10012	ENABLE FLEX OUT	13300	10012
RPT • 10000	DELAY	70000	10000
ENT • B^0 • 0		12000	00000
STR • C^3 • A		17340	00000
JP • EFLEXOUT		61000	α

TABLE D4-4. OPERATIONS TO DISABLE FLEXOWRITER

MNEMONIC CODE		NOTES	ABSOLUTE CODE	
DFLEX	JP • 0		α 61000	00000
	SEL-CHAN • C^3OUT		11000	00160
			10000	77617
			57030	00003
			13030	00003
	RPT • 1000	DELAY	70000	01000
	ENT • B^0 • 0		12000	00000
	ENT • C^3 • 10016	DISABLE MODE	13300	10016
	RPT • 10000	DELAY	70000	10000
	ENT • B^0 • 0		12000	00000
	STR • C^3 • A		17340	00000
	JP • DELEX		61000	α

SECTION D5

FUNCTIONAL SPECIFICATIONS

FOR THE

MONITORING TYPEWRITER (AN/USQ-20)

CONTENTS

ILLUSTRATIONS

TABLES

FUNCTIONAL SPECIFICATIONS FOR THE
MONITORING TYPEWRITER
(AN/USQ-20)

1. BASIC INFORMATION

The Monitoring Typewriter described in this section is a low-speed input/output device to be used with the NTDS Service Test Unit Computer, AN/USQ-20. Inputs may be either perforated paper tape or manual keyboard entry. Output is presented as typewritten copy and, if desired, can be simultaneously punched on paper tape. A switch permits the Monitoring Typewriter to be disconnected from the Unit Computer and used independently as a standard electric typewriter.

The typewriter is a Flexowriter® , Model FL, manufactured by Commercial Controls Corporation. Figure D5-1 is a photograph of a Flexowriter. Attached to the Flexowriter are a paper-tape reader and punch and switches which actuate them.

Figure D5-1. Monitoring Typewriter (Flexowriter, Model FL)

The typewriter keyboard arrangement, shown in Figure D5-2, is essentially a standard keyboard. Some keys, however, were assigned special characters or functions (including uppercase superscript numerals). A typewriter character or operation is represented on perforated tape and on the communication channels by a special 6-bit code. Codes for alphabetic, numeric, and symbolic characters and for various typewriter operations are listed in Table D5-1. Special controls are incorporated for automatic tabulation and automatic carriage return. These initiate tabular or return motion of the carriage automatically after the carriage has been spaced past an adjustable, predetermined position.

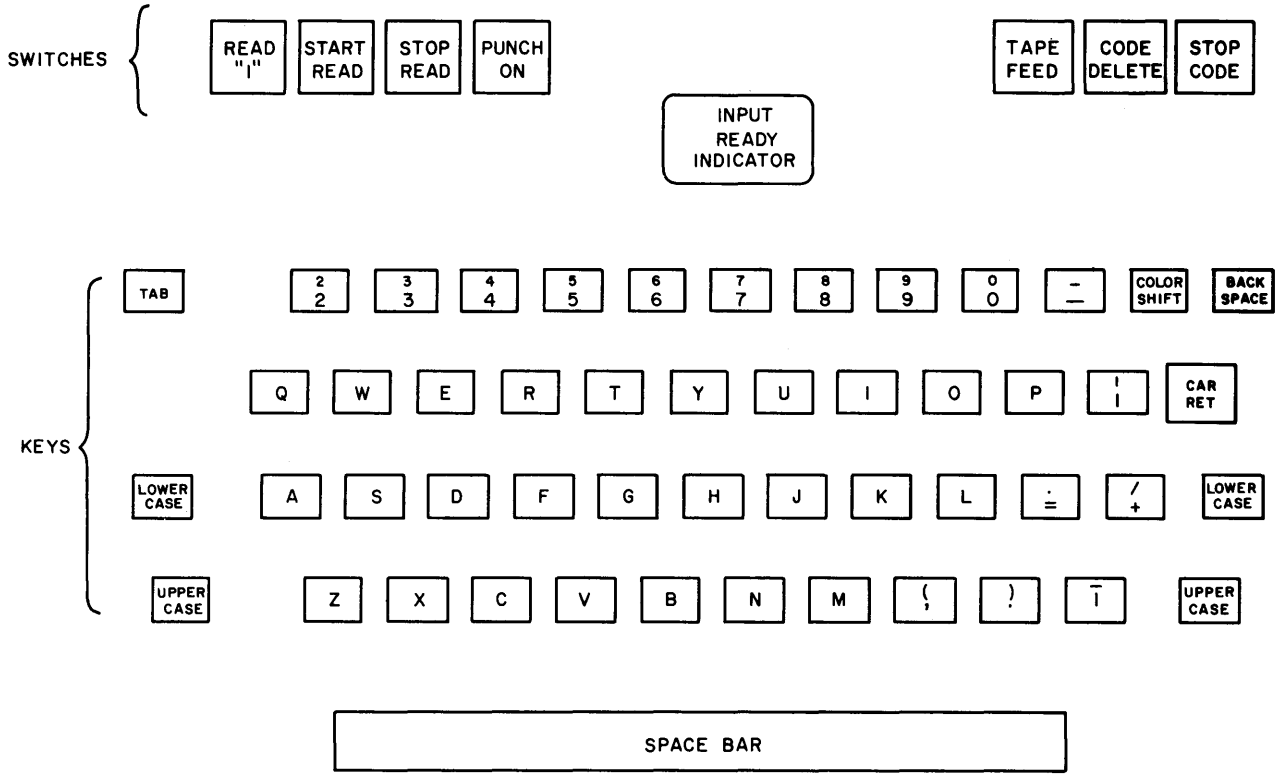


Figure D5-2. Monitoring Typewriter - Keyboard Layout

Major differences between the Monitoring Typewriter described here and the one presently used with the AN/USQ-17 Unit Computer are:

1) The input/output and function control circuitry has been redesigned to make the typewriter compatible with the AN/USQ-20 Unit Computer;

TABLE D5-1. TYPEWRITER CODES

The upper case, *UC*, or lower case, *LC*, character is typed according to the position of the type bars.

Type Letter *UC*	Type Letter *LC*	Octal
A	a	30
B	b	23
C	c	16
D	d	22
E	e	20
F	f	26
G	g	13
H	h	05
I	i	14
J	j	32
K	k	36
L	l	11
M	m	07
N	n	06
O	o	03
P	p	15
Q	q	35
R	r	12
S	s	24
T	t	01
U	u	34
V	v	17
W	w	31
X	x	27
Y	y	25
Z	z	21

Type Letter *UC*	Type Letter *LC*	Octal
!	1	52
"	2	74
#	3	70
$	4	64
%	5	62
¢	6	66
'	7	72
&	8	60
(	9	33
)	0	37

Type Symbol *UC*		Type Symbol *LC*		Octal
-	(Superscript Minus)	-	(Hyphen or Minus)	56
.	(Multiply)	=	(Equals)	44
/	(Virgule)	+	(Plus)	54
(	(Open Parens)	,	(Comma)	46
)	(Close Parens)	.	(Period)	42
_	(Underline)	|	(Absolute)	50

Perform Typewriter Operation	Octal
Space	04
Shift up	47
Shift down	57
Back space	61
Car. return	45
Tabulator	51
Color shift	02
Code delete	77*
Stop	43*

NOTE: Codes not used are: 00, 10, 40, 41, 53, 55, 63, 65, 67, 71, 73, 75, 76.
*Codes 43 and 77 are considered illegal codes in operations with the computer.

2) Function codes to start and stop the typewriter motor were eliminated (incorporated into other functions);

3) Manual controls enable Input and Output modes;

4) *Impossible Print* translator was eliminated. Illegal codes sent by the computer or read from paper tape are ignored;

5) PUNCH can be ON during the Input mode.

The typewriter (mounted on a cabinet which contains the typewriter, control circuitry, and a manual control panel) communicates with the Unit Computer via a normal input channel and a normal output channel. Figure D5-3 shows a block diagram of the Monitoring Typewriter.

The remainder of this section describes functional control, output operation, and input operation of the Monitoring Typewriter. The terms *input* and *output* always imply input to the *computer* and output from the *computer*.

2. FUNCTIONAL CONTROL

Functional control of the Monitoring Typewriter is performed manually from the control panel or by External Function commands sent via the computer output channel.

Control Panel

The control panel of the Monitoring Typewriter cabinet includes the following switches and indicators:

1) COMPUTER/MANUAL TWR toggle switch - When the switch is in the COMPUTER position the typewriter can perform input/output operations with the computer. The MANUAL TWR position disconnects the typewriter from computer control, starts the motor, and permits standard typewriter operation.

2) ENABLE OUTPUT pushbutton (with OUTPUT indicator light) - This button, when depressed, starts the motor, enables the Output mode, and illuminates the OUTPUT indicator.

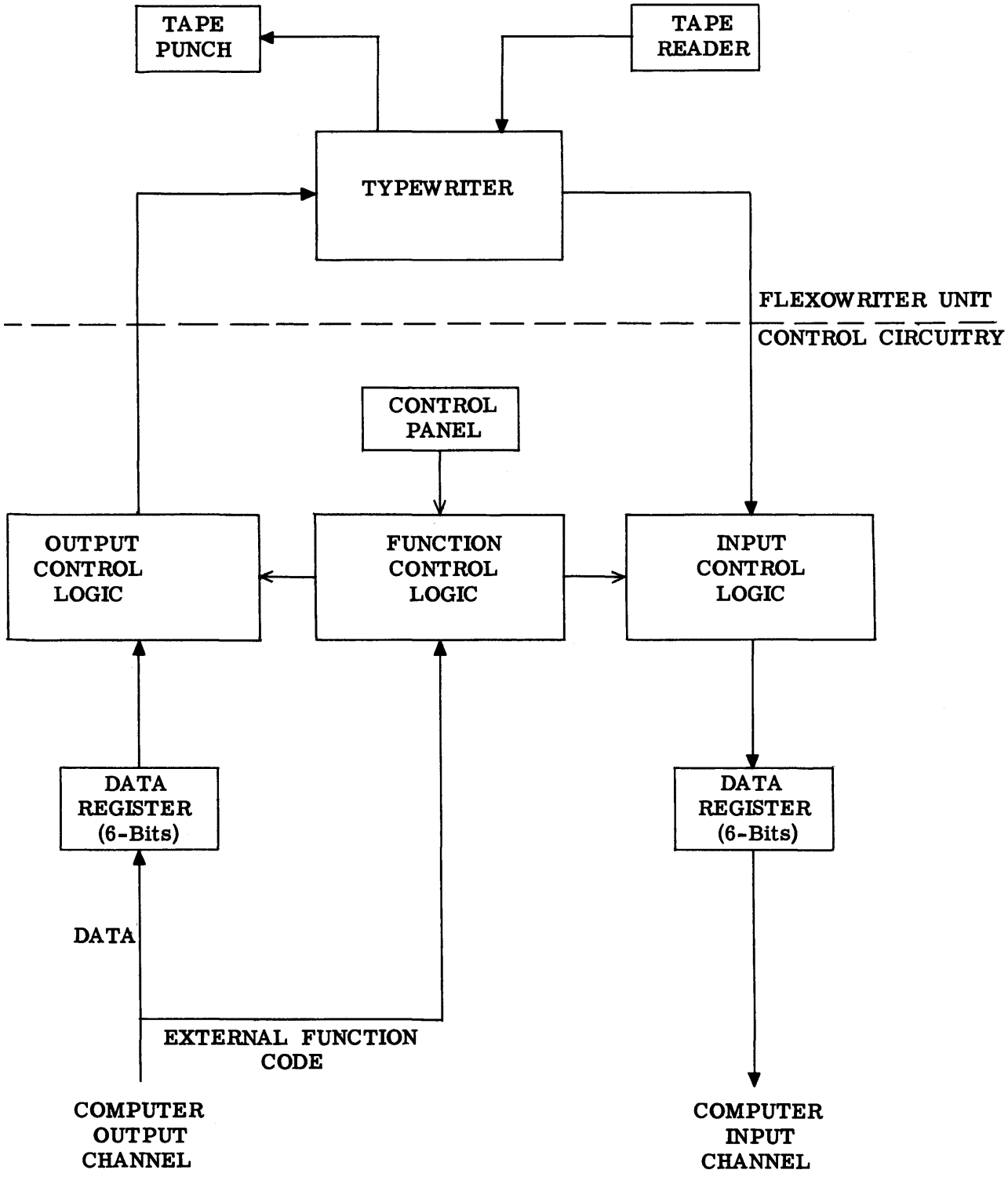


Figure D5-3. Block Diagram of Monitoring Typewriter

3) ENABLE INPUT pushbutton (with INPUT indicator light) - This button, when depressed, starts the motor, enables the Input mode, and illuminates the INPUT indicator.

4) CLEAR OUTPUT pushbutton - This button, when depressed, disables Output mode by clearing mode bit, extinguishes the indicator, stops the motor, but does not clear the data registers.

5) CLEAR INPUT pushbutton - This button, when depressed, disables Input mode by clearing mode bit, extinguishes the indicator, stops the motor, but does not clear the data registers.

6) MANUAL CLEAR pushbutton - This button, when depressed, clears all mode, control, and data bit registers. It disables any active mode and stops the motor.

7) INPUT FAULT indicator (with CLEAR FAULT button) - This indicator is illuminated when an *Input Fault* External Function code is received from the computer. While it is ON, typewriter operation is suspended. When the CLEAR FAULT button is depressed, the indicator is extinguished, and typewriter operation is resumed.

In reading the foregoing description of the functional controls, it is understood that the ON-OFF power switch on the typewriter is in the ON position.

Computer Control

The Monitoring Typewriter function control logic recognizes six External Function codes from the Unit Computer. These function codes are bit-position coded; a *one* in a particular bit position indicates what function is desired (see Figure D5-4 for format). The External Function commands are as follows:

1) *Enable Output* - This command starts typewriter motor, enables Output mode, and illuminates OUTPUT indicator. If the Input mode was active prior to receipt of this command, it is disabled by this code.

2) *Enable Input* - This command starts typewriter motor, enables Input mode, and illuminates INPUT indicator. If the Output mode was active prior to receipt of this command, it is disabled by this code.

3) *Disable Output* - This command disables the Output mode (by clearing the mode bit) and stops the motor. It does not clear the data registers.

4) *Disable Input* - This command disables the Input mode (by clearing the mode bit) and stops the motor. It does not clear the data registers.

5) *Master Clear* - This command clears all mode, control and data bit registers. It disables any active mode and stops the motor.

6) *Input Fault* - This command suspends typewriter operations and illuminates the INPUT FAULT indicator. It permits the computer program to indicate an error in input data from the Monitoring Typewriter.

For any of the foregoing External Function commands to be recognized, the COMPUTER/ MANUAL TWR switch on the control panel of the Monitoring Typewriter must be in the COMPUTER position.

Only one function bit may be set in an External Function word. Combinations are not permitted.

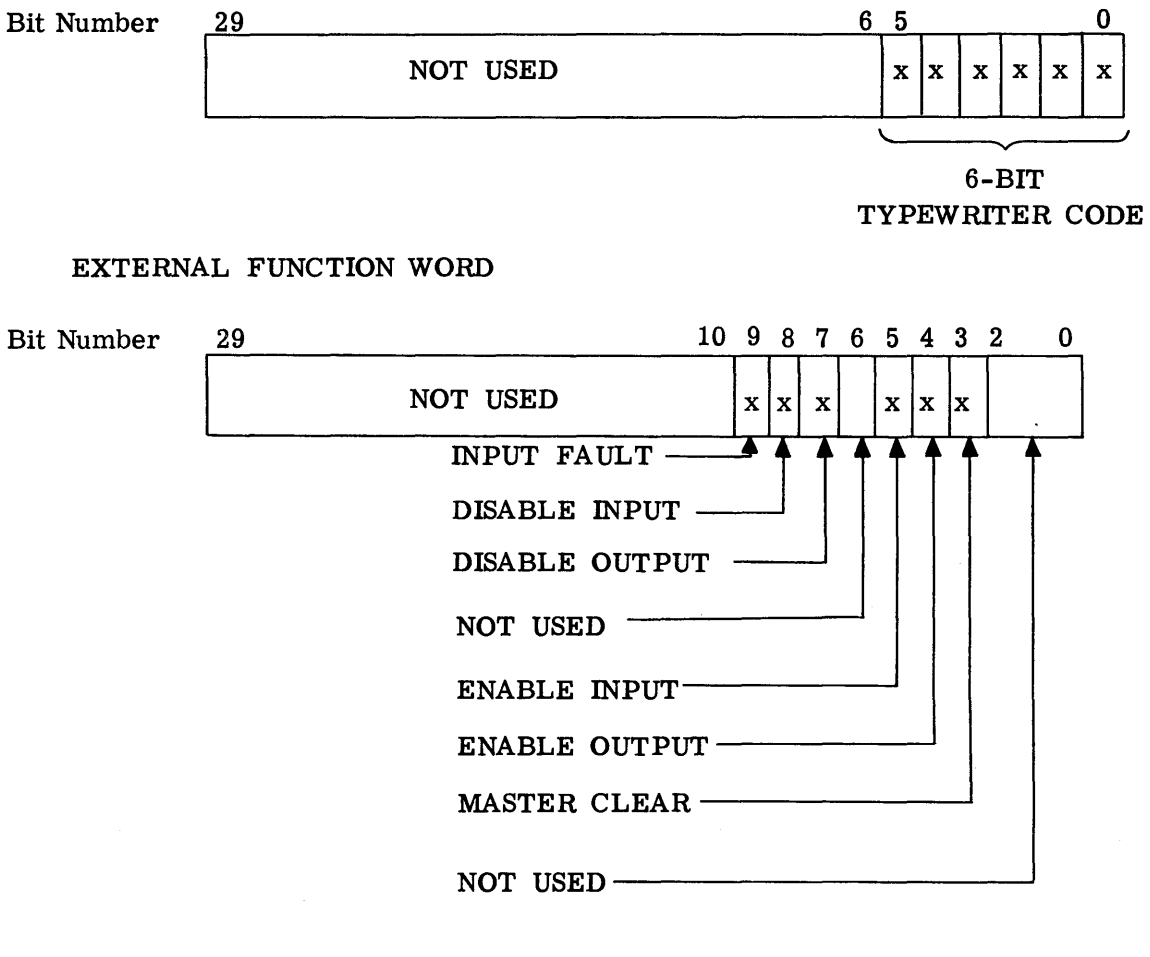


Figure D5-4. Word Formats

3. OUTPUT OPERATION

The Output mode is enabled by an *Enable Output* External Function command or by the ENABLE OUTPUT pushbutton. The Monitoring Typewriter control logic delays transmission of the first *Output Data Request* signal until necessary relays are energized and the motor has sufficient speed. If the computer has initiated an output buffer, output data transfer can begin. The computer places the first data character on six information lines of the output channel (bits 0 to 5) accompanied by an *Output Acknowledge* signal. The Monitoring Typewriter stores the character in a 6-bit data register and drops the *Output Data Request*. The typewriter begins a cycle to type the character (and perforate the paper tape if the punch is enabled). Completion of the typing cycle is sensed by timing contacts which are actuated by the typewriter translator shaft, and the *Output Data Request* line is again set. The computer may then send the next character. Data transfer continues in this manner (at a maximum rate of approximately 10 characters per second) until no more *Output Acknowledge* signals are received from the computer or until the Output mode is disabled.

The output operation is different from that just described when the Monitoring Typewriter receives a delay code or an illegal code. The delay codes, 45 (CR), 51 (TAB), and 61 (BS) require more time than the regular characters. The control logic must therefore inhibit the next *Output Data Request* signal until the typewriter is ready for another operation. When one of the illegal codes is received, it is ignored by the typewriter and data transfer continues uninterrupted. The 15 illegal codes are 00, 10, 40, 41, 43, 53, 55, 63, 65, 67, 71, 73, 75, 76, and 77.

The Output mode is disabled by *Disable Output*, *Master Clear*, or *Enable Input* External Function commands, or by the CLEAR OUTPUT, MANUAL CLEAR, or ENABLE INPUT pushbutton. Disabling the Output mode also stops the typewriter motor. The computer program may disable the Output mode immediately after the Output Buffer terminates, and the final typing cycle will still be completed.

4. INPUT OPERATION

The Input mode is enabled by an *Enable Input* External Function command or by the ENABLE INPUT pushbutton. This also starts the typewriter motor and illuminates the INPUT indicator. The Monitoring Typewriter does not permit the first keyboard entry to be made or the first tape frame to be read until the necessary relays are energized and the typewriter motor has sufficient speed.

The state of the INPUT READY indicator, a plastic indicator located in the center of the typewriter front panel and operated by the input control logic, becomes significant during the Input mode. When illuminated, a keyboard entry can be made. Keyboard entries are locked out when the INPUT READY indicator is not illuminated.

When an entry is made from the keyboard or a frame is read from tape, the character code is stored in a 6-bit data register and placed on six information lines (bits 0 to 5) of the computer input channel accompanied by the *Input Data Request* signal. If an input buffer has been initiated, the computer can sample the data and send an *Input Acknowledge* signal to enable the next keyboard entry to be made or the next tape frame to be read. The INPUT READY indicator is extinguished when a key is depressed and re-illuminated by an *Input Acknowledge*.

When an illegal code is read from the tape, it is ignored by the typewriter and an *all zero* code will be sent to the computer; however, if the illegal code is 41, 43, 53, 55, 63, 65, 71, 73, or 75, the reader will be disabled. It is then necessary to again depress the START READ switch.

If the computer program detects a data error during an input operation (format error, for example), it can notify the operator by sending an *Input Fault* External Function command which illuminates the INPUT FAULT indicator on the Monitoring Typewriter control panel. This command also stops the typewriter. Typewriter operation may be resumed, if desired, by depressing the CLEAR FAULT button.

The Input mode is terminated by *Disable Input*, *Master Clear*, or *Enable Output* External Function commands, or by the CLEAR INPUT, MANUAL CLEAR, or ENABLE OUTPUT pushbuttons. Disabling the Input mode also stops the typewriter motor. *Master Clear* is the only computer command which clears the data registers. If it is desired to disable the Input mode without losing a character which may be stored in the data register, *Disable Input* should be used. This might occur, for example, if part of an input tape is read, and then the Input mode is disabled before being re-enabled at a later time to continue reading the tape. In any other case, the computer will normally use *Master Clear* to initialize the Monitoring Typewriter control circuitry.

When switching from one mode to another, the computer program should send a *Disable* or a *Master Clear* to disable the old mode before it enables the new mode.

SECTION D6

PUNCHED TAPE SYSTEM

FOR THE

AN/USQ-17 UNIT COMPUTER

PUNCHED TAPE SYSTEM
FOR THE
AN/USQ-17 UNIT COMPUTER

1. BASIC INFORMATION

The NTDS Punched Tape System consists of a high-speed punch, a photoelectric reader, their associated circuitry, and a control panel. Operation of these units is normally in response to computer commands, but certain manual controls are available on the cabinet.

The information medium is seven-level punched paper (or Mylar) tape. A section of punched tape is shown in Figure D6-1. A single row of positions across the width of the tape is called a frame. Frames are divided into seven levels. In Figure D6-1, levels are numbered from bottom to top as follows: 0-1-2-feed hole-3-4-5-X, where the level marked X is available but not used. The other six levels are used to represent data bits. A hole punched in any of the six data-bit levels represents a *one*; the absence of a hole represents a *zero*. The tape is thus coded in binary-number notation with two 3-bit groupings in each frame representing octal numbers or Flexowriter codes. A row of feed holes, parallel to the length of the tape is found between levels 2 and 3. These holes are used, in the punch, to advance the tape and, in the reader, to generate timing pulses. A blank leader, consisting only of feed holes, should precede the first information frame. The leader should be long enough so that the tape can be threaded through the tape reader without having an information frame over the read head. Orientation and movement of tape through the punch and reader should be such that the side of the tape nearest the machine contains the three levels, 0, 1, and 2. (See Figure D6-2.)

2. PHOTOELECTRIC READER

The photoelectric reader has a single reading station composed of a row of seven photocells associated with the seven levels of a tape frame. The reading station is identified by a scribed line on the tape guide. Another scribed line parallel to the direction of tape motion indicates the position of the feed hole as it moves through the reading station.

The normal rate of tape speed through the reader is approximately 200 frames per second, but since the speed is a function of line voltage it may be as high as 230 frames per second. Thus, the time interval between sensings of successive frames of a tape may vary from 4.3 to

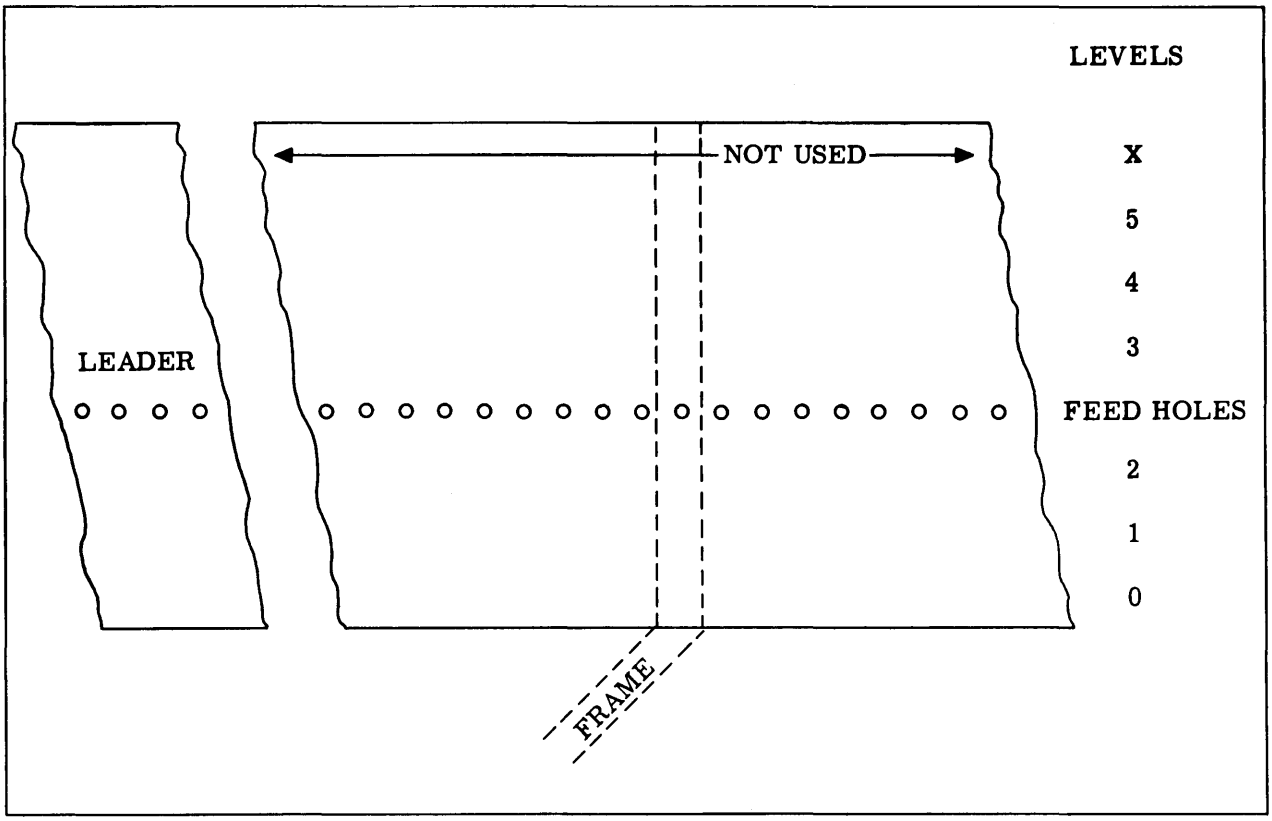


Figure D6-1. Portion of Punched Tape

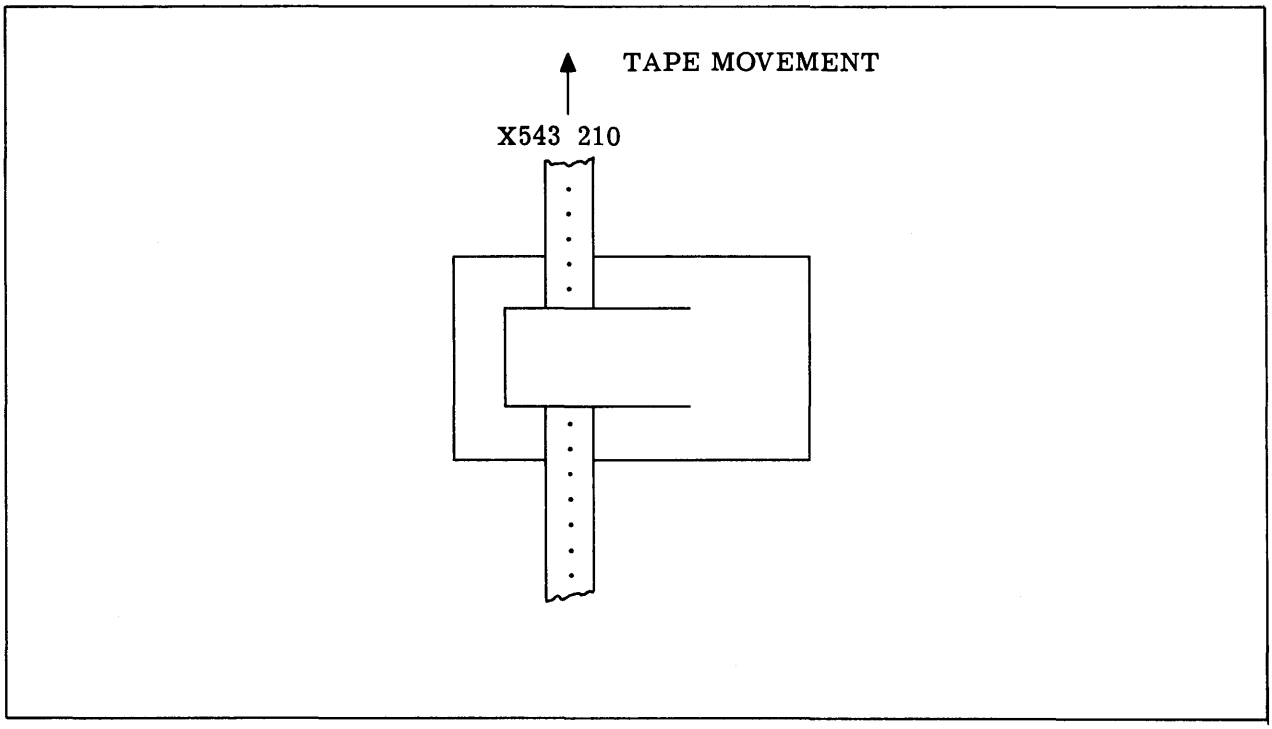


Figure D6-2. Orientation of Tape

5 milliseconds.

The reader can be selected for operation either manually or by External Function code from the computer. See Table D6-1 for External Function codes. These codes are incorporated into the program by the operator to select the mode of operation desired. The code is sent from the computer on the function cable along with a *Ready* signal which enables the function-deciphering circuitry in the paper-tape cabinet. It then performs the function requested.

TABLE D6-1. EXTERNAL FUNCTION CODES

High-Speed Punch		Photoelectric Reader	
Function Performed	Code*	Function Performed	Code*
Enable Punch Mode	XX22	Enable Reader Mode	XX31
Start Punch Motor	XX23	Start Reader Motor	XX33
Stop Punch Motor	XX24	Stop Reader Motor	XX34
Disable Punch Mode	XX26	Disable Reader Mode	XX36

* X denotes nonsignificant digit position. Notation for codes is octal; thus the lower 12 bits of the register are indicated.

For manual operation, switches located on the control panel of the paper-tape console perform the same functions as the External Function code and *Ready* signal. A switch, when momentarily held in the UP position, enables the function it controls, and when held momentarily in the DOWN position, disables the same function.

Once the reader is enabled and the motor is turned on, reader performance is the same whether selected manually or by External Function code.

For the computer to receive information from the reader, a *Ready* signal to the reader must be sent on the input cable. Upon receipt of this signal, the reader, using the same input cable, sends the information which is at that time over the row of photocells to the computer along with a *Resume* signal. At the same time, the reader starts moving the tape to the next frame. The *Resume* signal informs the computer that the reader sent a frame of data and turns the *Ready* signal off until the computer requests more data by sending another *Ready* signal. This cycle of Ready/Resume timing continues until the entire amount of information desired is read from the tape into the computer.

While the computer is operating, switches on the paper-tape cabinet control panel should not be manipulated.

3. HIGH-SPEED TAPE PUNCH

The High-Speed Tape Punch is capable of punching seven levels of information at a rate of 60 frames per second. The control circuitry senses the content of the selected C register and energizes the punch to perforate the tape frame in the levels whose corresponding data lines contain representations of *ones*. After information is received, the tape is advanced through the punch by one frame, placing the next tape frame in position to receive information.

To start the punch, two signals must be placed in the Function register. Ready/Resume signals are placed on the data cable. The punch receives its commands from Output Cable 1 of C-register 3. Effect of the signals is similar to that of the reader signals except that the data are received from the computer instead of being sent to it. External Function codes used are listed in Table D6-1.

The cycle of operation is as follows: A *Ready* signal is sent by the computer on the output cable and at the same time the computer places data on the lines. Upon receipt of data, the punch, at the next punch interval, perforates this information in a pattern of punched-out holes on the tape and sends back a *Resume*. The normal punch cycle requires about 16.7 milliseconds.

The high-speed punch is ready for operation when the tape is properly positioned in the punch with a leader preceding the first punching position. The leader may be fed through the punch by depressing either the TAPE FEED button on the punch or the TAPE FEED switch on the control panel. The punch is started in continuous cycles of operation by setting the HSP MOTOR switch on the control panel momentarily in the ON position, or by entering the External Function code XX23 in the lower 12 bits of C-register 3. To stop the punch motor, the HSP MOTOR switch is turned to the OFF position. This can be done only by transmission of *Ready* and *Resume* signals on the output cable.

4. PUNCHED TAPE CONTROL

The Function cable carries a code representing the type of action desired by the computer. Eight commands are available in this code, four of which pertain to the punch and the other four to the reader. The four commands in each group are for enabling and disabling that mode of operation and for starting and stopping the motors. The codes are entered in the lower-order 6 bits of C-register 3. Operations represented by these codes are listed in Table

5. PROGRAMMING

An outline of the various functions to be performed in enabling the reader or punch is given below:

1) Select input and output channels.

2) Provide for a delay of approximately 5 milliseconds.

3) Start motor.

4) Provide a delay of approximately 40 milliseconds for the punch or 640 milliseconds for the reader.

5) If the punch is to be enabled, the Flexowriter must first be disabled followed by a delay of approximately 40 milliseconds.

6) Disable *Ready* signal by storing contents of C-register 3 in A register.

7) Enable reader or punch.

8) Provide a delay of approximately 40 milliseconds for the punch or 170 milliseconds for the reader.

9) Disable *Ready* signal by storing contents of C-register 3 in A register.

To disable the reader or punch, the following functions should be performed:

1) Select output channel.

2) Provide a delay of approximately 5 milliseconds.

3) Disable reader or punch.

4) Provide a delay of approximately 40 milliseconds.

5) Disable *Ready* signal by storing contents of C-register 3 in A register.

The following routines (see Tables D6-2 through D6-5) show the CS-1 Compiler operations used to enable and disable the reader and punch along with machine instructions generated by these operations. Note that the SELECT-CHANNEL operation maintains a pseudo-CO at memory location 00003 because of interrupt possibilities.

TABLE D6-2. ENABLE READER OPERATIONS

SEL-CHAN • C^3OUT • C^1 IN 1	11000	00161	SELECT CHANNEL
	10000	77614	
	57030	00003	
	13030	00003	
RPT • 1000	70000	01000	DELAY
ENT • B^0 • 0	12000	00000	
ENT • C^3 • 10033	13300	10033	START MOTOR
RPT • 40000	70000	40000	DELAY
ENT • B^3 • 0	12000	00000	
STR •C^3 • A	17340	00000	
ENT • C^3 • 10031	13300	10031	ENABLE READER
RPT • 40000	70000	40000	DELAY
ENT • B^0 • 0	12000	00000	
STR •C^3 • A	17340	00000	

TABLE D6-3. ENABLE PUNCH OPERATIONS

SEL-CHAN • C^3OUT • C^1OUT	11000	00163	SELECT CHANNEL
	10000	77614	
	57030	00003	
	13030	00003	
RPT • 1000	70000	01000	DELAY
ENT • B^0 • 0	12000	00000	
ENT • C^3 • 10023	13300	10023	START MOTOR
RPT • 10000	70000	10000	DELAY
ENT • B^0 • 0	12000	00000	
STR •C^3 • A	17340	00000	
ENT • C^3 • 10016	13300	10016	DISABLE FLEX
RPT • 10000	70000	10000	DELAY
ENT • B^0 • 0	12000	00000	
STR •C^3 • A	17340	00000	
ENT • C^3 • 10022	13300	10022	ENABLE PUNCH
RPT • 10000	70000	10000	DELAY
ENT • B^0 • 0	12000	00000	
STR •C^3 • A	17340	00000	

TABLE D6-4. DISABLE PUNCH OPERATIONS

SEL-CHAN • C^3 OUT	11010	00160	SELECT CHANNEL
	10000	77617	
	57030	00003	
	13030	00003	
RPT • 1000	70000	01000	DELAY
ENT • B^0 • 0	12000	00000	
ENT • C^3 • 10026	13300	10026	DISABLE PUNCH
RPT • 10000	70000	10000	DELAY
ENT • B^0 • 0	12000	00000	
STR • C^3 • A	17340	00000	

TABLE D6-5. DISABLE READER OPERATIONS

SEL-CHAN • C^3 OUT	11000	00160	SELECT CHANNEL
	10000	77617	
	57030	00003	
	13030	00003	
RPT • 1000	70000	01000	DELAY
ENT • B^0 • 0	12000	00000	
ENT • C^3 • 10036	13300	10036	DISABLE READER
RPT • 10000	70000	10000	DELAY
ENT • B^0 • 0	12000	00000	
STR • C^3 • A	17340	00000	

SECTION D7

PUNCHED TAPED SYSTEM

FOR SERVICE TEST

CONTENTS

ILLUSTRATIONS

TABLE

PUNCHED TAPE SYSTEM FOR SERVICE TEST

1. BASIC INFORMATION

The NTDS Punched Tape Unit for Service Test is strictly for off-line operations (*not* for use while system programs are being run) such as utility and debugging functions, emergency loading of system programs (if, for example, magnetic tape system should become inoperative).

The Punched Tape Unit consists of a photoelectric reader, a tape punch, associated circuitry, and a manual control panel in a single cabinet. The unit is connected to one normal computer output cable which carries punch data and External Function Codes, and one normal computer input cable which carries reader data. In a multicomputer installation, the Punched Tape Unit cables are routed to an Interconnection Panel where they may be manually switched to any computer. In the computer output cable, 10 of the 30 data signal lines are used (20 spares). In the computer input cable, only seven data signal lines are used (see Figure D7-1).

The information medium is seven-level punched paper or Mylar® tape. A transverse row of holes across the tape is called a frame. Each frame has seven information levels in which a hole represents a *one* and the absence of a hole represents *zero*. Tape levels are numbered 0-1-2-3-4-5-6- as shown in Figure D7-2. Levels 0 through 5 are data bits which may be straight binary or a coded representation. Level 6 is a utility bit which may be used for parity or control information as desired. If it is employed as a frame parity bit, its encoding and decoding will be confined to the computer program. The photoelectric reader logic will not be capable of detecting parity failure, and tape punch logic will not be able to determine and punch parity unless this information is supplied by the computer.

Data transfer between computer and Punched Tape Unit takes place in the buffer mode with one frame (7 bits) being transferred during each buffer operation. Both punch and reader may be operated simultaneously.

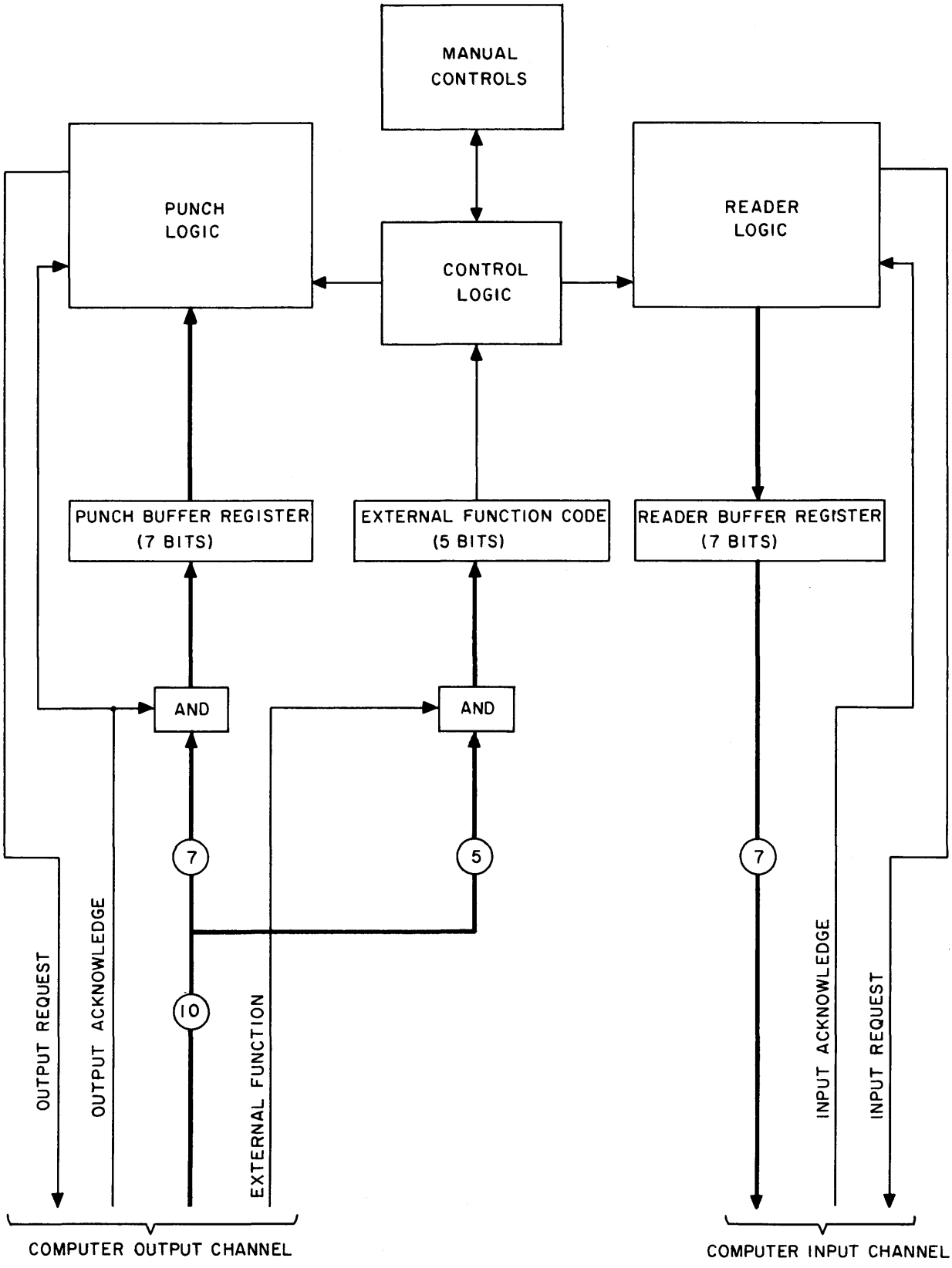


Figure D7-1. Block Diagram - Punched Tape Unit

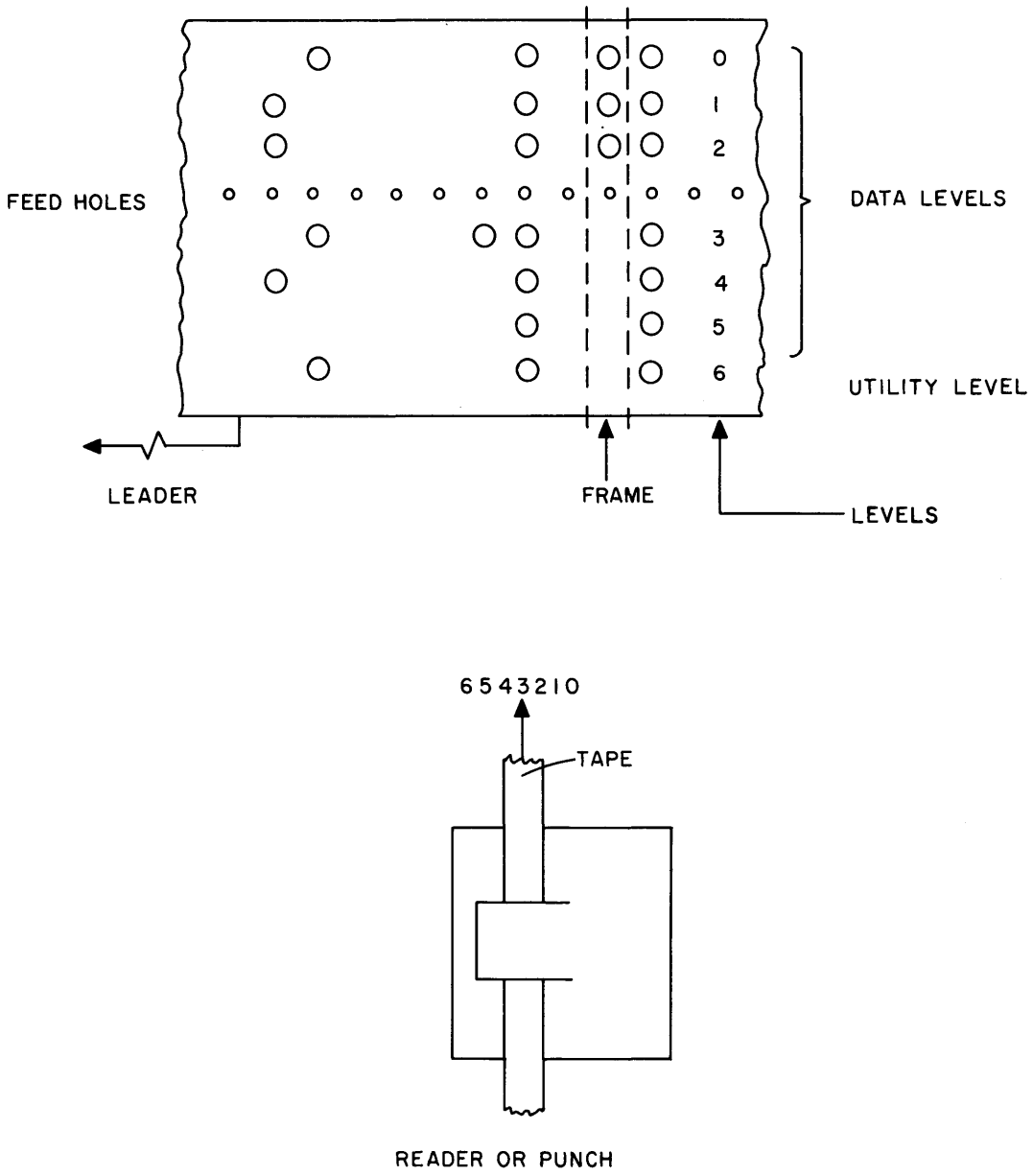


Figure D7-2. Seven-Level Punched Tape

2. PHOTOELECTRIC READER

The photoelectric reader has a reading station composed of eight photosensitive cells. Seven of these cells read the seven levels of a tape frame while the eighth senses the tape feed hole for timing purposes. Tape is transported through the reader by an electric motor with a clutch and brake device. A single External Function Code from the computer starts the reader motor and enables reader logic. Another External Function Code stops the motor and

disables the reader. Alternatively, these functions may be selected by depressing push buttons on the manual control panel. Transmission of the Input Data Request signal is inhibited until the motor has attained sufficient speed.

The cycle of operation consists of reading one frame from the tape as follows (assuming the photoelectric reader has been previously enabled): the reader engages the clutch, advances tape, and reads one frame of data into a seven-bit buffer register. These data are then placed on the computer input channel followed by the Input Data Request signal. These signals are maintained until the computer responds with an Input Acknowledge signal indicating that it has accepted the data. Meanwhile, the reader engages brake and begins to stop tape. Receipt of Input Acknowledge permits the reader to drop data and Input Data Request lines, advance tape, read next frame, etc. Since these steps occur very rapidly, tape moves continuously as long as the computer responds with Input Acknowledge signals.

3. TAPE PUNCH

The tape punch is capable of punching seven information levels plus a tape feed hole. A synchronous motor advances tape and supplies timing control signals for the punch cycle. A single External Function Code from the computer starts the punch motor and enables punch logic; another External Function Code stops the motor and disables punch. Alternatively, these enable and disable functions can be selected by push buttons on the manual control panel. Transmission of the Output Data Request signal is inhibited until the motor has attained sufficient speed.

The cycle of operation consists of punching one frame on the tape as follows (assuming punch has been previously enabled): When ready to accept data, the punch sends an Output Data Request signal which is maintained until the computer sends data and an Output Acknowledge signal. Then data are stored in a seven-bit buffer register and the Output Data Request signal is dropped. When a synchronizing signal is received from the punch motor, contents of the register are punched on tape. On completion of the punch cycle, a new Output Data Request signal is sent to the computer.

4. PUNCHED TAPE UNIT CONTROL

As indicated previously, function control of the Punched Tape Unit may be accomplished either manually or by the computer.

1) *Manual Control*

The Punched Tape Unit control panel contains the following controls and indicators:

a) *Enable Reader* push button (with indicator)

b) *Disable Reader* push button

c) *Enable Punch* push button (with indicator)

d) *Disable Punch* push button

e) *Manual Clear* push button

f) *Tape Feed* push button

g) *Input Fault* indicator (with *Clear* button)

h) *Operating Mode* toggle switch — computer/copy with start and stop buttons for copy mode

Functions of the Enable and Disable buttons have been described in Subsections 2 and 3. The Manual Clear button clears all registers and control bits in the Punched Tape Unit. The Tape Feed button permits punching of tape leader with feed holes. The Fault Clear button extinguishes the Input Fault indicator and re-enables the Input Data Request (see following paragraph, *Computer Control*). The Operating Mode switch has two positions:

a) Computer Mode - Unit on-line to computer

b) Copy Mode - Unit off-line from computer. Punched Tape Unit may be used off-line for copying tapes. To do this, tape is placed in the reader and then reader and punch are manually enabled and Start button depressed. The seven tape levels are read by the reader and reproduced directly by the punch.

2) *Computer Control*

The computer can send six different External Function Codes to the Punched Tape Unit (see Table D7-1). They are bit-position encoded; i.e., a bit set to *one* means the associated function is applicable.

Table D7-1. EXTERNAL FUNCTION CODES

Bit Number	Meaning
3	Master Clear
4	Enable punch and start motor
5	Enable reader and start motor
7	Disable punch and stop motor
8	Disable reader and stop motor
9	Input Error

If computer program sends a Disable Punch code as soon as output buffer to the punch terminates, last data frame is still punched before punch is disabled.

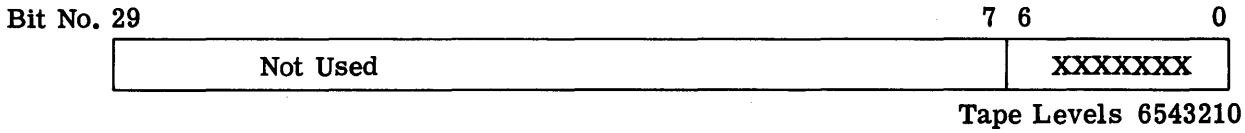
The Input Error function code permits the computer to inform the Punched Tape Unit operator (by means of an indicator light) of parity, check sum, or format errors in input data from the reader. This code also inhibits the Input Data Request signal suspending input data transfer until the Proceed button is pushed.

Receipt of an illegal function code combination simultaneously enabling *and* disabling either reader or punch will result in an enable function.

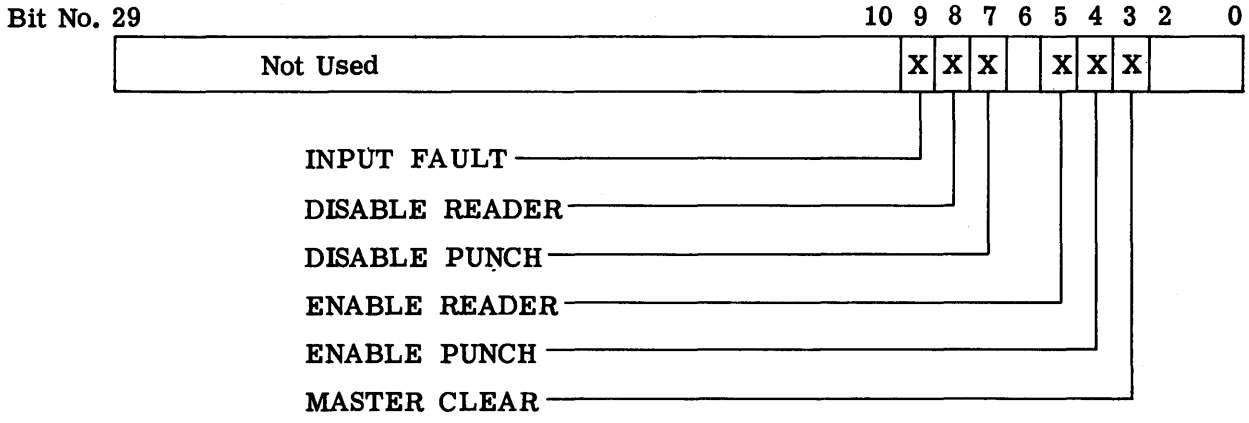
5. WORD FORMATS

Formats shown below will be used for input and output buffer data words and for External Function Codes to the Punched Tape Unit.

Reader and Punch Data Word



External Function Word



SECTION D8

MAGNETIC DRUM SYSTEM

FOR THE

AN/USQ-17 UNIT COMPUTER

MAGNETIC DRUM SYSTEM
FOR THE
AN/USQ-17 UNIT COMPUTER

1. BASIC INFORMATION

The Magnetic Drum is a medium-speed, medium-capacity storage device accessible to the NTDS Unit Computer (AN/USQ-17) through a normal communication channel. Its dependability, relatively short total access time (as compared with magnetic tape), and non-volatile storage characteristics make the Magnetic Drum well suited as an auxiliary storage device for use in NTDS.

2. DESCRIPTION

The main registers and their functions include

1) STORAGE ADDRESS REGISTER (SAR) - a 15-bit register that holds a 12-bit address and a 3-bit group location designator.

2) FUNCTION REGISTER - a 6-bit register that holds the function to be executed which may be one of five: a) *read single address*, b) *write single address*, c) *read block*, d) *write block*, and e) *clear*.

3) E REGISTER - this 30-bit input/output register holds information word to be stored on or read from the drum.

4) ANGULAR INDEX COUNTER (AIK) - this register indicates the angular position of the drum. There are 4096 discrete positions.

The NTDS Magnetic Drum has the following physical and functional characteristics:

- The data word transferred across the computer/drum interface consists of 30 bits.
- Drum storage capacity is 32,768 words (983,040 bits).
- Access time varies from 12.8 microseconds to 32.7 milliseconds. An average time of 17 milliseconds is required to locate any word on the drum. Successive words in a block are buffered one every 128 microseconds.

- Provisions have been made to address a single word or a block of words, initial and terminating addresses of which are controlled by the computer. One to 32,768 words can be transferred as a block.
- Drum rotates at 1800 rpm.

3. MAGNETIC DRUM CONTROL

The Magnetic Drum is capable of interpreting five External Function commands issued by the Unit Computer:

1) 41 Single address read
2) 42 Single address write
3) 44 Block transfer read
4) 45 Block transfer write
5) 47 Clear drum designator

The operational sequence of the drum for each of the drum commands is as follows:

1) *General* - The Function register accepts an External Function code from C-register 3 of the computer along with the *Ready* signal. The External Function code, being translated, sets a bit register, initiates the proper *enables*, and prepares control for the proper operation of the desired function.

2) *Single Address Read* - During this operation, SAR is set to the desired address from the drum address output channel on the computer. When this address is located by the AIK, information stored at this address is read into the E register of the drum unit. When the drum recognizes an Input *Ready* from the computer, the contents of Register E are fed to the drum input channel of the computer.

3) *Single Address Write* - The write External Function is entered into the drum unit from an External Function channel of the computer. The address is then sent to the SAR, and information is entered in the E register from the drum output channel of the computer. When the proper address is located by the AIK, the contents of Register E are stored on the drum.

4) *Block Transfer Read* - The operational sequence is the same as for a *Single Address Read*, except that the SAR becomes a counter and is advanced by one each time an Input *Ready* signal is received from the computer.

5) *Block Transfer Write* - The operational sequence is the same as for a *Single Address Write*, except that the SAR becomes a counter and is advanced by one each time an Output *Ready* signal is received from the computer.

6) *Clear Drum Designator* - The effect of this External Function command is to clear the previous External Function request. This *must* be initiated whenever a change of External Function is desired.

4. PROGRAMMING SPECIFICATIONS

The order of drum commands and timing necessary are outlined below. These ordered drum commands plus computer input/output rules will enable the programmer to reference the drum for single-address reading or writing.

1) Select necessary input/output function channels.

2) Clear previous drum External Function code.

3) Provide a 20-microsecond delay.*

4) Send new drum External Function code.

5) Provide a 20-microsecond delay.*

6) Send drum address for *Read* or *Write* command.

5. DRUM-HANDLER ROUTINE

When it is desired to use the drum in the buffer mode, it is recommended that the buffering be done by the Drum-Handler Routine. This routine is a part of the CS-1 Compiler, see CS-1 PROGRAMMER'S REFERENCE MANUAL, PX 1349, page II-C-21. It is important to note that an exit is not made from this Drum-Handler Routine until after data transfer is completed. Table D8-1 shows the parameters for data transfer to and from the Magnetic Drum; Table D8-2 shows the drum-handler routine.

A. *WRITE ON DRUM*

Enter lower half of Q register with the base address of the block of core memory which is to be transferred to the drum.

Enter lower half of A register with the number of words in the block which are to be transferred.

---

*The CS-1 compiler operator EX-FCT provides a sufficient delay after sending a given command.

Enter B-register 7 with the drum address which designates the beginning of the area of drum into which the block is to be transferred.

Return Jump to the first address of the Drum-Handler Routine + 2 (DRUMRO+2) or DRUMWO.

B. *READ FROM DRUM*

Enter lower half of Q register with the base address of an area of core memory into which a block of data is to be transferred from the drum.

Enter lower half of A register with the number of words in the block which are to be transferred.

Enter B-register 7 with the base drum address of the block of data which is to be transferred.

Return Jump to the first address of the Drum-Handler Routine, DRUMRO, which then reads the block of data from the drum and stores it in core memory.

TABLE D8-1. PARAMETERS FOR DATA TRANSFER TO AND FROM MAGNETIC DRUM

	A_L	Q_L	B^7	
DRUM WRITE (Core→Drum)	Number of words in block to be written onto drum	Base Core Address of block to be written onto drum	Base Address of drum area where block is to be stored	RJP•DRUMWO
DRUM READ (Drum→Core)	Number of words in block to be read into core	Base Core Address where block is to be read	Base Address of drum area from which block is to be read	RJP•DRUMRO

TABLE D8-2. DRUM-HANDLER ROUTINE

LABEL	STATEMENT	NOTES
DRUMRO	JP•0	EXIT
	JP•DRUMR7	
DRUMWO	JP•0	WRITE EXIT
	RJP•DRUMW5	
DRUMR7	STR•Q•W(DRUMR5)ANOT	SET CORE ADDRESS
	JP•DRUMRO	ZERO BUFFER
	STR•A+Q•U(DRUMR5)	SET UPPER LIMIT
	SEL-CHAN•DRADDOUT•FCTNOUT•DRUMIN	
	ENT•DRUM•A	CLEAR D
	EX-FCT•DRUMC8•47	CLEAR DRUM DESIGNATOR
	EX-FCT•DRUMC8•42	BUFFER READ
	ENT•DRADD • B7	SET DRUM ADDRESS
DRUMR3	JP•DRUMR3•DRADDFULL	
	IN•DRUM•W(DRUMR5)	BUFFER
DRUMR4	ENT•A•L(BUFFERADD)	COMPLETE BUFFER
	SUB•A•U(BUFFERADD)•AZERO	
	JP•DRUMR4	
DRUMR6	JP•DRUMR6•DRUMEMPTY	LAST WORD
	EX-FCT•DRUMC8•47	CLEAR DRUM DESIGNATOR
	JP•DRUMRO	EXIT
DRUMR5	0•0	BUFFER STORE
DRUMW5	JP•0	
	STR•Q•W(DRUMR5)ANOT	SET CORE ADDRESS
	JP•DRUMWO	ZERO BUFFER
	STR•A+Q•U(DRUMR5)	SET UPPER LIMIT
	SEL-CHAN•DRADDOUT•FCTNOUT•DRUMOUT	
	STR•DRUM•Q	CLEAR D
	EX-FCT•DRUMC8•47	CLEAR DRUM DESIGNATOR
	EX-FCT•DRUMC8•45	DRUM BUFFER WRITE
	ENT•DRADD•B7	SET DRUM ADDRESS
DRUMW3	JP•DRUMW3•DRADDFUL	
	OUT•DRUM•W(DRUMR5)	BUFFER WRITE
DRUMW4	ENT•A•L(BUFFERADD)	COMPLETE BUFFER
	SUB•A•U(BUFFERADD)AZERO	
	JP•DRUMW4	
DRUMW6	JP•DRUMW6•DRUMFULL	LAST WORD
	EX-FCT•DRUMC8•47	CLEAR DRUM DESIGNATOR
	JP•DRUMWO	EXIT

A declaration tape is needed to define input, output, and function channels in order to compile the DRUM-HANDLER Routine.

If, for example, the drum were assigned to use C-4 input channel 1, the C-4 output channel and output functions channel 5, the declaration tape would appear as in Table D8-3, below.

TABLE D8-3. SAMPLE DECLARATION TAPE

DRUMHAND	PROGRAM•VER STEEG • 11OCT1960
DRADD	MEANS• C2
DRUM	MEANS• C4
DRADDFULL	MEANS• C2FULL
DRUMEMPTY	MEANS• C4EMPTY
DRUMFULL	MEANS• C4FULL
DRADDOUT	MEANS• C2OUT
FCTNOUT	MEANS• C3OUT
DRUMIN	MEANS• C4IN
DRUMC8	MEANS• 5
DRUMOUT	MEANS• C4OUT
BUFFERADD	EQUALS•4

SECTION D9
FUNCTIONAL SPECIFICATIONS
FOR THE
SERVICE TEST MAGNETIC TAPE SYSTEM

CONTENTS

CONTENTS (Cont.)

ILLUSTRATIONS

FUNCTIONAL SPECIFICATIONS

FOR THE

SERVICE TEST MAGNETIC TAPE SYSTEM

1. BASIC INFORMATION

The Magnetic Tape System described in this section meets the requirements, as specified by the Bureau of Ships, for a simple Service Test tape system for NTDS. It will provide, as a subsystem, a large capacity, medium-speed, auxiliary storage potential; its application in the system must be thoroughly delineated.

Considering total cost and the quite limited development time available, the needs of NTDS for a Service Test Magnetic Tape System will best be served by building this system for *off-line* applications only. This choice imposes the following restrictions on use of this tape system in Service Test:

1) Essential data will not be recorded on magnetic tape in *real-time* for later *real-time* recovery

2) Data may be extracted in *real-time* only if occasional data loss is permitted

3) In essential applications such as program recovery, it is desirable to have a second transport on a standby basis to perform the same function

4) Program recovery is to be considered a *nonreal-time* function. This permits sufficient time to position the tape, reread, etc

This tape system will employ two separate formats, designated Remington Rand Univac Format and NTDS Format, in writing and reading. With Univac Format, which provides com-

patibility with the High-Speed Printer and the Remington Rand Univac File Computer Tape System, 128 lines per inch can be written with 120 lines or tape frames to each block of information. One block contains 24 computer words. Univac Format, consistent with the 112-1/2 ips tape speed, provides an average (30-bit) word-transfer rate of approximately 1.2 kc.

With NTDS Format, 200 lines per inch can be written in variable block length. Length of blocks vary between 24 computer words and 16,000 words. By redundant recording (which in this case means twice recording the same data on tape), NTDS Format will provide tape input/output with a high degree of reliability. Word-transfer rate in NTDS Format is approximately 2.25 kc, not considering interblock space.

A block diagram of the tape system is illustrated in Figure D9-1. The system consists basically of three sections: 1) Input/Output Duplexer, 2) Magnetic Tape Control (MTC), and 3) Tape Transport (TT) units. A general description of functions performed by each of these three sections is given here; detailed specifications of each section are contained in the three following subsections.

I/O Duplexer enables the tape system to communicate with two AN/USQ-20 Unit Computers. Only one computer may communicate with the tape system at a given time. Before either computer may reference the tape system with a particular command, it must first gain *control* of the system. It is a function of the Duplexer to grant or refuse control of the tape system to computers that request it. Control will be granted a computer upon request when the system is not being controlled by either computer. When, for example, Computer A requests control of the tape system while it is being controlled by Computer B, the request will be "remembered"; control will be granted to Computer A when Computer B releases it.

The Magnetic Tape Control determines what operation, other than *control,* is commanded by a computer and initiates necessary actions to perform that operation. MTC communicates with a computer via the Duplexer and also with Tape Transports. The computer indicates what operation it desires of the tape system by an appropriate code in the 30-bit Function word. MTC is responsible for assembly and disassembly of computer words and transmits data to or receives them from both the computer and Tape Transport units. Upon completion of any tape-system function except *Master Clear,* MTC transmits an *Interrupt* to the computer; a *Status* word, which describes successful or unsuccessful completion of the function, is placed on the 30 data lines of the input cable to the computer. This Interrupt "informs" the computer that the last requested function was completed.

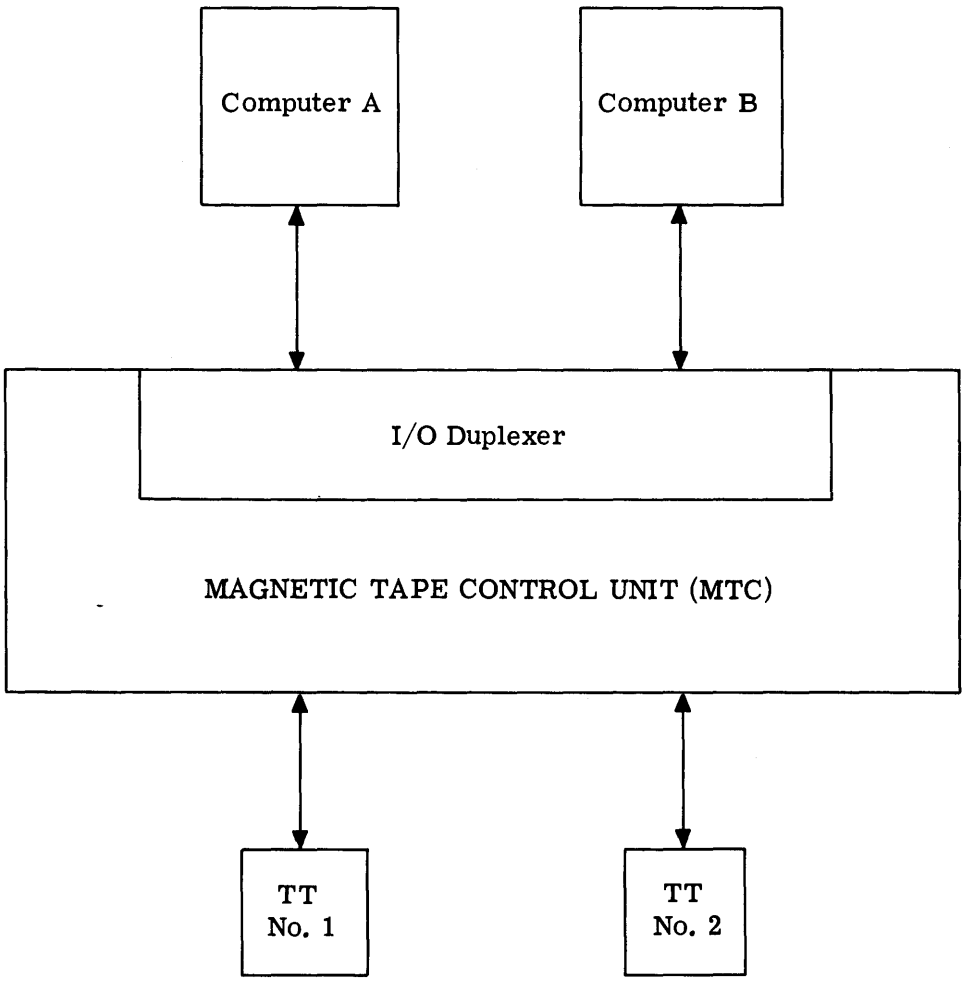


Figure D9-1. Block Diagram of Magnetic Tape System

Tape Transports handle 1/2-inch tapes and contain logical circuitry necessary to move the tapes forward or backward, recognize beginning and end of tapes, and inform MTC of operational capabilities. There will be two Tape Transports associated with each Service Test Magnetic Tape System. All such Tape Transports will be completely compatible with either Univac or NTDS Format. A tape written on any TT may be read on any other TT.

2. TAPE TRANSPORT GENERAL REQUIREMENTS

For a detailed specification of the Service Test Tape Transport, refer to the latest revision of Remington Rand Univac document DS-4504, *Transport Magnetic Tape*. The Tape Transport that will be used with this tape system has the following general characteristics:

1) *Tape* - TT handles 1/2-inch width tape on B-wound reels (oxide coating on outside of tape).

2) *Tape Speed* - Writing and reading are performed at a tape speed of 112-1/2 inches per second; Rewind is performed at a tape speed of 225 inches per second.

3) *Reels* - TT accepts RRU-type reels.

4) *Compatibility* - Recorded tapes are compatible from TT to TT within all Service Test Tape Systems and are compatible with the High-Speed Printer and Univac File Computer Tape Systems.

5) *Manual Controls* - The controls for the TT are located in the MTC. These controls are discussed in subsection 3.J of this section.

3. MAGNETIC TAPE CONTROL SPECIFICATIONS

The Magnetic Tape Control (MTC) communicates directly with the Tape Transports and with one of two computers through the Duplexer. Information flow between Unit Computer and MTC consists of the following:

From Unit Computer to MTC

a) External Function enable

b) Output Acknowledge

c) Input Acknowledge

d) 30-bit words that contain either data to be written on tape or an External Function code

From MTC to Unit Computer

a) Output Request

b) Input Request

c) Interrupt

d) 30-bit words that contain either data read from the tape or a Status word that describes condition of the tape system or Improper Conditions detected in most recent operation

Information flow between MTC and Tape Transport units (TT) consists of the following:

From MTC to TT

a) Commands such as *Move Forward, Move Reverse,* and *Rewind*

b) Eight bits (to be written on tape) which contain sections of a computer word, parity information, and a sprocket bit

From TT to MTC

a) Eight bits (read from the tape) which contain a section of a computer word, parity information, and a sprocket bit

b) An *Available/Not Available* line which describes operational availability of a particular TT

c) A line that signifies end of tape

d) A line that signifies beginning of tape

e) A line that signifies that a Master Tape is mounted on the TT

Duties of MTC vary from operation to operation. Over-all MTC responsibilities for performance of all operations indicated in the Function word are now defined.

A. *PERFORMANCE OF FUNCTIONS*

The Function word indicates to the tape system which operation is called for by the computer. Certain functions performed by the Duplexer are discussed in Section D11 of this document and NTDS Technical Note 242, *Functional Specifications for Peripheral Equipment Duplex Operation.* If the operation called for is not to be performed by the Duplexer, the Function word is transmitted to MTC where it is translated and the function called for is executed. If the operation is to be performed by MTC, the following communication sequence occurs (it is assumed that the computer which issues the External Function command is in control of the tape system):

1) Computer executes an External Function instruction which places 30 bits on the 30-bit output cable and sets External Function line to the tape system.

2) Duplexer recognizes the signal on External Function line and determines that Function word does not demand a Duplexer function. If the tape system is not *active,* the Function word is allowed to pass into the MTC *function-decoding* register. If the tape system is actively engaged in a function, the present Function word (except a *Master Clear* function) will be ignored. It will be the responsibility of the computer not to issue an External Function command while the tape system is active. How the computer handles this is a program responsibility and is discussed under "Programming Considerations," Subsection 5.

3) MTC receives the Function word and initiates the indicated function. What occurs during the performance of the function is discussed for the individual functions under "Function Word," Subsection 3.B.

4) The computer automatically removes the External Function-line signal and the 30-bit word from the output cable. Timing for signals on these lines is discussed in *Revised Input/Output Specifications for the AN/USQ-20 Unit Computer and Associated Equipment,* NTDS Technical Note No. 233. The tape system will be able to receive the Function word within 23.6 microseconds and receive the External Function-line signal within 8.8 microseconds.

5) When the function, other than a *Master Clear,* has been completed, the tape system places a Status word on the 30 data lines of the input cable to the computer and sets the Interrupt line of the input cable. Completion of the *Master Clear* does not interrupt the computer.

6) At its convenience, the computer recognizes the Interrupt signal and samples the data lines. The computer then sets the Input Acknowledge line to indicate that it has sampled the data.

7) Tape system senses the Input Acknowledge line.

8) Tape system drops the signals on the Interrupt line and the 30 data lines and becomes inactive.

Events outlined in the previous paragraphs are illustrated graphically in Figure D9-2. Only the sequence and not exact timing relationships are indicated.

The above procedure is carried out for each function performed by the tape system. The difference between t_o and t_n (shown in Figure D9-2) vary from one function to another and may also

vary as a result of computer's *freedom* in acknowledging the Interrupt. The latter variation should be slight, however, in comparison to the former.

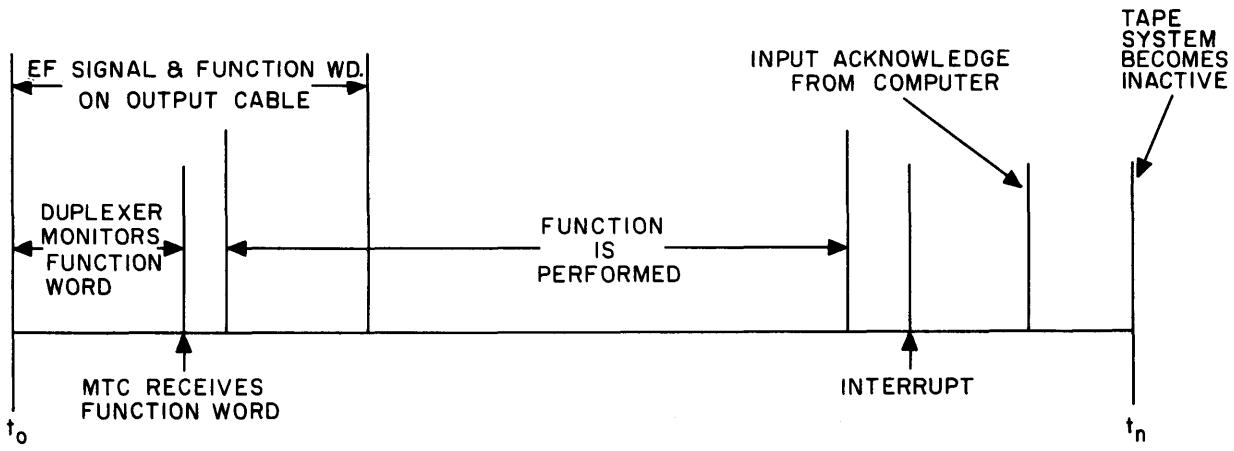


Figure D9-2. Timing of Function Word Performance

B. *FUNCTION WORD*

Individual functions performed by the tape system as they are defined by the Function word are discussed herein. It is assumed in all cases that the computer that transmits the Function word has *control* of the tape system. Figure D9-3 shows the structure of the Function word and the operations that may be requested.

Four basic operations called for by the Function word received by MTC are Read, Write, Rewind, and Master Clear. The four operations have variations involving the selection of direction, format, and Tape Transport. *Rewind with Lockout* may be considered a variation of *Rewind.* Each of the functions performed by the MTC is discussed below in the order of ascending bit position.

(1) *Duplexer Functions* (Bits 0 - 2)

These functions are discussed in Section D11 of this document and Technical Note 242, *Functional Specifications for Peripheral Equipment Duplex Operation.*

(2) *Master Clear* (Bit 3)

Execution of this function stops motion of all tapes, and "clears" MTC so that it is ready to receive a new Function word. *Master Clear* could be used at the beginning of a program before another function is requested or with discretion when doubt exists as to the condition of MTC. In the latter case, *Master Clear* assumes the role of a re-sync tool. When sent by the com-

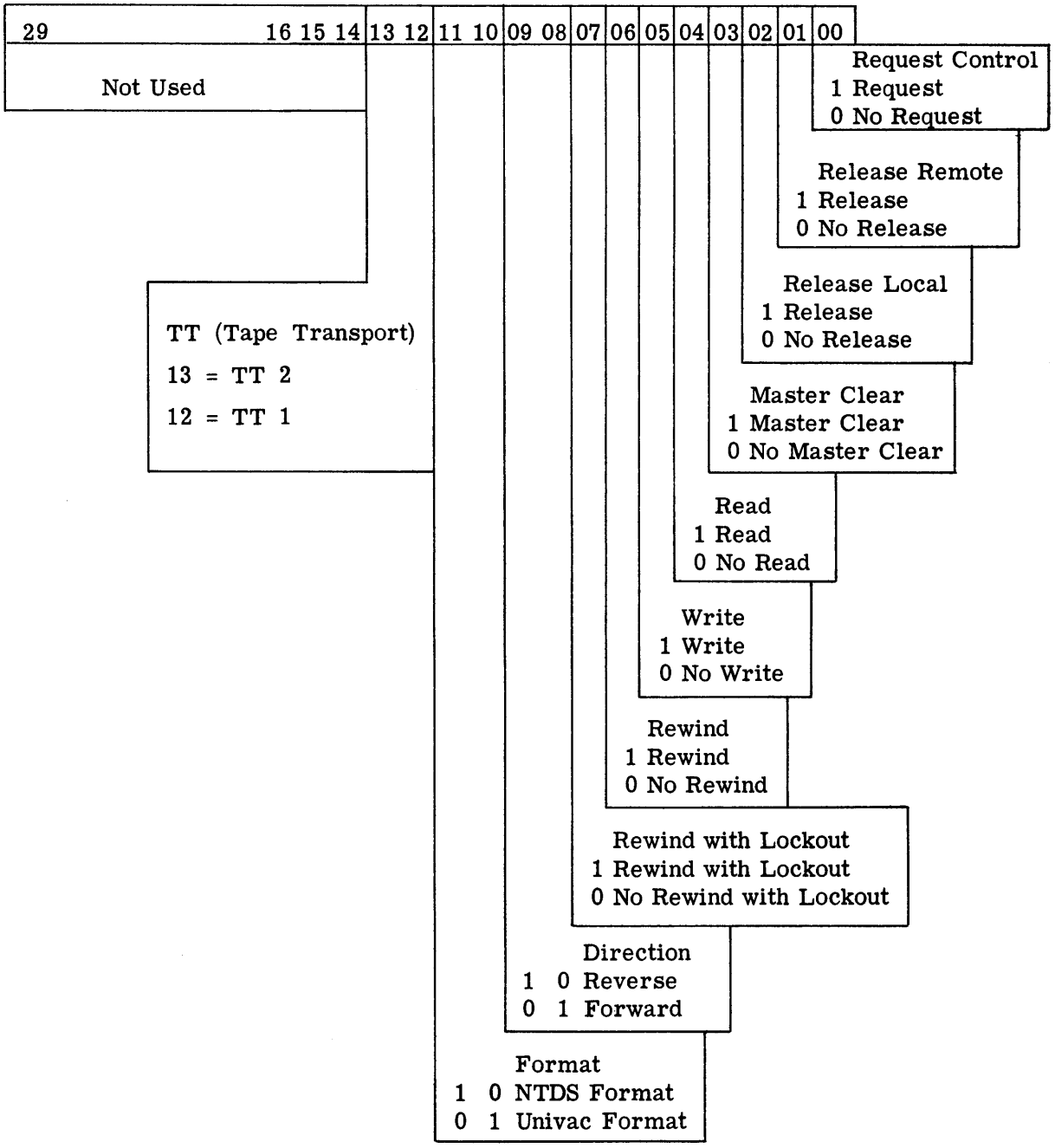


Figure D9-3. Function Word Format

puter in *control,* the *Master Clear* will be executed even if the tape system is *active.* Since the completion of the *Master Clear* does not interrupt the computer, it may be followed immediately by another External Function.

Sequence of events in execution of the *Master Clear* function is illustrated in Figure D9-4.

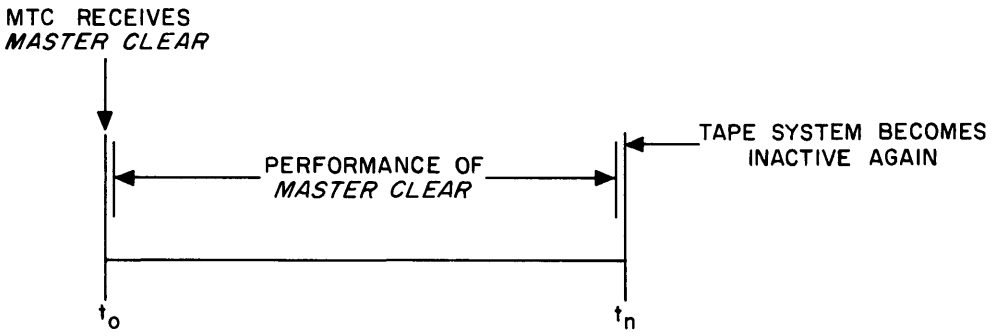


Figure D9-4. Execution of *Master Clear*

(3) *Read* (Bit 4)

This function is supplemented by selection of tape direction, format selection, and TT selection. The selected TT moves tape at 112-1/2 ips in the selected direction and transfers 8-bit frames (read from the tape) to MTC. Depending on the format (see Subsection 3.D), either three or six data bits are taken from the eight bits received from TT and assembled into 30-bit computer words. In Univac Format, the 8-bit frame contains six data bits, one parity bit (odd), and one sprocket bit. In NTDS Format, the eight bits include three pairs of data bits redundantly (two sets of three bits supposedly identical) and one pair of sprocket bits redundantly (two sprocket bits).

The assembled computer word is placed on the 30 data lines of the input cable, and an *Input Request* signal is sent to the computer. The computer samples the 30 data lines at its convenience and returns an *Input Acknowledge* to the tape system. Because the tape continues to move and a new word is being assembled (until block end is reached), the computer should sample data lines within a specific time period to prevent loss of one or more words in the block. This time period is dependent on tape speed, line density on tape, and the number of data bits per line. For Read Univac Format (see Subsection 3.D), time to assemble a computer word is 347 microseconds. For Read NTDS Format, word assembly time is 444 microseconds.

If the computer, because of higher priority inputs or outputs, fails to sample the tape-system

data lines within (347-x) microseconds and (444-y) microseconds, respectively, for Univac and NTDS Formats, an input timing error will occur. The values of x and y are selected to provide a margin of safety for the tape system. The computer is informed of an input timing error by a code in the Interrupt Status word at the end of the *Read* function.

MTC determines End of Block by noting lack of data transfer from TT within an alloted period of time. The time is a function of format selection. When MTC recognizes a block end, it initiates end of *Read* function. When the computer samples the last word on the data lines, a Status Interrupt is sent to the computer. When the Interrupt is acknowledged, the *Read* function is terminated. Figure D9-5 indicates the sequence of events in execution of the *Read* function.

(4) *Write* (Bit 5)

This function is supplemented by TT selection and format selection. Writing is performed only when the tape moves in the forward direction. Selected TT moves forward at 112-1/2 ips; MTC takes 30-bit computer words from the data lines of the output cable, disassembles them according to format selection (either three bits at a time in NTDS Format or six bits at a time in Univac Format), and transfers 8-bit frames to TT to be written on tape. In NTDS Format, the eight bits include three bits of data written redundantly and a redundant sprocket bit. Write density of NTDS Format is 200 lines per inch. In Univac Format, the eight bits include six data bits, one odd parity bit, and one sprocket bit. Write density of Univac Format is 128 lines per inch.

An *Output Request* signal is sent to the computer when MTC recognizes the *Write* function. The computer responds, at its convenience, with an *Output Acknowledge* and a 30-bit word on the data lines of the output cable. MTC recognizes the *Output Acknowledge*, samples the 30

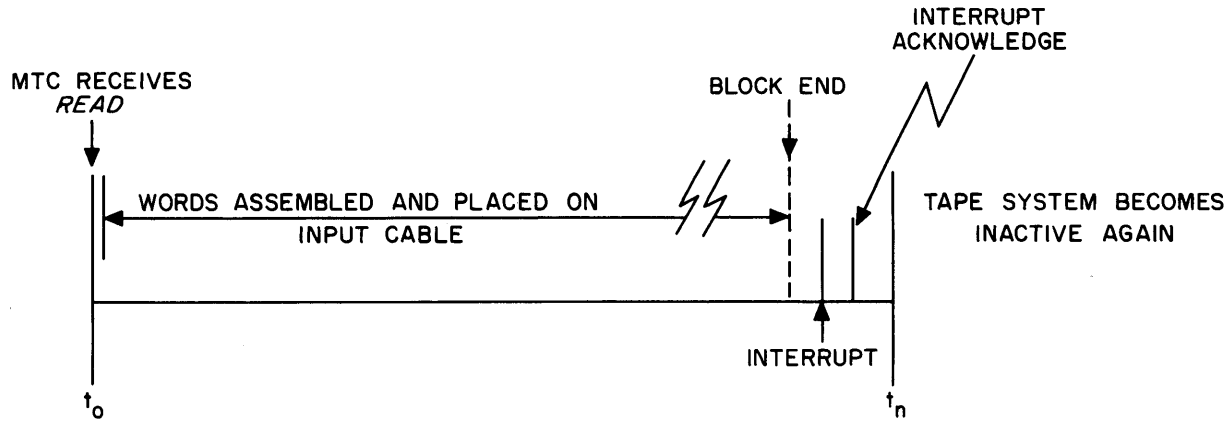


Figure D9-5. Execution of *Read* Function

data lines, and removes the *Output Request*. MTC disassembles the word just received and starts the writing process; however, when the word transfers to the Disassembly Register, MTC sends another *Output Request* to the computer.

This process continues until the computer no longer acknowledges the *Output Request* within a time period sufficient to allow the word to be written in its proper place on tape. This time period, as quoted before, is (347-x) microseconds in Univac Format and (444-y) microseconds in NTDS Format. There are two situations where the computer will not acknowledge the *Output Request* "in time". One is where the Output Buffer is finished; the other is where the computer acknowledged higher priority *input* and *Output Requests* and time "ran out". Either situation is recognized by MTC as *End of Block* and precedes termination of the *Write* function; however, if the computer acknowledges an *Output Request* after the prescribed time expires, MTC senses an output timing error. Notification of such an output timing error will be sent to the computer by means of the Status word sent at the end of the *Write* function via Interrupt. After MTC determines *End of Block* and starts termination procedure, the Interrupt must be withheld for at least 140 microseconds to permit determination of an output timing error.

When MTC detects *End of Block*, it stops tape motion allowing for an interblock space of approximately 1.2 inches. MTC removes *Output Request* and sets Interrupt line, and a 30-bit Status word is placed on the input cable. The computer samples the input cable at its convenience and acknowledges the Interrupt. The Interrupt is dropped by MTC, and the tape system becomes inactive. Execution of the *Write* function is demonstrated graphically in Figure D9-6.

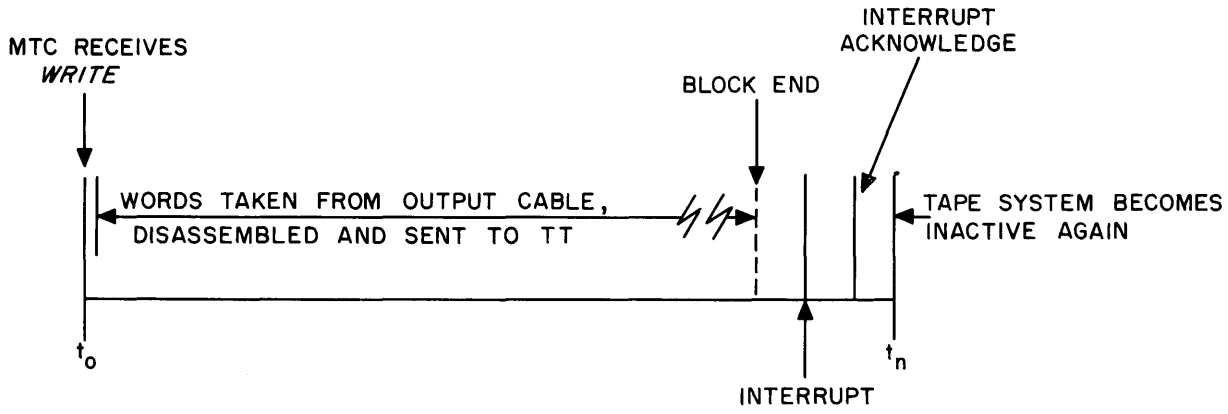


Figure D9-6. Execution of *Write* Function

(5) *Rewind* (Bit 6)

This function is supplemented by TT selection. The indicated TT is instructed to move tape in the reverse direction at 225 ips. Once tape starts to move, the function ends and a Status Interrupt is signaled to the computer. When the computer acknowledges the Interrupt, the tape system becomes inactive. Actual rewind of the tape might be completed much later in time; *Rewind* is completed at the *start* of tape movement. The TT moves the tape to its beginning, stops tape movement, and notifies MTC that tape is rewound. Further references to this TT must call for forward direction of tape movement except when *Rewind* is commanded of a TT already rewound. *Rewind* to a TT already rewound terminates immediately because MTC is cognizant that this TT is rewound.

A function commanded of a TT that is rewinding before it reaches beginning of tape is locked out and not executed until *Rewind* is completed. The tape system will be active (that is, not available for another External Function command) until it completes the *Rewind* and the *Pending* function. Execution of the *Rewind* function is illustrated graphically in Figure D9-7.

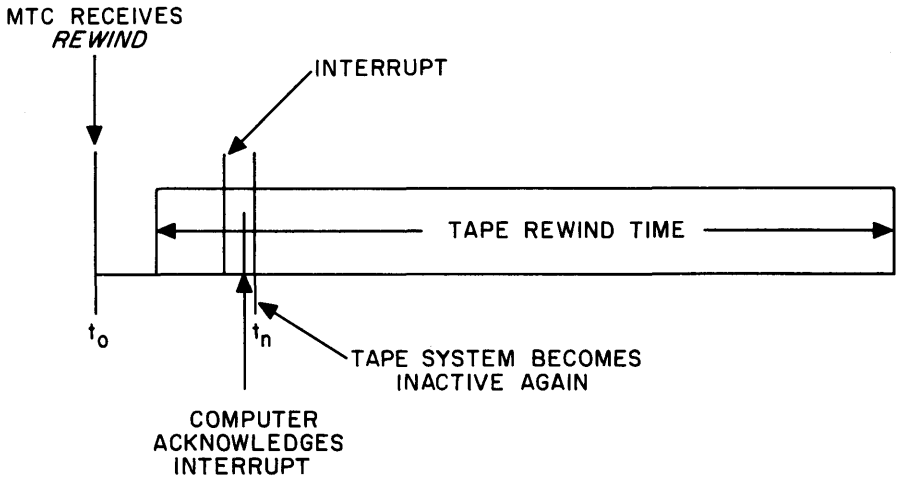


Figure D9-7. Execution of *Rewind* Function

(6) *Rewind with Lockout* (Bit 7)

This function is similar to the *Rewind* function described above except that the indicated TT is placed off-line (out of automatic) upon completion of the *Rewind* function. Further reference to this TT results in an Improper Condition. An indication of this condition appears in the Status Register of MTC. The *Rewind with Lockout* function may be used to prevent destroying pre-

viously recorded information and also to indicate to the computer operator that the tape on a certain TT should be removed. Physical intervention is required to place this TT back into the Automatic condition (*on-line* for computer reference).

(7) *Direction* (Bits 8 and 9)

These bits indicate selected direction of tape motion. Direction has no significance on functions other than *Read.* As illustrated in Figure D9-3, either bit 8 or bit 9 should be selected on a *Read* function. In a *Read* function, if both bits are set to *one,* a reverse tape direction results; if neither bit is set to *one,* a forward tape direction results.

(8) *Format* (Bits 10 and 11)

By appropriate bit selection, the desired format will be indicated. The tape system incorporates two formats: Univac Format and NTDS Format. Since it will be possible to *Read* or *Write* in either format, four combinations of format selections are possible:

1) Write Univac Format
2) Write NTDS Format
3) Read Univac Format
4) Read NTDS Format

These are discussed in Subsection 3.D in the order listed.

(9) *Tape Transport Selection* (Bits 12 and 13)

A *one* placed in the bit corresponding to the desired TT selects that TT. Duplexer functions and the *Master Clear* are the only functions that do not require TT selection. If a TT is not selected when one is needed, or if more than one are selected when *one* is needed, an Improper Condition signal will be sent to the computer via the Status Interrupt, and an Improper Condition will be indicated in the Status Register.

If the selected TT is inoperative (because of broken tape, not in Automatic condition, power off, etc.), an Improper Condition will be signaled to the computer via the Status Interrupt, and an Improper Condition will be indicated in the Status Register.

C. *STATUS WORD*

An Interrupt is sent to the computer at the end of every function. A Status word is placed on the 30 data lines of the input cable. The bit structure of the Status word enables the computer

to determine whether or not the previous function was successfully completed and, also, whether or not the computer is in control of the tape system.

The computer must recognize that after issuing an External Function code to the MTC, no subsequent External Function code can be legally issued until receipt of the Interrupt which signifies the end of the first one.

It should be noted that only two fundamental messages are conveyed to the computer via the Status word. These messages inform the controlling computer that either Improper Conditions or errors were incurred during the execution of the last function, or it is in control.

If bit 24 of the Status word is set to *one*, the Interrupt just received was a Control Acknowledge Interrupt. If bit 24 is *zero*, the Interrupt signifies the end of one of the three basic MTC functions of the Interrupt mode; the configuration of the remaining Status word bits will indicate successful or unsuccessful completion of that function.

Figure D9-8 shows bit assignments in the Status word. These conditions are described below.

(1) *Improper Condition* (Bit 29)

This bit set to one implies that operator intervention may be necessary to overcome the difficulty.

An Improper Condition will occur whenever

1) Referenced Tape Transport is not in Automatic condition
2) No Tape Transport is selected when one is required
3) Both Tape Transports are selected when only one is required
4) A forward command is sent to a Tape Transport whose tape is positioned at *End of Tape*
5) A reverse command is sent to a Tape Transport whose tape is positioned at *Beginning of Tape*
6) A reference is made to write on a Master Tape

When the computer has been notified of an Improper Condition, by bit 29 being set to *one*, the Improper Condition is indicated in the Status Register. The computer program may then refrain from issuing further External Function commands to the tape system to allow visual inspection of the trouble, or it may issue another External Function command. An incoming

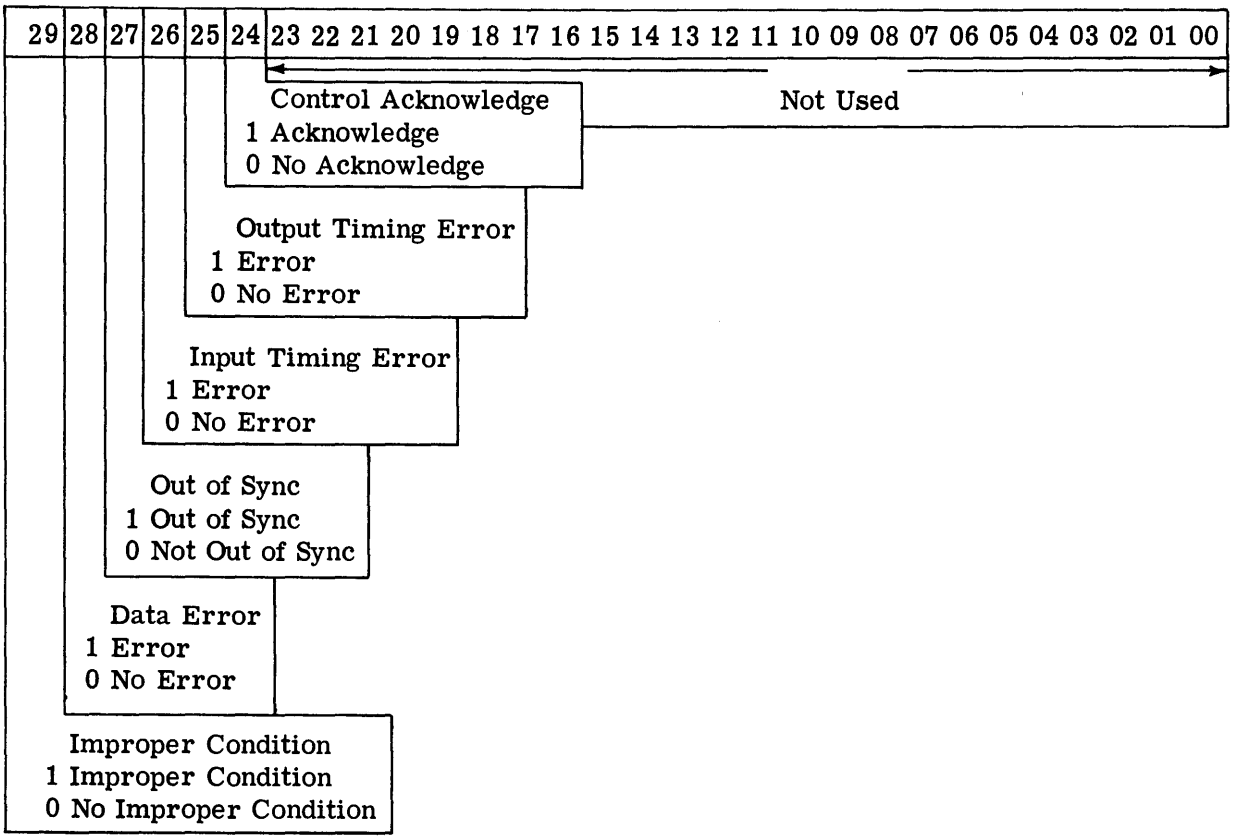


Figure D9-8. Status Word Format

External Function command to the tape system extinguishes the Improper Condition indication.

A TT not in Automatic condition implies one of the following situations:

1) TT was manually removed from Automatic

2) A *Rewind with Lockout* has occurred

3) TT not in *Ready* condition for one of the following reasons:

   - Power off
   - Tape broken
   - Lamp burnout
   - Tape load was not accomplished when tape was mounted

(2) *Data Error* (Bit 28)

This bit set to *one* informs the computer that one or more frames read in Univac Format did not show agreement between generated parity and read parity.

(3) *Out of Sync* (Bit 27)

This bit set to *one* indicates that at least one frame was not read from tape when it should have been; computer words read were not assembled correctly.

(4) *Input Timing Error* (Bit 26)

This bit set to *one* indicates that MTC information on the input cable was not accepted by the computer before the subsequent word was placed on the input cable. This condition indicates that the computer effectively "lost" one or more words of the *last* block.

(5) *Output Timing Error* (Bit 25)

This bit set to *one* indicates that the computer did not place information on the output cable in time for the tape system to write it in the proper position on the moving tape. It also indicates, however, that the computer did acknowledge the Output Request within 140 microseconds after the End of Block was declared.

(6) *Control Acknowledge* (Bit 24)

This bit when set to *one* indicates to the computer that it has just received control of the Magnetic Tape System. This is a status condition sent to the computer by Interrupt *only* in response to a *Request Control* Function code. It is a Duplexer function.

D. *TAPE SYSTEM FORMATS*

(1) *Write Univac Format*

With Univac Format, information is written on tape at a density of 128 lines per inch. To be compatible with the High-Speed Printer and the Univac File Computer Tape System, blocks of information have 120 lines or frames. Each frame contains six data bits, one parity bit (odd), and one sprocket bit; thus, a block contains 24 computer (30-bit) words. Figure D9-9 shows one 30-bit word (Bits 29 through 0) written on tape in Univac Format. Twenty-three more words are written in the same manner to comprise one block of information in Univac Format. Writing occurs on tape that moves forward at 112-1/2 ips. Word rate during writing is 2.88 kc. Average word rate is approximately 1.2 kc, owing to a space of 1.2 inches between blocks.

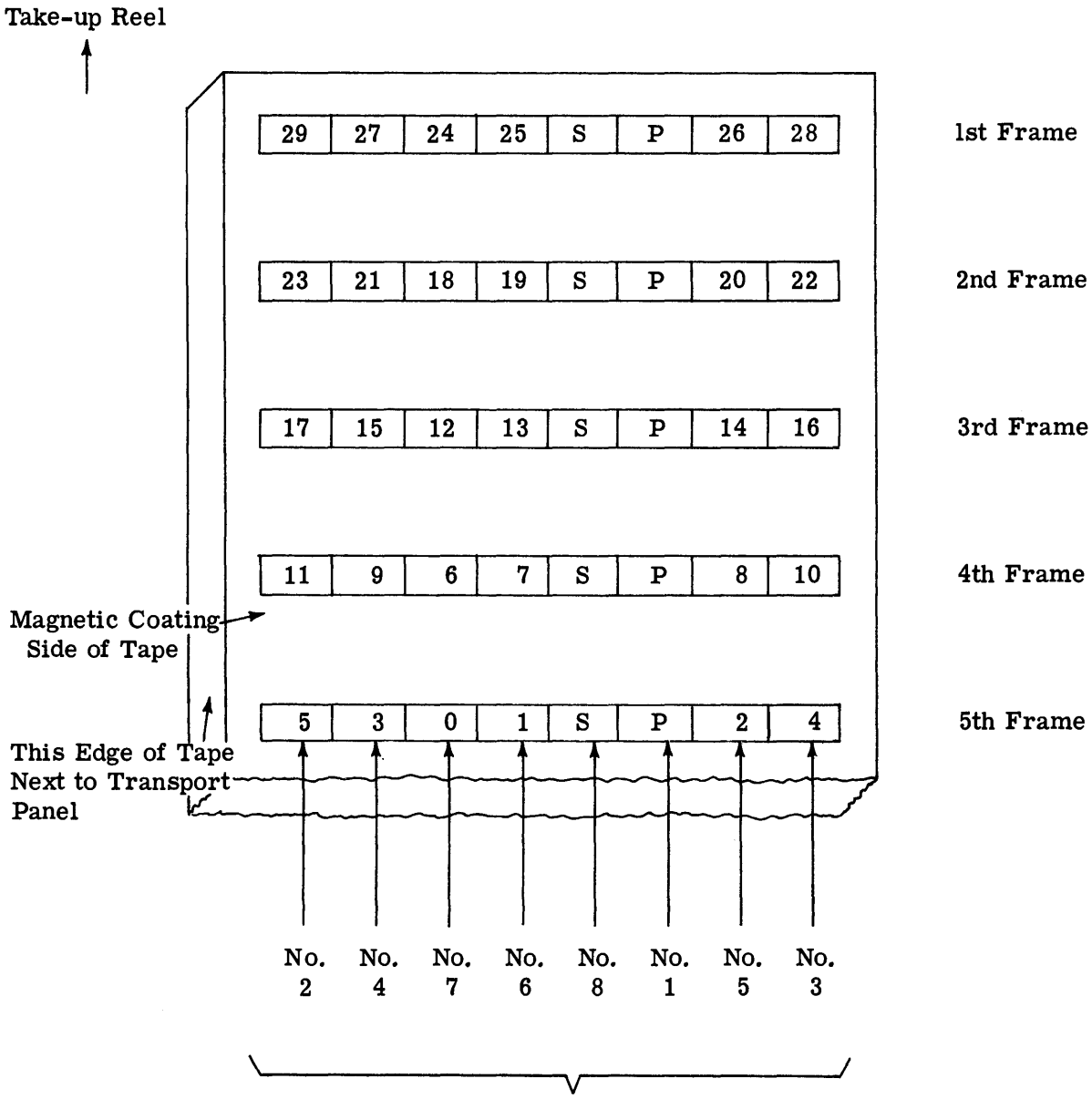


Figure D9-9. Univac Format

(2) *Write NTDS Format*

With NTDS Format, information is written on tape at a density of 200 lines per inch. Each line or frame on tape contains three data bits and one sprocket bit, each written twice.

As illustrated in Figure D9-10, a computer word is written in 10 frames. A block in NTDS Format may vary from 24 computer words to 16,000 computer words; block length is established when the buffer is initiated in the computer. Space between blocks is 1.2 inches in both NTDS and Univac Formats. Word-transfer rate of NTDS Format is 2.25 kc (200 lines per inch; tape speed, 112-1/2 ips) not counting block spaces.

Figure D9-11 shows the order in which the eight bits are written in any one frame. X_1, X_2, X_3 and Y_1, Y_2, Y_3 represent the two 3-bit groups of data; $X_1 = Y_1$, $X_2 = Y_2$, $X_3 = Y_3$. S_1 and S_2, the sprocket bits, are of course equal.

(3) *Read Univac Format*

During a *Read* function in Univac Format, 8-bit characters are transferred from TT to MTC at a rate of approximately one every 70 microseconds. MTC assembles these incoming frames on either Read Reverse or Read Forward operations into computer words of their original structure. As each frame is read, MTC generates an odd-parity bit for the six data bits read. This generated parity is compared to the read parity bit. If they agree, MTC assumes the six bits of data are correct. If they disagree, MTC "remembers" the failure and notifies the computer via the Status Interrupt (at the end of block) that a parity disagreement or data error was detected. Such notification via the Status word implies that data errors were detected in one or *more* of the 120 frames of the *last* block. If no data errors were detected, the Data-Error bit in the Status word is left *zero*.

Transfer of data from TT to MTC continues until the end of the block is reached. There, because MTC senses that no transfer takes place, the *Read* function is terminated. If an error causes loss of an entire frame while reading 120 frames, the last computer word is not completely assembled. This implies that somewhere in the block of 24 computer words, one (or more) group of six data bits is missing. As a result, all data that follow are displaced in the word by 6-bit multiples. This is called an *Out of Sync* condition. If MTC detects that a word is not completely assembled by the time that the block ends, a bit in the Status word will be set to signify *Out of Sync*.

Reading is done at a tape speed of 112-1/2 ips; thus, the word rate is the same for *Read* as for *Write* functions on either Univac or NTDS Formats (approximately 1.2 kc for Univac For-

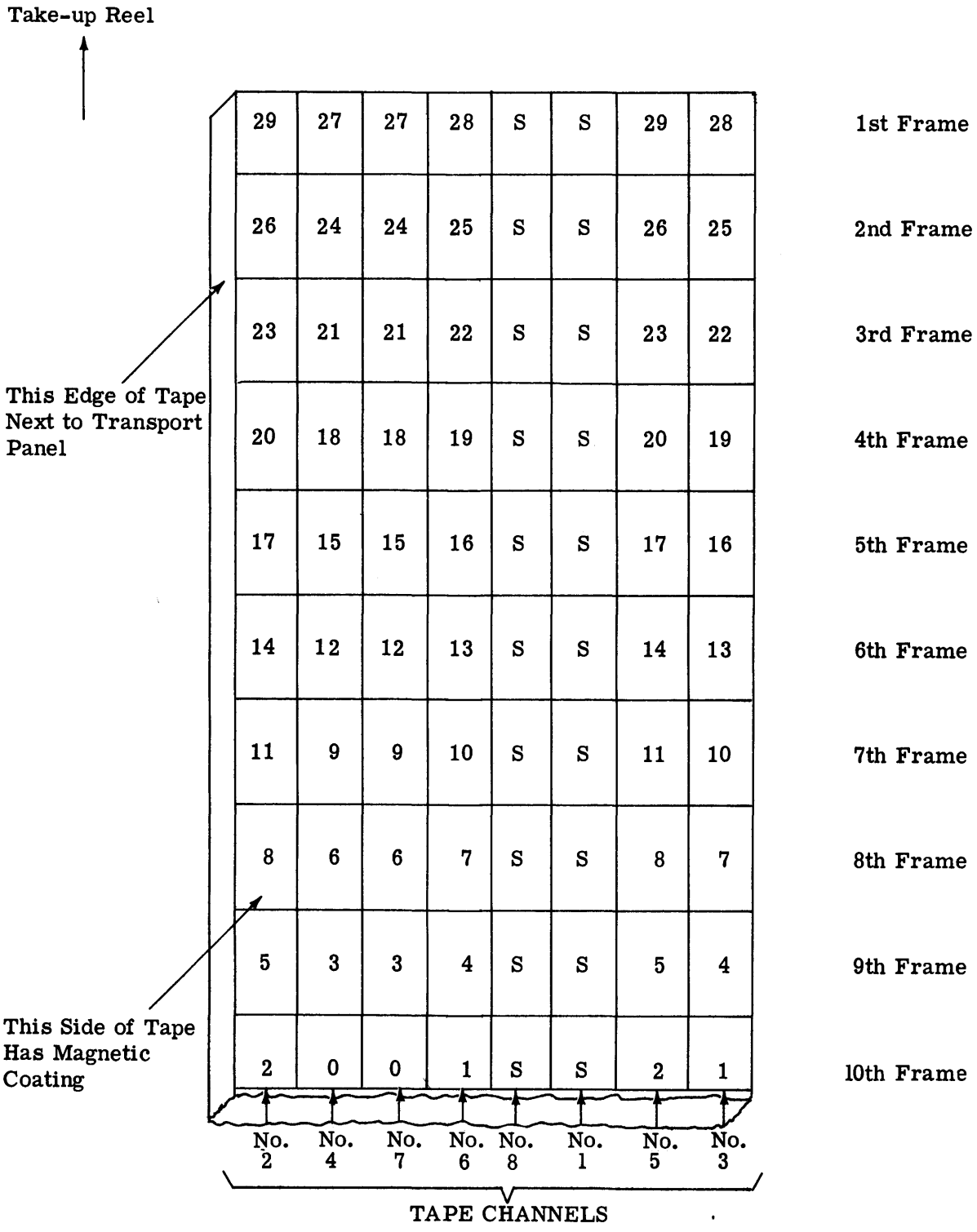


Figure D9-10. NTDS Format

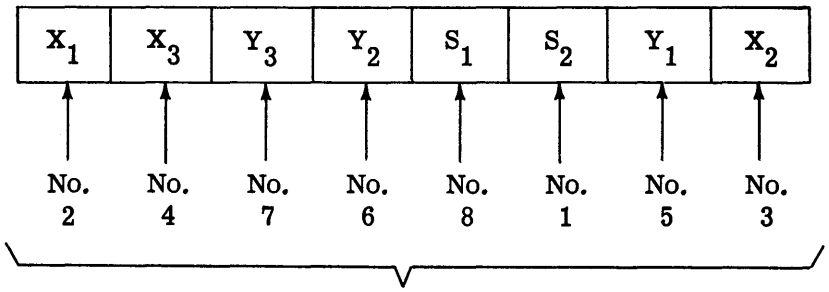


Figure D9-11. Bit Arrangement of Frame in NTDS Format

mat and 2.25 kc for NTDS Format).

(4) *Read NTDS Format*

The basic logic of Read NTDS Format is similar to Read Univac Format with minor differences in frame rate and word rate. Frames are transferred at a rate of one per 44 microseconds, and the word rate is 2.25 kc. Three bits, instead of six, are assembled at one time, and 10 frames are read to assemble one word. As in Univac Format, an *Out of Sync* condition may occur during reading. This condition is detected by MTC, and the computer is informed in the manner previously discussed. In NTDS Format, an *Out of Sync* implies that the data in the words are misplaced in all words beyond the *lost* frame by three bits instead of six bits.

There is a major difference in error-deterrent techniques for the two Read Formats. In Univac Format, a data error indicated by the Status word signifies disagreement between generated and a read parity. In NTDS Format, there is no parity bit, but two 3-bit data groups are read from the same frame. As these two groups are read, the *sum* of corresponding bits is produced and used as *the* final three bits of data from a frame. Again referring to Figure D9-11, X_1, X_2, X_3 and Y_1, Y_2, Y_3 represent the two groups of data where $X_1 = Y_1$, $X_2 = Y_2$, $X_3 = Y_3$ and S_1 and S_2 are sprocket bits where $S_1 = S_2$.

The three bits from this frame assembled into the final word are

$X_1 + Y_1$,
$X_2 + Y_2$, and
$X_3 + Y_3$.

+ is the symbol used to indicate the "OR".

E. *MULTIPLE EXTERNAL FUNCTIONS*

If the Magnetic Tape System receives an External Function other than *Master Clear* while it is actively engaged in the operation of an earlier function, the later External Function will be ignored.

F. *LOGICAL SELECTION OF TAPE TRANSPORTS*

A selector switch to change the logical number of a TT will be provided. Physical TT No. 1 may be switched to logical TT No. 2 and vice versa. Logical TT numbers will not be duplicated. This eliminates chances of both TT's being assigned the same logical number.

This selection switch will allow programs that reference a particular logical Tape Transport to be run even when the corresponding physical TT may be malfunctioning.

G. *DETECTION OF BLOCK END BY MTC*

During a Write operation, MTC senses that a block should end whenever a new computer word is not placed on the output cable in response to an *Output Request* within a prescribed time period. *Block End* is initiated and tape movement is stopped after a delay to allow for a 1.2-inch block space. (The full block space will include the Stop delay and the Start delay.) A Status Interrupt is sent to the computer to signify the end of the *Write* function. After receipt and acknowledgement of the Status Interrupt, if the computer issues another External Function command calling for a *Write* on the same TT within the duration of the Stop delay, a block space is generated on tape but the tape does not stop. This feature accomplishes two purposes: It allows continuous tape motion on successive Write operations (to save time) and it reduces wear on the pinch-roller mechanism. On successive Write operations, the computer issues External Function codes within five milliseconds to assure continuous tape motion.

During a Read operation, MTC senses that a block should end when 8-bit frames are no longer transferred from TT to MTC. Detection of the Block End initiates termination of the *Read* function, calling for a Stop delay and a Status Interrupt. Successive *Read* functions (just as successive *Write* functions) in the same direction on the same TT allow continuous tape motion when the computer issues External Function codes within the allotted time. For successive *Read* functions, this time is also five milliseconds.

H. *MAINTENANCE AIDS*

In keeping with general requirements for this tape system, simple maintenance features will be provided. Switches and manual insertion registers will provide the means for operating

and visually inspecting the tape system off-line to the computer. It is expected that a large share of the malfunctions will be overcome and some preventive checks on the system will also be possible by these methods.

Emergency situations and preventive maintenance periods that demand a dynamic checkout of the tape system will require maintenance routines with a computer. Since most peripheral equipment associated with NTDS communicate with two computers, it should be possible, where necessary, to "release" one computer for testing the tape system under actual operating conditions. Certain maintenance aids will be included to test the tape system in marginal conditions.

The following list of items is considered essential to allow the proper degree of maintenance:

1) A means for reading in high and low marginal conditions will be provided.

2) The ability to perform all tape-system functions by manual insertion of the codes into a register. This will probably necessitate a manual Start switch.

3) The ability to write *ones* on all tape channels in variable-length blocks and read them out to permit the maintenance man to check the read levels with a scope.

4) A variable Test Oscillator for Write and Read operations in a slow mode to allow visual monitoring of assembly and disassembly in either format.

5) A means of inhibiting either side (X or Y) of the "OR" input to the three data bits of NTDS Format. Maintenance routines can thus determine when certain channels deteriorate on NTDS Format Read operations. The routine can be run first with neither side inhibited, then with X inhibited, and finally with Y inhibited. Words received by the computer during these three runs should indicate reliability of the "OR" inputs.

I. *WRITE INHIBIT ON MASTER TAPE*

RRU-type reels used with this tape system have slots inside the hub to receive a metallic ring. When the ring is in place, it signifies a Master Tape. This is sensed by MTC which allows no Write operations to be performed on this tape.

If reference is made to *Write* on a Master Tape, an Improper Condition will result. The Improper Condition will be indicated in the Status Register.

J. *TAPE TRANSPORT CONTROL*

(1) *Tape Transport Status Signals*

Each TT will inform MTC when the following indications exist:

(a) *Master Tape* - When a ring is fitted in the slotted hub of the RRU reel, TT will inform MTC that a Master Tape is mounted on the transport. (Writing will not be performed on this tape.) When a Master Tape is mounted on a TT, an indicator will be illuminated.

(b) *End and Beginning of Tape* - The TT will inform MTC when a tape is positioned at its end or beginning. This will be indicated by illuminated MTC indicators.

(c) *Operational Availability* - If any of the below conditions exist, a signal will be sent to MTC from the TT. Any of these conditions will extinguish the READY indicator of the TT associated with the condition and will remove the TT from Automatic (that is, will place it off-line).

1) Power Off
2) Broken Tape
3) Lamp Burnout

(2) *Switches For TT*

The following TT control switches will be located on the TT control panel:

1) POWER ON/OFF
2) AUTOMATIC (on-line to computer)
3) FORWARD
4) REVERSE
5) CLEAR-STOP
6) REWIND

(3) *Indicators for TT*

The following indicators will be located on the TT control panel:

1) BEGINNING OF TAPE
2) END OF TAPE

3) MASTER TAPE

4) READY

5) SELECT

6) CLEAR-STOP

7) AUTOMATIC

4. DUPLEXER SPECIFICATIONS

For Duplexer Specifications and a discussion of the External Functions performed by the Duplexer portion of the Tape System, refer to Section D11 of this document, "Functional Specifications for Peripheral Equipment Duplex Operation."

5. PROGRAMMING CONSIDERATIONS

This subsection includes certain considerations which stress programming responsibilities rather than system capabilities. The system specified herein is *simple* in nature; therefore, the task of the programmer is great for he must "program around" the limitations that would not exist in a more complex tape system. Items are discussed in this subsection in a concise manner. A more complete discussion of programming this system is to be found in the appropriate section of RRU Document PX 1489, *Planning for Service Test Programming.*

A. *CHECK SUM*

The last word of each written block will be a computer word that contains the sum of all preceding half-words in the block. This sum will be computed by considering each computer word as two 15-bit words (Bits 29-15 and Bits 14-0) and by taking the sum of all half-words in the block.

This check sum will be included in all blocks that are to be read back. Whether or not it is to be included on tapes to be read by other compatible systems will depend on read routines of those systems.

In Read operations, the final decision as to the correctness of the Read data will always depend on agreement of read check sum with a computed check sum. If they agree, the data are assumed correct. If they do not agree, the data are assumed incorrect.

B. *TAPE SYSTEMS CONTROL*

External Function commands, other than *Request Control* and *Release Remote,* from a com-

puter not in control of the tape system will be ignored by the tape system. Programmers must remember when a computer maintains control of the tape system.

C. *TAPE SYSTEM AVAILABILITY*

Once the tape system receives and starts to execute an External Function code, further External Function codes from the controlling computer, other than *Master Clear,* are ignored. The programmer must remember when External Function codes may be logically issued. After issuing any External Function code other than *Master Clear, Release Local,* or *Release Remote,* the computer may not legally issue another until receipt of an Interrupt from the Tape System.

D. *BLOCK LENGTH*

In Univac Format, all blocks must be 24 computer words in length. The programmer must take this into account when he sets the Output Buffer on Write operations.

In NTDS Format, the minimum block length is 24 words and the maximum is 16,000 words. The programmer must stay within these limits when setting the Output Buffer. The maximum block length is equal to the approximate number of words that may be *check summed* (in the manner previously described) without exceeding a 30-bit modulus, assuming all 15-bit words are of maximum value.

E. *END OF TAPE*

If, in running a program, it is possible to write more words than one tape will hold, it is the responsibility of the programmer to prevent writing *off the end* of the tape. No *End-Of-Tape* warning is issued; however, once the end of a tape is reached, the TT will "Not be Ready" to forward references.

F. *RECORD END*

In reading a tape with an unknown number of blocks, the programmer must determine when the *last block* is read. It may be performed by writing a special code in the *last block* and monitoring this code when reading.

G. *SEARCH AND MOVE*

These functions are not included in the Function word and must be implemented by the programmer. One-word Input Buffers and continuous tape motion might assist the programmer in these tasks.

H. *UNKNOWN BLOCK LENGTH*

When reading NTDS Format tapes of unknown block lengths, the programmer must make the Input Buffers sufficiently large to ensure reading the complete block. Otherwise, words will be lost and an input timing error will result.

I. *EDITING THE TAPE*

It is not possible to Write in the middle of recorded information without destroying a portion of information following what was written because of the placement of the broad Pre-Erase Head with respect to the Write Head.

SECTION D10

FUNCTIONAL AND PROGRAMMING SPECIFICATIONS

FOR THE

FORMAT CONTROL UNIT

CONTENTS

PART ONE

FUNCTIONAL SPECIFICATIONS

CONTENTS (Cont.)

PART TWO

PROGRAMMING SPECIFICATIONS

ILLUSTRATIONS

FUNCTIONAL AND PROGRAMMING SPECIFICATIONS FOR THE FORMAT CONTROL UNIT

PART ONE

PROGRAMMING SPECIFICATIONS

1. BASIC INFORMATION

The Format Control Unit (FCU) is a device which permits an AN/USQ-20 Unit Computer to communicate with as many as seven Univac File Computer Magnetic Tape Units, Type 4950. This tape system is intended for computing center applications. Type 4950 tape units used in this tape system are identical to those used with the Interim Tape System Adapter (ITSA) for the AN/USQ-17 Unit Computer.

This part of this section describes functional aspects of FCU: Equipment interconnections, timing, FCU registers, computer control of the tape system, data transfer, and FCU maintenance panel.

FCU performs the following functions:

1) Enables the computer to exercise *demand station* control of individual tape units
2) Performs parallel-to-serial conversion of data transferred from computer to MTU and serial-to-parallel conversion of data transferred from MTU to computer
3) Generates and supplies all external timing pulses required by the tape units.

Type 4950 tape units are described in detail in the following publications:

1) RRU document PX 738, *Service Manual, Magnetic Tape Unit, Type* 4950
2) RRU document PX 581, *Circuit Diagrams for Magnetic Tape Unit, Type* 4950

2. EQUIPMENT INTERCONNECTIONS

A. *FCU/COMPUTER INTERFACE*

FCU communicates with the AN/USQ-20 Unit Computer via two cables, one normal computer output channel and one normal computer input channel (see Figure D10-1).

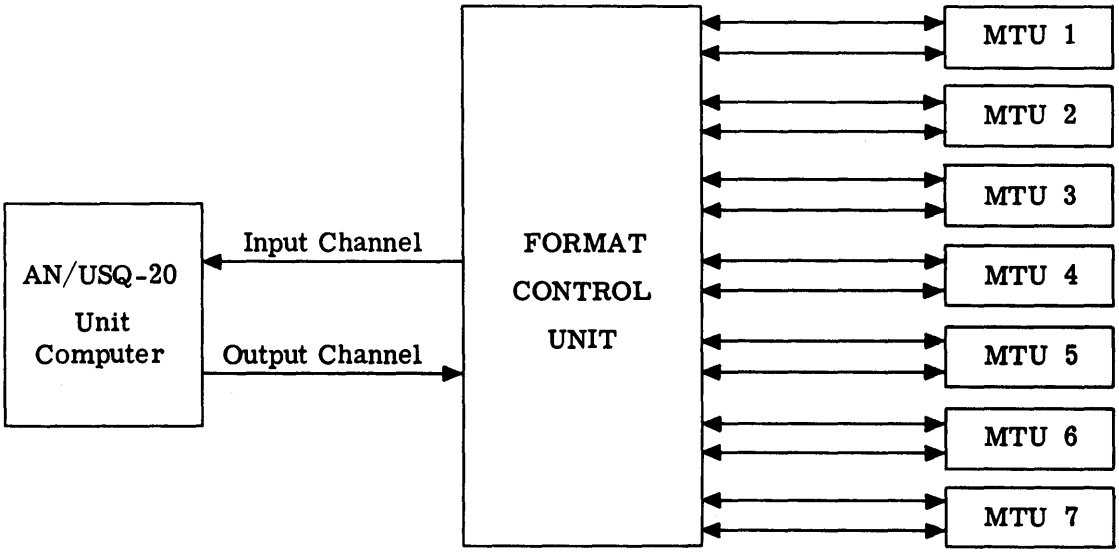


Figure D10-1. Interconnections of Tape System Equipment

(1) *Computer Output Channel*

This channel carries function commands and parallel, 30-bit output data from the computer to FCU. The channel consists of 30 information lines, one *Output Data Request* line, one *Output Acknowledge* line, and one *External Function* line.

(2) *Computer Input Channel*

This channel carries equipment status information and parallel, 30-bit input data from FCU to the computer. The channel consists of 30 information lines, one *Input Data Request* line, one *Input Acknowledge* line, and one *Interrupt* line.

Communication across FCU/Computer interface is subject to rules and restrictions imposed by NTDS Technical Note No. 233, *Revised Input/Output Specification for the AN/USQ-20 Unit Computer and Associated Equipment.*

B. *FCU/MTU INTERFACE*

FCU communicates with each tape unit via two cables. These cables carry control signals, timing signals, and serial output data from FCU to MTU, and control signals, status signals, and serial input data from MTU to FCU.

Signal lines from FCU to each MTU include the following:

1) Test In
2) Demand In
3) Demand Clear
4) A Instruction
5) B Instruction
6) C Instruction
7) D Instruction
8) Master Clear
9) TA Clock
10) TB Clock
11) CMA-1 Clock
12) RMA-6 Clock
13) *Serial Data "1's"*
14) *Serial Data "0's"*

Signal lines from each MTU to FCU include the following:

1) Ready
2) Not Ready
3) Normal Demand Out
4) MTU Error
5) W Status
6) X Status
7) Y Status
8) Z Status
9) *Serial Data "1's"*

Demand In signal is sent only to the addressed MTU. The *Serial Data "1's"* from MTU to FCU and the *Not Ready* signal are gated only from the addressed MTU. None of the other signals are gated by the address bit.

All signals sent across FCU/MTU interface are in the form of pulses. The standard pulse has the following characteristics:

1) Based at 0 volt
2) Minimum amplitude: -15 volts
3) Pulse width, T: 0.75 μsec < T < 1.0 μsec.

3. TIMING SIGNALS

FCU generates all timing pulses required by the tape units to control their internal operations and to assure synchronous data transfer. In addition, FCU generates timing signals to control its own internal sequences.

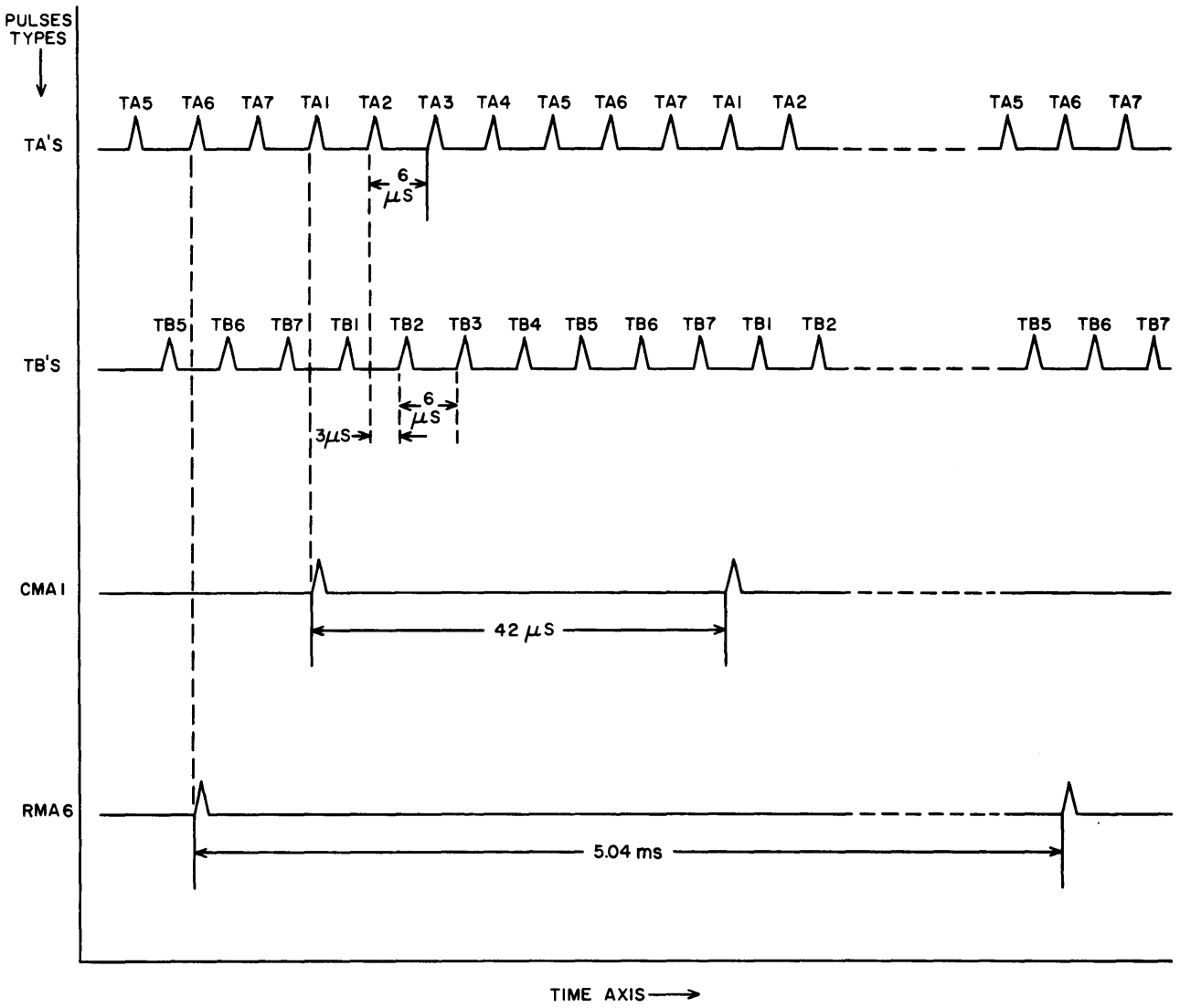


Figure D10-2. Timing Signals Supplied to MTU by FCU

The following timing pulses are supplied to each MTU on a continuous basis (see Figure D10-2).

1) TA - the basic pulse generated once each 6 $\binom{+0.3\ \mu s}{-0.0\ \mu s}$ microseconds.

2) TB - also generated once each 6 microseconds, but it is delayed from TA by 3 $\binom{+0.2\ \mu s}{-0.0\ \mu s}$ microseconds.

3) CMA 1 - generated once each 42 microseconds, coincident with a TA-1 pulse

4) RMA 6 - generated once each 5.04 milliseconds, coincident with a TA-6 pulse, precedes a CMA-1 pulse by 12 microseconds.

4. FCU REGISTERS

FCU has the five registers listed below which perform the described duties.

A. *FUNCTION REGISTER (F)*

This register holds part of the most recent External Function command from the computer. It is divided into three sections:

1) FCU instructions (2 bits)
2) MTU address bits (7 bits)
3) FCU availability bit (1 bit)

B. *STATUS REGISTER (S)*

This register is an 18-bit register that holds all status information that pertains both to FCU and to the tape units.

C. *CHARACTER REGISTER (X)*

This is a 6-bit shift register used to serialize a character of output data to an MTU and staticize a character of input data from an MTU.

D. *ASSEMBLY/DISASSEMBLY REGISTER (Z)*

During output data transfer, this 30-bit register holds the 30-bit computer word while it is disassembled and shifted out serially to an MTU. During input data transfer, this register accepts data from the $\tilde{X}$ register and assembles a 30-bit computer word.

E. *COMMUNICATION REGISTER (C)* (30 bits)

This register performs buffering function between the computer input/output channels and the Z register in FCU. During input data transfer, C takes an assembled word from the Z register and holds it until the computer accepts it. During the output data transfer, C accepts a word from the computer and holds it until the command C→Z is generated. Also, all External Function commands and interrupt codes pass through the C register.

5. COMPUTER CONTROL OF THE TAPE SYSTEM

A. *EXTERNAL FUNCTION COMMAND*

The computer sends External Function commands on the output channel to FCU to control operation of the tape system. In general, each External Function command includes an instruction addressed to a specific MTU and an instruction to FCU. An MTU can execute the following instructions:

1) Write
2) Read Forward
3) Read Backward
4) Search Forward Equal
5) Search Backward Equal
6) Search Forward, Equal To or Greater Than
7) Search Backward, Equal To or Less Than
8) Transfer Blockette, MTU to Computer
9) Transfer Blockette, Computer to MTU
10) Wind Forward
11) Wind Forward with Interlock
12) Rewind (Wind Backward)
13) Rewind with Interlock

(The *Write and Check* instruction of MTU is not implemented during operation with FCU.)

FCU can execute two major sequences:

1) Transfer Data In (to computer)
2) Transfer Data Out (from computer)

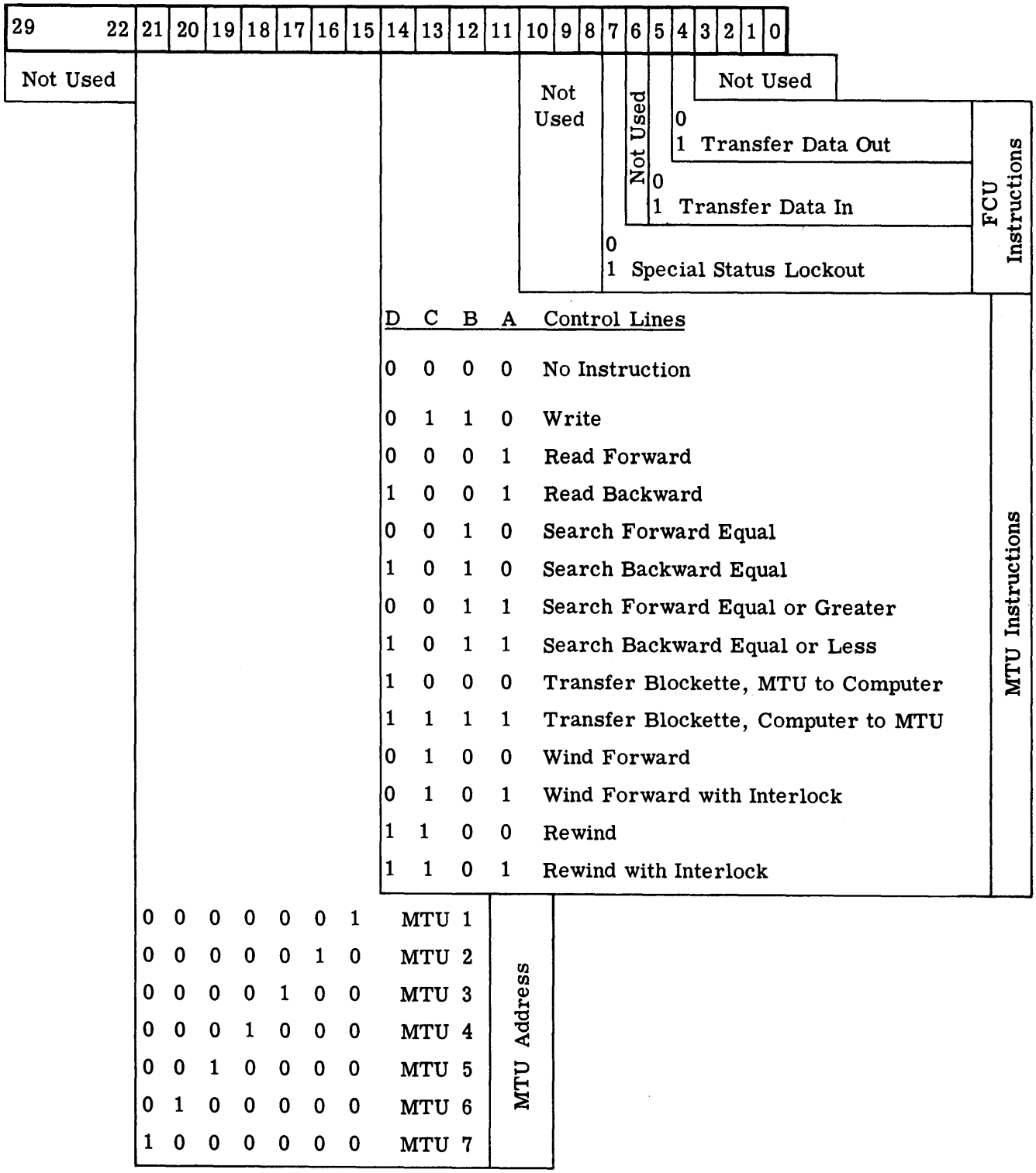


Figure D10-3. External Function Word (Computer to FCU)

The External Function word received by FCU has the configuration shown in Figure D10-3 and listed below.

Bits 0 through 3	Not Used
Bit 4	Transfer Data Out
Bit 5	Transfer Data In
Bit 6	Not Used
Bit 7	Special Status Lockout
Bits 8 through 10	Not Used
Bit 11	MTU Instruction (line A)
Bit 12	MTU Instruction (line B)
Bit 13	MTU Instruction (line C)
Bit 14	MTU Instruction (line D)
Bit 15	Address Bit, MTU 1
Bit 16	Address Bit, MTU 2
Bit 17	Address Bit, MTU 3
Bit 18	Address Bit, MTU 4
Bit 19	Address Bit, MTU 5
Bit 20	Address Bit, MTU 6
Bit 21	Address Bit, MTU 7
Bits 22 through 29	Not Used

The MTU instruction (bits 11 through 14) and FCU instruction (bits 4 and 5) must be matched. That is, an MTU instruction which requires data from the computer (Write, Transfer Blockette, Computer to MTU, all Search operations) must be accompanied by FCU instruction, Transfer Data Out.* Similarly, MTU instruction, Transfer Blockette, MTU to Computer, must be accompanied by FCU instruction, Transfer Data In.** None of the other MTU instructions involve data transfer through FCU. After a Read or a Search (satisfied) operation, data remains in MTU buffer memory; therefore, these operations must be followed by a Transfer Blockette, MTU-to-Computer instruction.

Only one MTU address bit may be set in any External Function command.

*Failure to do this will cause the MTU to stop with a parity error.

**Failure to do this will prevent the transfer of data but no error will occur.

B. *FCU AVAILABILITY*

Before the computer can send an External Function command, it must ascertain that FCU is capable of accepting and performing it. FCU is said to be *Available* when it is prepared to accept an instruction. When FCU accepts an External Function command, it sets its status to *Not Available*. It is the responsibility of the computer program to remember when FCU is *Available*. A new External Function command may not be issued until an interrupt has been received from FCU after it has responded to the previous External Function command. It remains *Not Available* during the time that it is engaged in a data-transfer sequence or during the exchange of control information with an MTU. While in the *Not Available* state, FCU locks out (ignores) any External Function commands sent by the computer.

Whenever FCU completes its internal sequence, it sends a status-report interrupt signal on the computer input channel. When the computer responds with an *Input Acknowledge* signal, FCU reverts to the *Available* state. Along with this interrupt, FCU places contents of its Status register on the input channel. (See subsection 5.D for Status word format.) Receipt of the interrupt always implies an FCU *Available* condition and the computer is then free to send an External Function command.

C. *MTU READINESS AND TERMINAL CONDITIONS*

Control information which an MTU sends to FCU includes readiness indication and terminal condition information.

(1) *MTU Readiness*

A tape unit is *Ready* if it has completed its last operation and is prepared to accept another instruction. It is *Not Ready* as long as it is performing its internal sequences or if an MTU error condition is present. When FCU desires to determine the ready status of an MTU, it sends a Test In pulse to the unit. This signal samples the *Ready* flip-flop of the MTU and returns either a *Ready* pulse or a *Not Ready* pulse to FCU. FCU will not send an instruction to an MTU unless a *Ready* signal has been returned.

(2) *Terminal Conditions*

Each tape unit is capable of detecting seven terminal conditions:

1) End of File - MTU detects tape sentinel consisting of *Z* characters
2) End of Data - MTU detects tape sentinel consisting of % characters
3) End-of-Blockette Count - MTU compares automatic counter against preset limit
4) Beginning of Tape - MTU detects clear tape leader

5) End of Tape - MTU detects clear tape leader

6) Search Equal Find

7) Search Unequal Find.

There are four Special Status lines from each tape unit to FCU (W,X,Y,Z lines). Therefore, only four of the seven detectable conditions may be reported. The desired terminal condition must be patched at the pinboard of the control panel of the tape unit to one of the Special Status lines. When FCU sends a Demand In pulse to the MTU, MTU responds with either a *Normal Demand Out* or a *Terminal Condition* pulse on one of the W,X,Y,Z lines. A *Normal Demand Out* signifies that none of the above-mentioned terminal conditions are present and plugged.

D. *STATUS-REPORT INTERRUPT*

The status-report interrupt which the computer receives on the input channel from FCU has the configuration shown in Figure D10-4 and is listed below.

Bit 29	Ready Indicator, MTU 1	
Bit 28	Ready Indicator, MTU 2	
Bit 27	Ready Indicator, MTU 3	
Bit 26	Ready Indicator, MTU 4	
Bit 25	Ready Indicator, MTU 5	
Bit 24	Ready Indicator, MTU 6	
Bit 23	Ready Indicator, MTU 7	
Bits 22 through 15	Not Used	
Bit 14	MTU Terminal Condition (line W)	Special Status
Bit 13	MTU Terminal Condition (line X)	
Bit 12	MTU Terminal Condition (line Y)	
Bit 11	MTU Terminal Condition (line Z)	
Bits 10 through 7	Not Used	
Bit 6	I/O Timing Error	
Bit 5	Parity Error	
Bit 4	No MTU Response	
Bit 3	MTU Error	
Bit 2	Special Status Present	
Bit 1	Addressed MTU *Not Ready*	
Bit 0	Abnormal Condition Present	

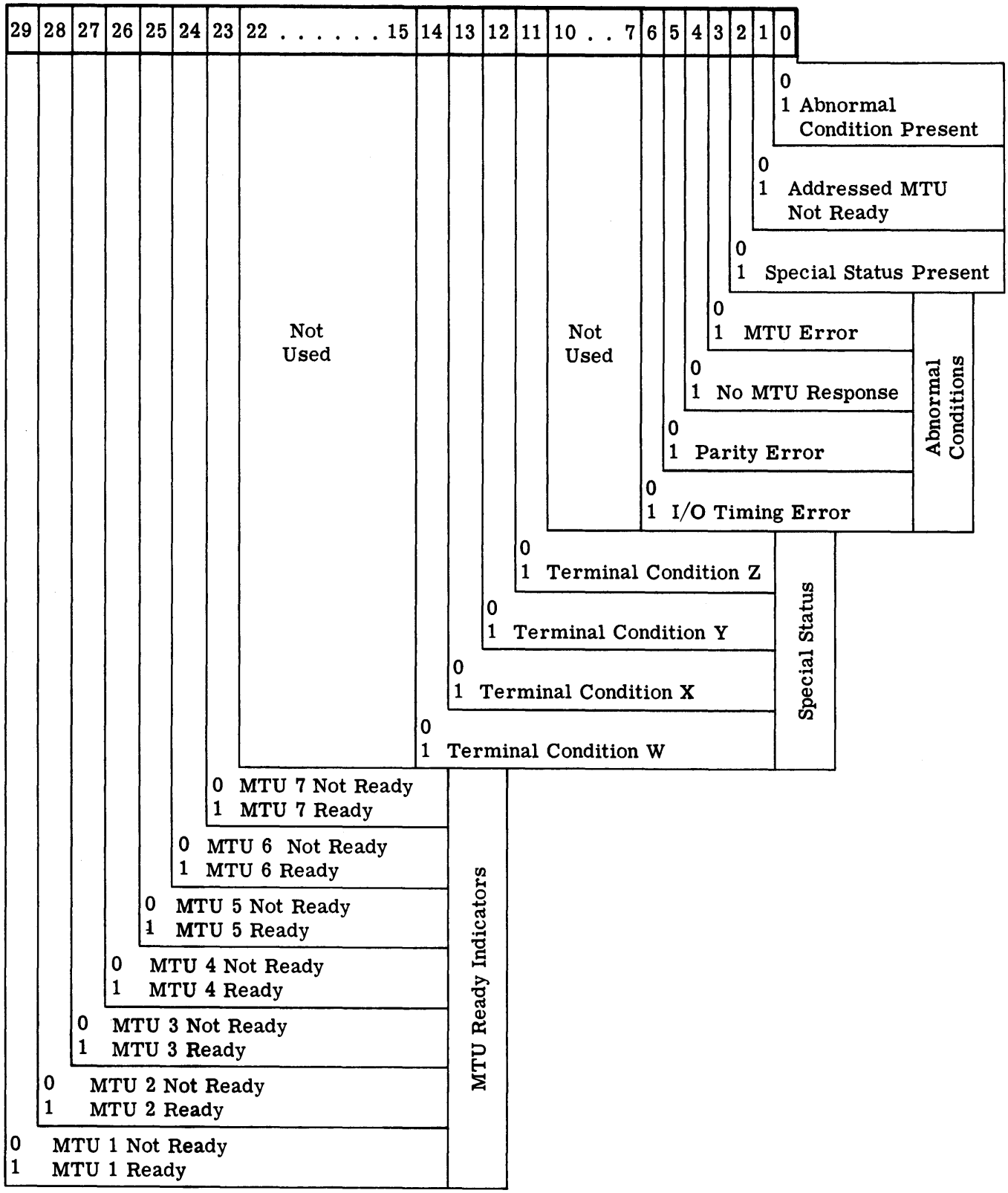


Figure D10-4. Status Report Word (FCU to Computer)

All tape unit *Ready* indicators (bits 23 through 29) are included with every interrupt sent to the computer. Whenever any MTU is demanded, all units are tested for readiness and bits 23 through 29 are updated.

FCU sets bit 1 when it determines that the addressed MTU is in a *Not Ready* condition.

Special Status bits (11 through 14) contain terminal condition information which has been detected by the MTU and sent on the W,X,Y,Z lines. Whenever one of these bits is set, FCU also sets bit 2.

Bits 3 through 6 represent specific errors or abnormal conditions which are detected by FCU or by the addressed MTU. Whenever one of these bits is set, FCU also sets bit 0 as an indicator.

E. *CONTROL SEQUENCES*

After FCU accepts an External Function command from the computer, it must exchange certain control and status signals with the addressed MTU before it can enable its own data-transfer sequence or send the instruction to the MTU. On the basis of the response signals from the MTU, FCU decides whether or not to proceed with the commanded operation. If FCU does proceed, it sends a status-report interrupt as soon as it has completed its task and is *Available* for another command. If FCU decides not to proceed, it immediately sends a status-report interrupt to inform the computer of the reason for not performing the operation.

The sequence of events is the following:

1) FCU receives an External Function command from the computer.

2) FCU issues *Test In* and *Demand Clear* signals to all units, and a *Demand In* signal to the addressed unit.

3) The addressed MTU responds with one or more of the following signals:

   a) *MTU Error*

   b) *Not Ready*

   c) *Ready*

   } *all* tape units send either a *Ready* or a *Not Ready* signal

   d) *Normal Demand Out*

   e) *Special Status* (W,X,Y, or Z)

4) FCU stores MTU response signals in the Status Report according to the rules described below.

Status Bit	*Condition for Setting*
29	*Ready* signal returned by MTU 1
28	*Ready* signal returned by MTU 2
27	*Ready* signal returned by MTU 3
26	*Ready* signal returned by MTU 4
25	*Ready* signal returned by MTU 5
24	*Ready* signal returned by MTU 6
23	*Ready* signal returned by MTU 7
14	Signal returned on W line and Special Status LOCKOUT not active
13	Signal returned on X line and Special Status LOCKOUT not active
12	Signal returned on Y line and Special Status LOCKOUT not active
11	Signal returned on Z line and Special Status LOCKOUT not active
6, 5	These bits can be set only during transfer of data
4	Neither *Ready* nor *Not Ready* signal returned by addressed MTU
3	*Error* signal returned by MTU
2	Special Status LOCKOUT not active and a signal returned on W *or* X *or* Y *or* Z line
1	*Not Ready* signal returned by *addressed* MTU
0	Bit 3 *or* Bit 4 *or* Bit 5 *or* Bit 6 set

5) FCU examines bits 0, 1, and 2 in the Status register and proceeds as follows:

a) If: None of these bits are set

Then: FCU proceeds with the commanded operation. It sends instruction to the addressed MTU and, if required, it enables a data-transfer sequence (see Subsections 6.B and 6.C).

b) If: One or more of these bits is set

Then: FCU does not proceed with the commanded operation. Instead, it immediately sends a status-report interrupt to the computer.

6. DATA TRANSFER

A. *DATA FORMAT*

The unit of data transfer between computer and MTU is the blockette. A blockette consists of 120 characters. In the MTU,a character is represented by seven bits, six information bits and one parity bit. The parity bit is adjusted to yield an odd number of *ones* in a character. In the computer, a character has no parity bit and is represented by six bits. FCU inserts a parity bit into each character of output data and checks and deletes it from each character of input data.

MTU's are designed to handle data consisting of Univac-coded (excess-three) characters; however, it is not necessary to use this type of coding if programming precautions are taken to guard against false recognition of *End-of-File* and *End-of-Data* sentinels.

A blockette transfer between computer and FCU consists of 24 thirty-bit words. Each word contains five characters which are defined as follows:

Character 0	Bits 0 through 5
Character 1	Bits 6 through 11
Character 2	Bits 12 through 17
Character 3	Bits 18 through 23
Character 4	Bits 24 through 29

The computer initiates a 24-word buffer (120 characters) for all data transfers to and from FCU. The following relationship exists between character positions in computer memory and line positions of the written blockette on magnetic tape. Word 0 is the first word of the computer buffer and word 23 is the last word (see Figure D10-5).

1st line of tape blockette = Word 23, Character 4
2nd line of tape blockette = Word 23, Character 3
3rd line of tape blockette = Word 23, Character 2
. . .
. . .
. . .
118th line of tape blockette = Word 0, Character 2
119th line of tape blockette = Word 0, Character 1
120th line of tape blockette = Word 0, Character 0

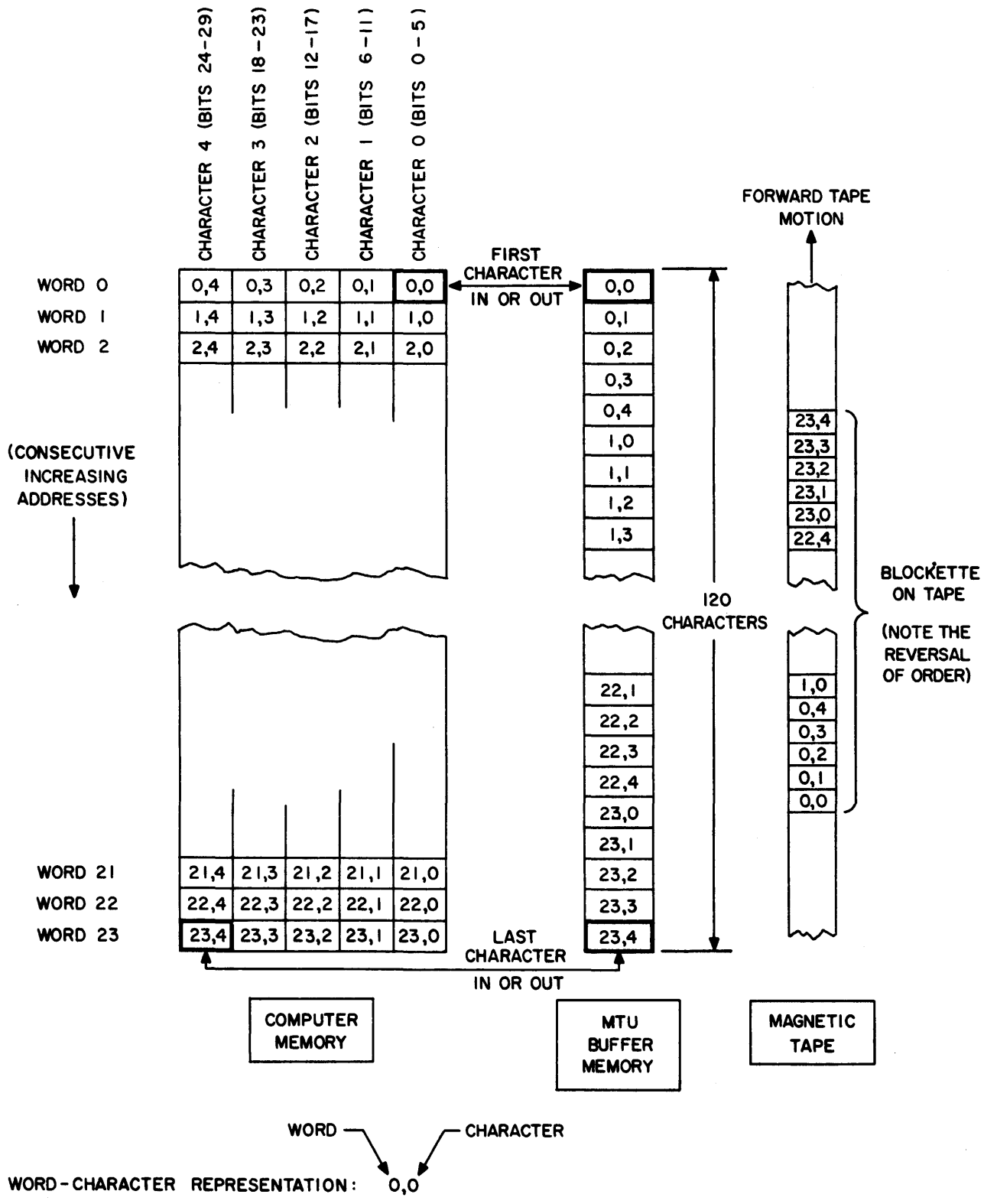


Figure D10-5. Corresponding Character Positions in Computer and MTU

B. *DATA TRANSFER - COMPUTER TO MTU*

After FCU accepts an External Function command from the computer and exchanges the necessary control and status signals with the desired MTU, it may then enable a data-transfer sequence if one has been commanded.

If the Transfer Data Out instruction (bit 4) is present, FCU enables this sequence immediately if the clock cycle is such that an RMA-6 pulse *will not* occur within the next 250 microseconds (approximately). If FCU determines that an RMA-6 *will* occur within this time interval, it will delay the enabling of the Transfer Data Out sequence until after the RMA-6. The purpose of this delay is to assure that the computer will have adequate time to transmit its first output word before data serialization must begin. It also permits the MTU to prepare for the start of data transfer. The following is a description of the Transfer Data Out sequence:

1) FCU transmits instruction to MTU (lines A, B, C, D).

2) FCU sets *Output Data Request* to computer.

3) Computer places first word on output channel with *Output Acknowledge.*

4) FCU accepts data word into C register and removes *Output Data Request.*

5) FCU transfers word from C to Z register and again sets *Output Data Request.* (Computer may send next word.)

6) FCU waits for RMA-6 pulse to synchronize start of data transfer. (First bit is sent on first TB 1 following RMA-6.)

7) After RMA-6 pulse, FCU shifts out characters 0, 1, 2, 3, 4 (in that order) in serial-bit form to the MTU. A parity bit is added to each character. Data are stored by the MTU in its 120-character buffer memory.

8) FCU transfers next computer word from C to Z and repeats the serializing procedure of step 7).

9) FCU accepts and serializes all 24 words in the same manner and data transfer terminates after the next RMA-6.

10) FCU sends interrupt to computer and then clears itself to *Available* upon receipt of an *Input Acknowledge.*

The MTU buffer memory accepts serial data at the rate of one bit each six microseconds. The character bits are transmitted to the MTU at the following times:

Bit 1 (low order bit)	TB 1
Bit 2	TB 2
Bit 3	TB 3
Bit 4	TB 4
Bit 5	TB 5
Bit 6	TB 6
Bit 7 (parity bit)	TB 7

There are two data lines from FCU to an MTU. A *one* is represented by a pulse on the *Data "1's"* line and a *zero* by a pulse on the *Data "0's"* line.

The computer has somewhat less than 210 microseconds to detect and respond to an *Output Data Request* signal. If the computer has not supplied the next data word before the time that FCU must begin serialization, FCU terminates the data transfer sequence, removes the *Output Data Request*, sets the *I/O Timing Error* and *Abnormal Condition* bits, and sends a status-report interrupt to the computer.

C. *DATA TRANSFER - MTU TO COMPUTER*

If the Transfer Data In instruction (bit 5) is present, FCU will enable this sequence immediately if the clock cycle is such that an RMA-6 pulse will not occur within the next 250 microseconds (approximately). If FCU determines that an RMA-6 pulse will occur within this time interval, it will delay the enabling of the Transfer Data In sequence until after the RMA-6 pulse. The purpose of this delay is to permit the MTU to prepare for the start of data transfer. The following is a description of the Transfer Data In sequence:

1) FCU transmits instruction to MTU.

2) FCU waits for RMA-6 pulse to synchronize start of data transfer. (MTU sends first bit on first TA 1 following RMA-6.)

3) FCU shifts in characters 0, 1, 2, 3, 4 (in that order) in serial-bit form from the MTU. The parity bit of each character is checked and deleted. The five characters are assembled into a 30-bit word in the Z register.

4) FCU transfers word from Z to C register, places data on input channel, and sets *Input Data Request.*

5) Shift-in and assembly of next word continues in Z.

6) Computer accepts word from C and sends *Input Acknowledge.*

7) FCU removes data and *Input Data Request* from channel.

8) After next word is assembled in Z, FCU transfers it to C and repeats steps 4), 5), 6), and 7).

9) FCU assembles and transfers all 24 words in the same manner and data transfer terminates after the next RMA-6 pulse.

10) FCU sends interrupt to computer and then clears itself to *Available* upon receipt of an *Input Acknowledge.*

The MTU transfers serial data to FCU at the rate of one bit each six microseconds. Character bits are sent by the MTU at the following times:

Bit 1	(low order bit)	TA 1
Bit 2		TA 2
Bit 3		TA 3
Bit 4		TA 4
Bit 5		TA 5
Bit 6		TA 6
Bit 7	(parity bit)	TA 7

There is only one data line from an MTU to FCU (the *Data "1's"* line). A pulse on this line at the required time represents a *one,* and the absence of a pulse at the required time represents a *zero.*

After FCU has filled the C register and set the *Input Data Request,* the computer has somewhat less than 210 microseconds to sample the data and respond with an *Input Acknowledge.* If the computer has not responded before FCU transfers the next word from Z to C, FCU terminates the data-transfer sequence, removes the *Input Data Request,* sets the *I/O Timing Error* and *Abnormal Condition* bits, and sends a status-report interrupt to the computer.

D. *DATA TRANSFER ERRORS*

1) Parity Error - FCU checks for odd parity on data characters sent by an MTU. If a parity error is detected during the Transfer Data In sequence, the *Parity Error* and *Abnormal Condition* bits are set. Then, after the Transfer Data In sequence is completed, FCU sends these bits in the status-report interrupt to the computer.

2) Input/Output Timing Error - As mentioned previously, FCU detects failure of the computer to accept an input data word or to supply an output data word within the imposed time limits. When this failure is detected, FCU terminates the data transfer sequence, removes the *Data Request*, sets the *I/O Timing Error* and *Abnormal Condition* bits, and sends a status-report interrupt to the computer.

7. FCU MAINTENANCE PANEL

The maintenance panel on FCU contains the following controls and indicators:

1) MANUAL CLEAR button - Clears FCU and all of the *on-line* tape units.

2) ON-LINE/OFF-LINE switches - One switch for each of the seven tape units. The OFF-LINE position disconnects the tape unit from computer control.

3) F REGISTER - Shows content of Function register.

4) S REGISTER - Shows content of Status register.

5) Z REGISTER - Shows content of Assembly/Disassembly register.

6) C REGISTER - Shows content of Communication register.

7) X REGISTER - Shows content of Character register.

PART TWO

PROGRAMMING SPECIFICATIONS

1. BASIC INFORMATION

Part Two of this section specifies ground rules and restrictions which must be observed when programming the Format Control Unit. Certain paragraphs in Part Two reference figures that appear in Part One.

2. MAGNETIC TAPE UNIT FEATURES

Each Type 4950 File Computer Tape Unit consists of a Tape Transport, a 120-character buffer memory, and associated control circuitry. It uses 1/2-inch Mylar-base magnetic tape. Data are recorded or read a line at a time with one 7-bit character stored on each line. In Read and Write operations, the MTU processes 120 consecutive lines (characters) before the tape stops. Basic format of the tape unit is a grouping of 120 characters, called a blockette.

Other MTU features and specifications include the following:

1) Tape Speed: 75 inches per second.

2) Recording Density: 139 lines per inch.

3) Blockette Length: 120 characters, 0.86 inch.

4) Interblockette Spacing:

   UFC Format - 0.5 inch.
   High-Speed Printer (Univac) Format - 1.0 inch with 2.4 inches after each sixth blockette.

5) Reading Density: MTU will read information recorded at a density of from 50 to 160 lines per inch.

6) Bad-Spot Detection: Photoelectric cells, transparent markers.

7) Number of Blockettes per Reel: Maximum 20,300 blockettes on 2400-foot reel with 0.5-blockette spacing. Automatic counter limit is 20,000 blockettes.

8) Automatic Terminal Condition Detection: MTU can detect the following logical conditions during its operation:

   a) Beginning and End of Tape (Photoelectric detection of transparent tape)

   b) End of File and End of Data (detection of sentinel written on the tape)

   c) End-of-Blockette Count (automatic blockette counter compared against preset limit)

   d) Tape Search Find (compares tape blockettes with identifier blockette for equality or inequality)

3. CONTROL PROGRAMMING PROCEDURES

Control messages exchanged by the computer and FCU consist of an External Function command sent from the computer to FCU and a status-report interrupt sent from FCU to the computer. In normal tape system operation there is a one-to-one correspondence between these two control messages. That is, for every External Function command sent by the computer, FCU responds with a status-report interrupt. This is the basic rule of control communication: *The computer program shall not issue a new External Function command until the FCU has responded with a status-report interrupt to the previous External Function command.*

A. *EXTERNAL FUNCTION COMMAND*

The computer program initiates tape system operation by sending an External Function command on the computer output channel to FCU. As shown in Figure D10-3, the External Function word is comprised of three parts: MTU address, MTU instruction, and FCU instruction.

1) *MTU Address* (Bits 15 through 21) - In each External Function command sent to FCU, ONE MTU address bit must be set to select the proper tape unit. It is illegal to set more than or fewer than one address bit. This would cause an abnormal condition (NO MTU Response) or other improper operation.

2) *MTU Instruction* (Bits 11 through 14) - The MTU instruction selects which one of 13 tape system operations is to be performed. See Figure D10-3 for instruction-code assignment.

3) *FCU Instruction* (Bits 4, 5, 7) - The Transfer Data Out instruction (bit 4) enables FCU to transfer data from the computer to the MTU. This bit *must* be set for each MTU instruction which requires data from the computer (Write, all Searches, and Transfer Blockette, Computer to MTU).

The Transfer Data In instruction (bit 5) enables FCU to transfer data from the MTU to the computer. This bit *must* be set for the MTU instruction, Transfer Blockette, MTU to Computer.

The Special Status Lockout (bit 7) may be sent in any External Function command. This bit causes FCU to ignore any Special Status conditions which may be present in the MTU.

When FCU accepts an External Function command, it becomes *Not Available* and locks out any other External Function commands until it completes its internal sequence, sends a status-report interrupt, and becomes *Available* again.

After the computer program issues an External Function command to FCU, it may *wait* for the status-report interrupt or it may perform useful activities until it is interrupted.

B. *STATUS-REPORT INTERRUPT*

The status-report interrupt supplies the computer program with the following information:

1) Every interrupt signifies that FCU is available to process a new External Function command

2) It indicates the absence or presence of abnormal conditions in the tape system

3) It indicates whether the tape units are *Ready* or *Not Ready*

4) It indicates the absence or presence of Terminal conditions in the addressed tape unit.

An external interrupt from FCU causes the computer to execute an interrupt interpretation routine. This routine must store the input channel to acquire the status-report word. (Execution of the 17 instruction, *Store Input Channel*, also signals FCU to remove the interrupt.)* The status-report word is then analyzed in the following manner. (See Figure D10-4 for Status Report format.)

*The interrupt interpretation routine must also execute a 600XXXXXXX or 601XXXXXXX instruction in order to release the interrupt lockout within the computer.

1) If bit 0 is set, an abnormal condition is present and bits 3, 4, 5, and 6 must be checked to determine the nature of the condition.

   a) If bit 3 is set, an *MTU Error* condition is present in the addressed MTU. This MTU cannot be used again until the error condition is *manually* cleared. Other tape units, however, may still be referenced.

   b) If bit 4 is set, the addressed MTU has failed to respond to interrogation by FCU. The *No MTU Response* condition can result from having no address bit set in the External Function command, having more than one address bit set, or failure to switch the MTU *on-line*.

   c) If bit 5 is set, FCU has detected a *Parity Error* during the Transfer In operation. This signifies to the program that the blockette of data which has just been buffered-in is incorrect. It is possible for the program to recover in this situation by re-performing the previous Transfer In operation.

   d) If bit 6 is set, FCU has detected a failure of the computer input/output to meet the timing requirement imposed by FCU. An *Input/Output Timing Error* can be caused by the computer initiating an input or output buffer which is shorter than 24 words or by a heavy load of input/output on higher priority computer channels. This indicates to the program that the blockette of data which has just been transferred in or out is incorrect. It is possible for the program to recover by re-performing the previous operation; however, if this operation has resulted in tape movement, it will be necessary to re-position the tape first.

2) If bit 1 is set, the addressed tape unit is *Not Ready* and the command has not been performed. In this case, it is often convenient to send the External Function command again and remain in this cycle until the tape unit becomes *Ready* and performs the operation.

3) If bit 2 is set, a terminal condition is present in the addressed tape unit and Special Status bits 11, 12, 13, 14 must be checked to determine the nature of the condition. The type of terminal condition reported on each of the W, X, Y, Z lines (Special Status bits 14, 13, 12, and 11, respectively) is the programmer's option. The following terminal conditions are available at the MTU:

   End of File (EF)
   End of Data (ED)
   End-of-Blockette Count (EC)

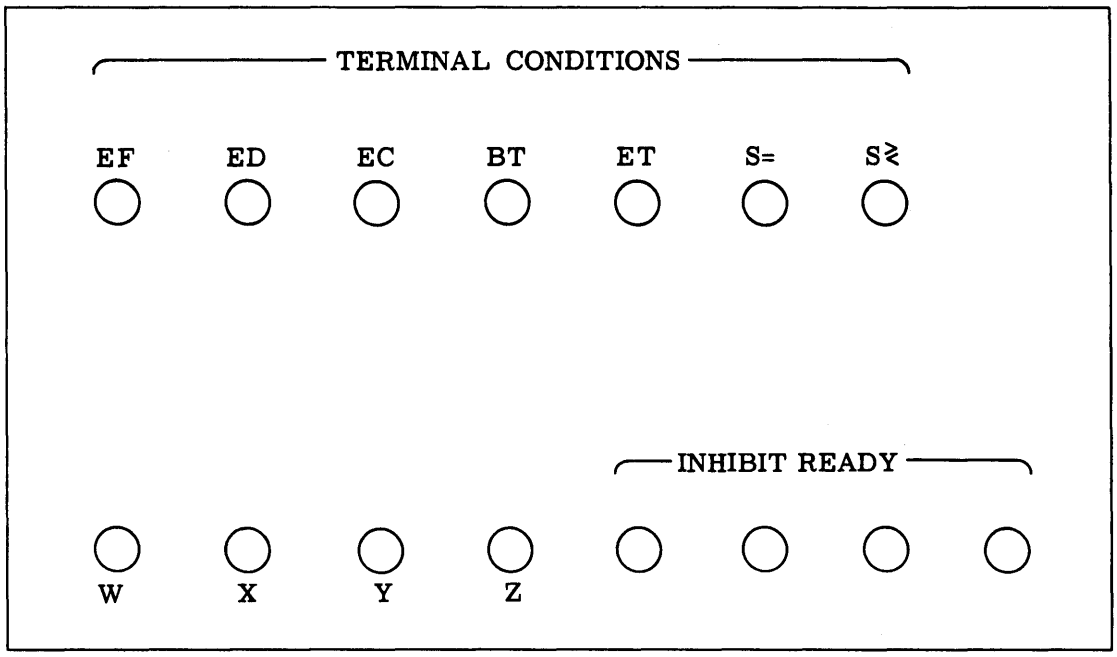
Beginning of Tape (BT)

End of Tape (ET)

Search Equal Find (S=)

Search Unequal Find (S≷)

Any four of these conditions may be plugged into the W, X, Y, Z lines at the pinboard on the MTU maintenance panel (see Figure D10-6). (If desired, any remaining terminal condition may be plugged into an INHIBIT READY receptacle on the MTU panel. If this is done, the presence of the terminal condition will prevent the MTU from becoming *Ready*.)



EF	END OF FILE
ED	END OF DATA
EC	END-OF-BLOCKETTE COUNT
BT	BEGINNING OF TAPE
ET	END OF TAPE
S=	SEARCH EQUAL FIND
S≷	SEARCH UNEQUAL FIND

Figure D10-6. Pinboard on MTU Panel

In order for a terminal condition to be reported in the status-report interrupt, it must have been present in the MTU *before* the External Function command was issued by the computer. Furthermore, the commanded operation will not be performed by the tape system in this event. After the program has evaluated the Special Status and desires to send another command to this MTU, it must set the Special Status Lockout (bit 7) in this External Function command. This prevents FCU from examining the W, X, Y, Z lines, and the operation proceeds as if the terminal condition were not present. Special Status is not included in the status-report interrupt which follows a command in which the Lockout is set. Note, however, that if the command does not move the tape (i.e., a Transfer operation), the terminal condition will remain present in the MTU.

4) If neither bit 0, or bit 1, nor bit 2 is set, it signifies that no abnormal condition, *Not Ready* condition, or Special Status is present and the operation has been (or is being) performed in the normal manner. This indicates that FCU has successfully completed its portion of the operation and is available for another command. In most cases, the MTU is still performing the operation at this time.

4. DATA

A. *DATA FORMAT*

In those tape system operations which require data transfer, one blockette of data (24 computer words) is sent from the computer to the MTU buffer memory or vice-versa. On the magnetic tape itself, each blockette of data appears as 120 characters with every character written on a separate line (frame) on the tape. The relationship between the data characters as they are stored in the computer memory and on the tape is diagrammed in Figure D10-5. Note that the first (most significant) character of the tape blockette is actually found in the last word of the computer buffer list and the last (least significant) character of the tape blockette is found in the first word of the computer buffer list. This reversal-of-order of the words in computer memory is dictated by the characteristics of the MTU. It is important to recognize that this relationship never varies, regardless of the direction of tape motion. Thus, whether a blockette is read forward or backward on the MTU, it will appear the same in the computer memory.

Either Univac-coded (excess-three) or uncoded binary (bioctal) data may be used.

B. *END-OF-FILE AND END-OF-DATA SENTINELS*

A file is a programming unit used for the over-all organization of data. It may consist of any number of blockettes. The tape unit detects the *End of File* condition by examining every blockette for the presence of a sentinel which is written on the tape along with the data. The *End-of-File* sentinel consists of at least twelve consecutive "74" characters (111100) in a blockette. When the *End-of-File* condition is detected, it is available to FCU via one of the W, X, Y, Z lines (if plugged at the MTU panel) and will appear in the status report.

The *End of Data* sentinel is detected and reported in exactly the same manner. It consists of at least twelve consecutive "75" characters (111101) in a blockette.

If bioctal coding is used on the tape, it is possible for these sentinels to occur inadvertently in a combination of data characters and, hence, be falsely detected by the tape unit. The following procedure is recommended to minimize the possibility of false detection:

1) Write each *End-of-File* and *End-of-Data* sentinel as an *entire* blockette of sentinel characters.

2) Whenever *End-of-File* or *End-of-Data* is reported, read in the last blockette and verify that it consists entirely of sentinel characters. If not, regard it as a false detection.

5. DESCRIPTION OF TAPE SYSTEM OPERATIONS

This subsection contains a description of the following tape system operations. (The description is limited to those features of the operation which are of interest to the programmer, see Figure D10-7.)

- Write
- Read Forward
- Read Backward
- Search Forward Equal
- Search Backward Equal
- Search Forward Equal or Greater Than
- Search Backward Equal or Less Than
- Transfer Blockette, MTU to Computer
- Transfer Blockette, Computer to MTU

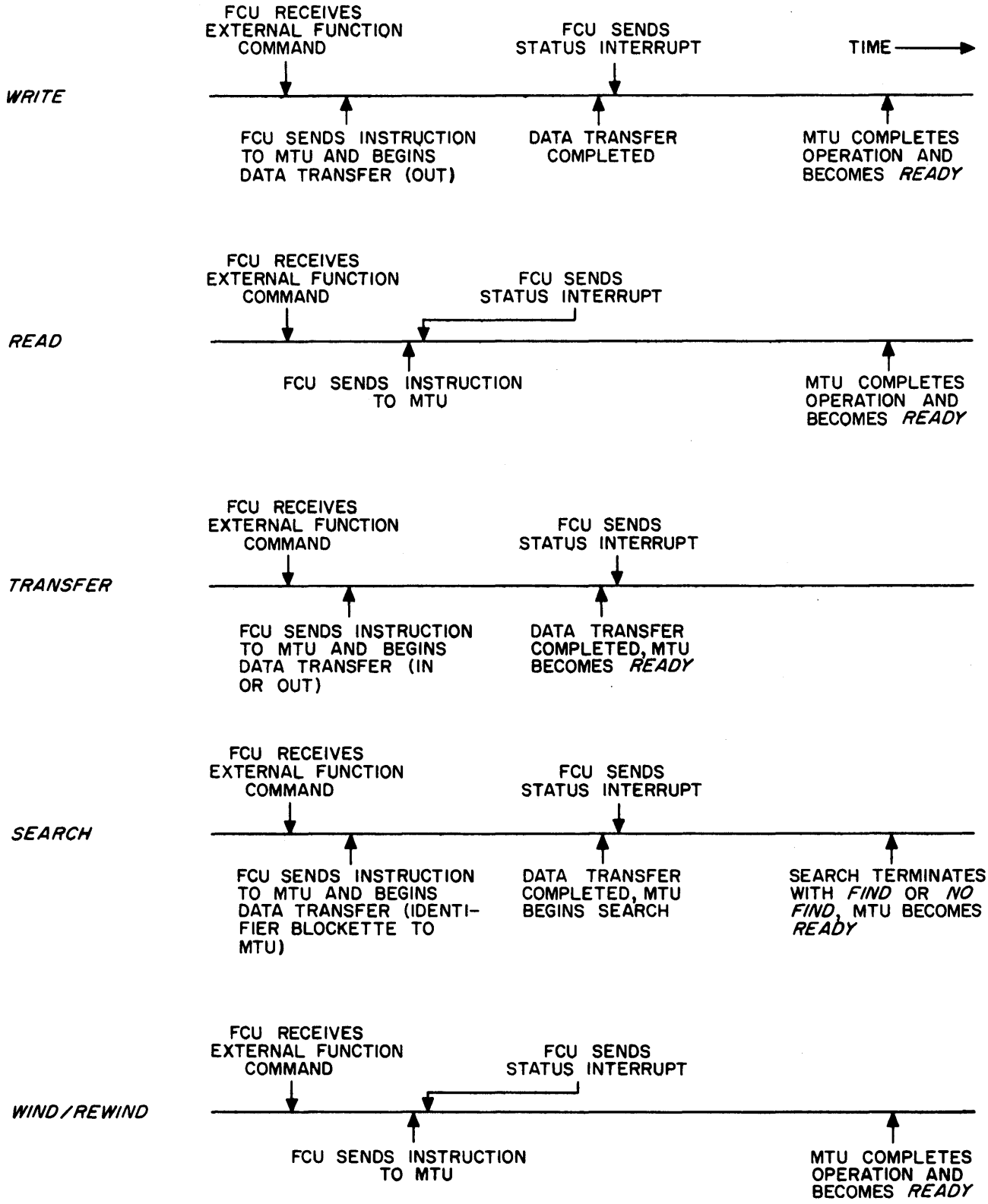


Figure D10-7. Tape System Operations

- Wind Forward
- Wind Forward with Interlock
- Rewind
- Rewind with Interlock

A. *WRITE*

In the Write operation, one blockette of data is transferred from the computer, through FCU, to the MTU buffer memory. Upon completion of the blockette transfer, FCU sends an interrupt and becomes *Available*, but the MTU does not become *Ready* until it has recorded the contents of its buffer memory on magnetic tape. Tape moves in the forward direction.

A Write instruction to the MTU and a Transfer Data Out instruction to FCU comprise the External Function command which initiates this operation.

It is not possible to write a blockette in the middle of a previously written tape without destroying information in the blockettes that follow; writing must be done sequentially from beginning of tape. A Write operation should not be attempted if the MTU is positioned at *End of Tape* since this will cause the MTU to "hang up" and remain *Not Ready*.

The following procedure is recommended for a Write operation:

1) FCU is known to be *Available*
2) Initiate 24-word output buffer (without monitor, 74 instruction)
3) Send External Function command (13 instruction)
4) Do not attempt any other tape system operation until interrupt is received from FCU
5) When status-report interrupt occurs, store the report (17 instruction) and analyze it as described in Part Two, Subsection 3.B.

B. *READ FORWARD*

In the Read Forward operation, the "next" blockette of data is read from tape (tape moves in forward direction) and stored in the MTU buffer memory. MTU does not transfer this blockette to the computer; however, a subsequent Transfer Blockette, MTU-to-Computer operation is required to obtain the data from the MTU buffer memory. FCU becomes *Available* shortly after the command is sent since no data transfer is required, but the MTU does not become *Ready* until its internal *Read* sequence is completed.

A Read Forward instruction to the MTU comprises the External Function command which initiates this operation. No data transfer instruction is sent to FCU.

A Read Forward operation should not be attempted if the MTU is positioned at *End-of-Data* or *End of Tape*. The former may cause an MTU Error (Tape 120 Error or Tape Parity Error) and the latter will cause the MTU to "hang up" and remain *Not Ready*.

The following procedure is recommended for a Read Forward operation:

1) FCU is known to be *Available*
2) Send External Function command (13 instruction)
3) Do not attempt any other tape system operations until interrupt is received from FCU
4) When status-report interrupt occurs, store the report (17 instruction) and analyze it as described in Part Two, Subsection 3.B
5) After Read Forward has been accomplished, the computer must command a Transfer Blockette, MTU-to-Computer operation (see Part Two, Subsection 5.H) to obtain the data.

During each Read Forward operation, the MTU scrutinizes the blockette for the presence of *End-of-File* or *End of Data* sentinels written with the data. If one of these conditions is detected, it will be included in the status report the *next* time the computer addresses a command to this MTU.

C. *READ BACKWARD*

The Read Backward operation is similar to the Read Forward operation in most respects; however, the tape is initially positioned at the end of the blockette. Tape movement is in the reverse direction and after the *Read* is completed, the tape is positioned at the beginning of the blockette.

An External Function command comprised of a Read Backward instruction to the MTU initiates this operation. No data transfer instruction is sent to FCU.

A Read Backward operation should not be attempted if the MTU is positioned at *Beginning of Tape* since this will cause the MTU to "hang up" and remain *Not Ready*.

The following procedure is recommended for a Read Backward operation:

1) FCU is known to be *Available*

2) Send External Function command (13 instruction)

3) Do not attempt any other tape system operation until interrupt is received from FCU

4) When status-report interrupt occurs, store the report (17 instruction) and analyze it as described in Part Two, Subsection 3.B

5) After Read Backward has been accomplished, the computer must command a Transfer Blockette, MTU-to-Computer operation (see Part Two, Subsection 5.H) to obtain the data.

It is important to note that the Read Backward operation does not alter the order of the data received by the computer. The blockette is always presented to the computer in the same format, whether it is read forward or backward.

The automatic sentinel detection feature is the same as in the Read Forward operation.

D. *SEARCH FORWARD EQUAL*

Search operations require that the computer prepare an identifier blockette which is transferred to the MTU during the operation and compared against the blockettes written on tape.

In the Search Forward Equal operation, the identifier blockette from the computer is first loaded into the MTU buffer memory. After this has been accomplished, FCU reverts to the *Available* state and MTU is released to perform the following sequence:

1) The magnetic tape begins moving in the forward direction, and each tape blockette as it passes under the read head is compared character-by-character against the identifier stored in the MTU buffer memory.

2) When the comparator finds 120 characters within a tape blockette equal to the 120 characters stored in the MTU buffer, the Search Equal Find signal is energized and will be available to the computer, through FCU, if this condition has been plugged into one of the W, X, Y, Z Special Status lines at the MTU panel.

3) When a *find* occurs, as described above, the tape blockette is read into the MTU buffer memory. It is not, however, transferred to the computer. A subsequent Transfer Blockette, MTU-to-Computer operation is required for the computer to obtain the data from the MTU buffer memory.

4) If *End of File*, *End of Data*, or *End of Tape* is detected before a *find* occurs, the search will end. Any one of these conditions is a signal to the program that *no find* has occurred.

5) After a find, the tape remains positioned immediately after the blockette.

It is possible to include ignore codes ("00" character) in the identifier blockette. An ignore code suppresses comparison for that character; in effect, any tape character will be found equal to an ignore code. This ability of the ignore code to suppress comparison allows the identifying field in the blockette to be from 1 to 120 characters in length and in any position(s) in the blockette.

An External Function command comprised of a Search Forward Equal instruction to the MTU and a Transfer Data Out instruction to FCU initiates this operation. Search Forward Equal should not be attempted if the MTU is positioned at *End of Data* or *End of Tape*.

The following procedure is recommended for a Search Forward Equal operation:

1) FCU is known to be *Available*.

2) Initiate 24-word output buffer (74 instruction) for the identifier blockette.

3) Send External Function command (13 instruction).

4) Do not attempt any other tape system operation until interrupt is received from FCU.

5) When status-report interrupt occurs, store the report (17 instruction) and analyze it as described in Part Two, Subsection 3.B. (Note: This status report will not include indication of a *find* or *no find* condition. This information will be available only after the search has terminated and the MTU becomes *Ready* again.)

6) It is convenient to repeatedly send the next External Function command until the MTU becomes *Ready*. Then, if a *find* is reported, a Transfer Blockette, MTU-to-Computer operation (see Part Two, Subsection 5.H) may be performed to obtain the data.

E. *SEARCH BACKWARD EQUAL*

Sequence of events in a Search Backward Equal operation is the same as in Search Forward Equal, except that the direction of tape movement is reversed. *Beginning of Tape* replaces *End of Tape* as one of the *no find* conditions.

An External Function command comprised of a Search Backward Equal instruction to the MTU

and a Transfer Data Out instruction to FCU initiates this operation. Search Backward Equal should not be attempted if the MTU is positioned at *Beginning of Tape*.

F. *SEARCH FORWARD EQUAL OR GREATER THAN*

In the Search Forward Equal or Greater Than operation, sequence of events is similar to Search Forward Equal except that the comparator looks for a blockette on the tape which is equal to or greater than the identifier blockette. In addition, the first character of the tape blockette is handled differently than other characters. This position (Character 4 of Word 23 in the computer buffer) contains the *Search Validation Character* (SVC). The corresponding character from the tape (line 1) must be *equal* to the SVC in the identifier in order for a *find* to occur. If no comparison of the SVC is desired, an ignore code (00) should be placed in this position of the identifier.

In the Search Forward Equal or Greater Than operation, after a *find*, the tape is re-positioned to the *beginning* of the blockette just found. If a *find* occurs, the *Search Unequal Find* or the Search Equal Find signal is activated.

A Search Forward Equal or Greater Than instruction to the MTU and a Transfer Data Out instruction to FCU comprise the External Function command which initiates this operation. It should not be attempted if the MTU is positioned at *End of Tape*.

G. *SEARCH BACKWARD EQUAL OR LESS THAN*

In the Search Backward Equal or Less Than operation, the sequence of events is similar to that of Search Forward Equal or Greater Than except that the direction of tape movement is reversed and the comparator looks for a blockette on the tape equal to or less than the value of the identifier in the buffer. The Search Validation Character performs the same function in the Search Backward Equal or Less Than as it does in the Search Forward Equal or Greater Than.

A Search Backward Equal or Less Than instruction to the MTU and a Transfer Data Out instruction to FCU comprise the External Function command which initiates this operation. It should not be attempted if the MTU is positioned at *Beginning of Tape*.

H. *TRANSFER BLOCKETTE, MTU TO COMPUTER*

In the Transfer Blockette, MTU-to-Computer operation, the information in the MTU buffer memory is transmitted, through FCU, to the computer. No tape movement is involved in this

operation. FCU becomes *Available* and the MTU becomes *Ready* as soon as the blockette transfer has been completed. Transfer Blockette, MTU to Computer is normally used to obtain the data subsequent to the completion of a Read or a Search operation.

A Transfer Blockette, MTU-to-Computer instruction to the MTU and a Transfer Data In instruction to FCU comprise the External Function command which initiates this operation. The following procedure is recommended for a Transfer Blockette, MTU-to-Computer operation:

1) FCU is known to be *Available*
2) Initiate 24-word input buffer (without monitor, 73 instruction)
3) Send External Function command (13 instruction)
4) Do not attempt any other tape system operation until interrupt is received from FCU
5) When status-report interrupt occurs, store the report (17 instruction) and analyze it as described in Part Two, Subsection 3.B.

I. *TRANSFER BLOCKETTE, COMPUTER TO MTU*

In the Transfer Blockette, Computer-to-MTU operation, the information in the computer output buffer list is transmitted, through FCU, to the MTU buffer memory. No tape movement is involved in this operation. FCU becomes *Available* and the MTU becomes *Ready* as soon as the blockette transfer has been completed. This operation has limited utility except in the area of maintenance programming.

A Transfer Blockette, Computer-to-MTU instruction to the MTU and a Transfer Data Out instruction to FCU comprise the External Function command which initiates this operation. The following procedure is recommended for a Transfer Blockette, MTU-to-Computer operation:

1) FCU is known to be *Available*
2) Initiate 24-word output buffer (74 instruction)
3) Send External Function command (13 instruction)
4) Do not attempt any other tape system operation until interrupt is received from FCU
5) When status-report interrupt occurs, store report (17 instruction) and analyze it as described in Part Two, Subsection 3.B.

J. *WIND FORWARD*

In the Wind Forward operation, tape movement begins in the forward direction and continues until *End of Tape* is detected. FCU becomes *Available* immediately after it relays the instruction to the MTU, but the MTU does not become *Ready* until the tape movement has stopped.

A Wind Forward instruction to the MTU comprises the External Function command which initiates this operation. No data-transfer instruction is sent to FCU. Wind Forward should not be attempted if the MTU is already at *End of Tape* since this will cause the MTU to remain *Not Ready*.

The following procedure is recommended for a Wind Forward operation:

1) FCU is known to be *Available*
2) Send External Function command (13 instruction)
3) Do not attempt any other tape system operation until interrupt is received from FCU
4) When status-report interrupt occurs, store report (17 instruction) and analyze it as described in Part Two, Subsection 3.B.

K. *WIND FORWARD WITH INTERLOCK*

This operation is the same as Wind Forward except that, when the tape has stopped, an interlock is set which prevents further use of the MTU until it is cleared manually at the MTU panel.

L. *REWIND*

In the Rewind operation, tape movement begins in the backward direction and continues until *Beginning of Tape* is detected. FCU becomes *Available* immediately after it relays the instruction to the MTU, but the MTU does not become *Ready* until the tape movement has stopped. During a Rewind, each blockette of data is read to check the parity and character count. If an error is detected, the MTU ERROR light is turned on, operation is suspended, and the MTU remains *Not Ready*.

A Rewind instruction to the MTU comprises the External Function command which initiates this operation. No data-transfer instruction is sent to FCU. Rewind should not be attempted if the MTU is already at *Beginning of Tape* since this will cause the MTU to remain *Not Ready*.

The following procedure is recommended for a Rewind operation:

1) FCU is known to be *Available*

2) Send External Function command (13 instruction)

3) Do not attempt any other tape system operation until interrupt is received from FCU

4) When status-report interrupt occurs, store report (17 instruction) and analyze it as described in Part Two, Subsection 3.B.

M. *REWIND WITH INTERLOCK*

This operation is the same as Rewind except that, when the tape has stopped, an interlock is set which prevents further use of the MTU until it is cleared manually at the MTU panel.

N. *DUMMY COMMAND*

It is possible to send a dummy (*do-nothing*) command addressed to a specific MTU. This is accomplished by an External Function command which contains only the desired MTU address bit, with *zeros* in the remainder of the word. This command will result in a status-report interrupt in the normal manner, but the MTU will perform no operation. The dummy command is useful in that it permits the computer to determine whether or not a terminal condition is present in a particular MTU without actually performing an operation if no terminal condition is present.

6. OPERATION TIMES

The following approximated average operation times are given as an aid to efficient programming: (All times are in milliseconds.)

Write one blockette	35
Read (and Transfer In) one blockette	40
Transfer one blockette (in or out)	8

During Search, Wind, and Rewind operations, the tape moves at a speed of 75 inches per second.

The time during which FCU is not *Available* (from receipt of External Function command to acknowledgement of status-report interrupt) depends upon whether or not a data-transfer sequence is performed. If no data transfer takes place, this interval is less than one millisecond. If data transfer does take place, the interval can vary between five and ten milliseconds (approximately).

7. TAPE REVERSAL DELAY

The computer program must include a 500-millisecond delay between successive operations which will result in a reversal of tape direction (i.e., Search Forward followed by Read Backward). This delay is measured from the time that the MTU becomes *Ready* after the first operation until the time the computer issues the second command.

If this delay is not included, there is danger of improper operation of the tape handler servo. If the MTU has been modified to eliminate this problem, the programmed delay is not required.

SECTION D11
FUNCTIONAL SPECIFICATIONS
FOR
PERIPHERAL EQUIPMENT DUPLEX OPERATION

CONTENTS

ILLUSTRATIONS

FUNCTIONAL SPECIFICATIONS
FOR
PERIPHERAL EQUIPMENT DUPLEX OPERATION

1. BASIC INFORMATION

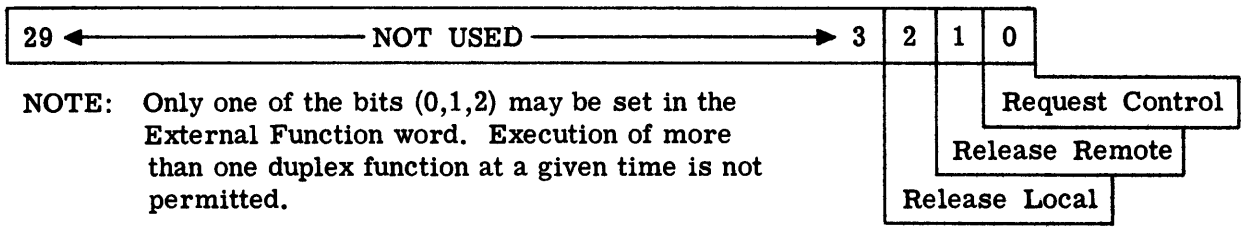
By utilizing computer duplex operation, a peripheral equipment can be physically connected to two computers and can operate with either computer as directed by a series of duplexing control signals. These duplexing control signals permit only one of the two computers to have control over the peripheral equipment at any given time.

Peripheral equipment employing computer duplexing contains logic circuitry which enables it to interpret duplexing control signals. Duplexing control codes are contained in the lower three bits of an External Function word: Bit 0, *Request Control;* Bit 1, *Release Remote;* and Bit 2, *Release Local.* These codes, in conjunction with the *Control Acknowledge* Interrupt which is generated by the peripheral equipment, are the only codes required for performance of the computer duplexing function. Computer programs, by enabling the transmission of External Function codes and by evaluating external interrupts, determine which of the computers' control and data lines are fully enabled for communication with the peripheral equipment. External Function lines and the lower three bits of the output data channels of both computers are always enabled in normal operation. The method used for performing computer duplexing allows a peripheral equipment to be in one of three conditions at any given time. By designating the two computers in a complex as *originating* (or *local*) computer and *other* (or *remote*) computer, the peripheral equipment can be described by one of the three following conditions:

1) Under control of *originating* computer
2) Under control of *other* computer
3) Under control of *neither* computer.

2. EXTERNAL FUNCTION CODES FOR DUPLEXING

Duplex circuitry of a peripheral equipment interprets three of the External Function commands which are sent to it. These three External Function codes use bit-position coding as shown in Figure D11-1. The following descriptions illustrate the actions that take place when the "duplex control" External Function codes are received by a peripheral equipment. All definitions of *input* and *output* lines are with respect to the computer. (*Input* means input *to* the computer, and *output* means output *from* the computer.)



NOTE: Only one of the bits (0,1,2) may be set in the External Function word. Execution of more than one duplex function at a given time is not permitted.

Figure D11-1. External Function Word Used for Duplex Operation

A. *REQUEST CONTROL* ($XXXXXXXXX1_8$)

The *Request Control* External Function code is sent by a computer at the time it desires control of the peripheral equipment. One of the following situations will exist at this time:

1) *Originating* computer in control - The *Request Control* External Function code is ignored; therefore, *originating* computer retains control and normal data communications may continue.

2) *Other* computer in control - Duplex circuitry of the peripheral equipment stores the *Request Control* External Function code until *other* computer releases control and the equipment reaches a logical termination point. When *other* computer does release control and the equipment reaches a logical termination point, a *Control Acknowledge* Interrupt is sent to *originating* computer. Normal data communications may be established as soon as the *Control Acknowledge* Interrupt is received and interpreted by *originating* computer.

3) Neither computer in control - The peripheral equipment responds with a *Control Acknowledge* Interrupt; *originating* computer may then establish normal data communications. If the peripheral equipment receives *Request Control* codes from both computers simultaneously when neither computer is in control, only one will receive control; the second computer's *Request Control* code will be stored.

When the *Request Control* External Function command is actually executed in the duplex circuitry of the peripheral equipment, the following are accomplished:

1) The following lines between *originating* computer and the peripheral equipment are enabled:

   a) Input and Output Data Request lines
   b) Interrupt line
   c) All 30 Output Data lines
   d) Input and Output Acknowledge lines.

2) A *Control Acknowledge* Interrupt is transmitted to *originating* computer

3) The following lines between *other* computer and the peripheral equipment are prevented from being enabled:

   a) Input and Output Data Request lines
   b) Interrupt line
   c) Upper 27 Output Data lines
   d) Input and Output Acknowledge lines.

4) When items 1) through 3) have been accomplished, storage of the *Request Control* code is terminated.

B. *RELEASE LOCAL* ($XXXXXXXXX4_8$)

The *Release Local* External Function code is sent by a computer when the control of the peripheral equipment used is to be released. The *Release Local* External Function code is not stored in the peripheral equipment but is performed upon receipt by the peripheral equipment.

One of the following situations will exist at the time the *Release Local* External Function code is sent by the computer:

1) *Originating* computer in control - If the peripheral equipment has completed performance of the last External Function code sent, it reverts to the neutral state and is *not* controlled by either computer. If the peripheral equipment is still performing the last External Function command, the *Release Local* External Function code is ignored.

2) *Other* computer in control - The peripheral equipment ignores the *Release Local* code.

3) Neither computer in control - The peripheral equipment ignores the *Release Local* External Function code.

When executed, the *Release Local* External Function code accomplishes the following in the peripheral equipment:

1) The following lines between *originating* computer and the peripheral equipment are disabled:

   a) Input and Output Data Request Lines
   b) Interrupt line
   c) Upper 27 Output Data lines
   d) Input and Output Acknowledge lines.

2) It provides an enable that allows the *Request Control* signal from the *other* computer to enable the following lines between *other* computer and the peripheral equipment:

   a) Input and Output Data Request lines
   b) Interrupt line
   c) All 30 Output Data lines
   d) Input and Output Acknowledge lines.

C. *RELEASE REMOTE* ($XXXXXXXXX2_8$)

This External Function code is sent by a computer to free the peripheral equipment from control of *other* computer. The *Release Remote* External Function command is performed upon receipt by the peripheral equipment and is not stored in the peripheral equipment.

The *Release Remote* External Function code is *not* the normal method of placing a peripheral equipment in a neutral state. When transmitted by the noncontrolling computer, it is immediately acted upon and could possibly leave the computer that formerly controlled the peripheral equipment with active buffers but not effectively connected to the equipment. The *Release Remote* External Function code, therefore, has severe restrictions placed on its use. Its use must be restricted to the unusual situation where the controlling computer is malfunctioning and is incapable of releasing control of the peripheral equipment. The use of *Release Remote* will be defined by the System Executive Control Program.

One of the following situations will exist at the time the *Release Remote* External Function command is sent by the computer:

1) *Originating* computer in control - The peripheral equipment ignores the *Release Remote* External Function code.

2) *Other* computer in control - The peripheral equipment ceases all communication with *other* computer immediately upon receipt of the *Release Remote* code, and the peripheral equipment is placed in a neutral state. *Originating* computer must have sent a *Request Control* code previous to the *Release Remote* code and must receive the Control Acknowledge Interrupt in order to gain control.

3) Neither computer in control - The peripheral equipment ignores the *Release Remote* External Function code.

The *Release Remote* External Function code accomplishes the following in the peripheral equipment:

1) The following lines between *other* computer and the peripheral equipment are disabled:

   a) Input and Output Data Request lines
   b) Interrupt line
   c) Upper 27 Output Data lines
   d) Input and Output Acknowledge lines.

2) It provides an enable that allows the *Request Control* signal from *originating* computer to enable the following lines between *originating* computer and the peripheral equipment:

   a) Input and Output Data Request lines
   b) Interrupt line
   c) All 30 Output Data lines
   d) Input and Output Acknowledge lines.

3. CONTROL ACKNOWLEDGE INTERRUPT

The *Control Acknowledge* Interrupt is sent to a computer which has requested control of a duplexed peripheral equipment to indicate that the requesting computer is in control. The interrupt is not sent until the equipment has reached a logical termination point and *other* computer

has released control as a direct result of a *Release Local* or *Release Remote.*

The *Control Acknowledge* code which is sent with the interrupt consists of the 30-bit data word with one bit set to *one.** After receiving the interrupt, the computer must acknowledge having received control by executing a *STORE* C^n instruction which generates an Input Acknowledge to the peripheral equipment. This Input Acknowledge clears the peripheral equipment interrupt line.

Receipt of a *Control Acknowledge* Interrupt by a computer does not by itself establish communications with a peripheral equipment. It does, however, acknowledge functional control of the equipment by the computer. The computer must transmit additional External Function codes as required by the particular equipment in order to establish data communications.

The duplexing logic does not include provision for informing the computers of the transmission or performance of the *Release Local* and *Release Remote* External Function codes.

4. NONDUPLEXING EXTERNAL FUNCTION CODES ($XXXXXXXXX0_8$)

Nonduplexing External Function codes are those required by the peripheral equipment to establish data communications with a computer after the computer has gained control of the peripheral equipment. These External Function codes are not recognized by the peripheral equipment unless *originating* computer has first obtained control of the equipment by following the correct sequence of duplexing External Function codes and *Control Acknowledge* code evaluation.

Nonduplexing External Function codes sent to a peripheral equipment containing the duplex feature must always contain *zeros* in bit-positions 0, 1, and 2. If any of these lower three bits are set to *one,* the External Function word is interpreted as a "duplex-control" word and the upper 27 bits are completely ignored.

The *Master Clear* is not a duplexing External Function code. A programmed *Master Clear* does not affect the duplexing status of peripheral equipment. The MASTER CLEAR switch on the peripheral-equipment cabinet does clear the duplex circuitry, however.

5. PROGRAMMING RULES

1) Only the following duplexing External Function codes will be used:

---

* Which bit is used depends on the particular peripheral equipment.

Request Control	$(XXXXXXXXX1_8)$
Release Remote	$(XXXXXXXXX2_8)$
Release Local	$(XXXXXXXXX4_8)$

2) The Interrupt-Evaluation routine for the *Control Acknowledge* code will include a *STORE* C^n instruction to transmit an Input Acknowledge signal to the peripheral equipment.

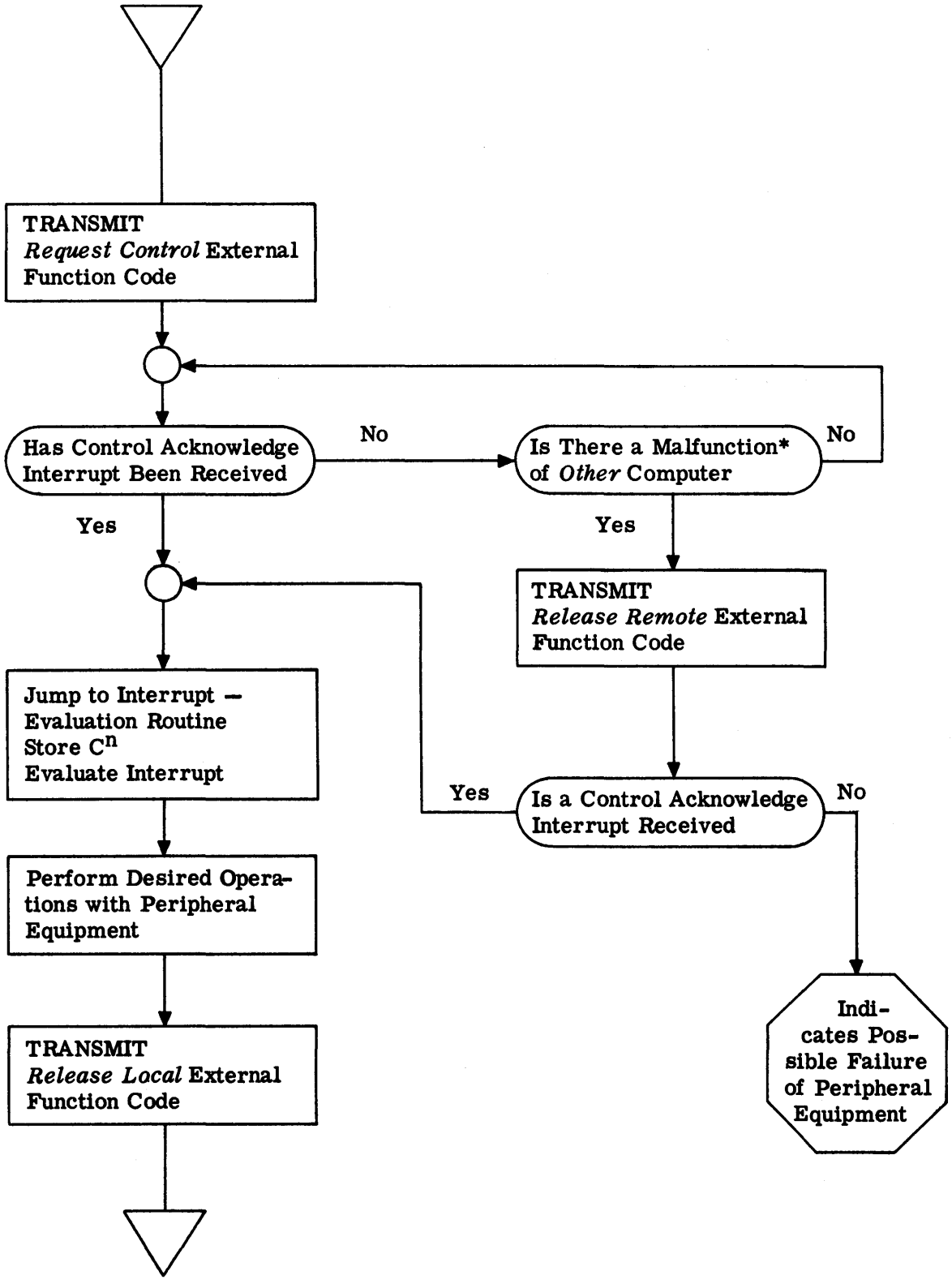
3) The External Function code *Release Remote* will be used only when the computer controlling a peripheral equipment is incapable of releasing control and only after the computer desiring control has sent a *Request Control* External Function code for which no *Control Acknowledge* Interrupt has been received. The use of *Release Remote* will be defined by the System Executive Control Program.

4) The *Release Local* External Function code will not be sent by a computer until it recognizes that the last External Function code it sent has terminated.

5) The *Request Control* External Function code will not be sent by a computer if it already has control of the peripheral equipment.

6. TYPICAL DUPLEX OPERATION

Figure D11-2 illustrates the normal manner in which the duplexing External Function codes are utilized. This same procedure is used in initial start-up assuming that the peripheral equipment has been placed in a neutral condition in preparation for initial start-up.



*See Section 5, "Programming Rules".

Figure D11-2. Computer Utilization of Duplexing External Function Code

SECTION E

UTILITY ROUTINES

Remington Rand

INTRA-COMPANY COMMUNICATION

FROM: John Kourajian
TO: St. Paul
CITY & DATE: St. Paul - July 24, 1961
PERSON OR DEPARTMENT: M. M. Koschmann
DEPARTMENT:
CARBONS:
SUBJECT: Service Test Utility Package

SERVICE TEST UTILITY PACKAGE

A. BASIC INFORMATION

The Service Test Utility Package (STUP) is a special modification of the NTDS Utility System. Desirable features of the Utility System and of the Paper Tape Control Program for use with Flexowriter and of Paper Tape Control Program for use with Teletype have been combined with other specific requests to form one complete package. Paper-tape and magnetic tape formats processed by STUP, as well as the Utility System, are consistent with those generated by the NTDS CS-1 Compiler. For description of formats see Programmers Guide, Section E1 , NTDS Utility Section. The final package contains the following routines:

A. Paper-tape operations

1. Bioctal, Flex, and TWX dumps, all with zero suppression
2. Bioctal checkread with either Flex or TWX code punched tape output of discrepancies
3. Bioctal, Flex, TWX, and Relative-bioctal loads
4. Duplicate paper tape of any format

B. Magtape operations

1. Octal dump
2. Octal and Relative loads
3. Octal checkread with either Flex or TWX code punched tape output of discrepancies
4. Rewind magtape and postioning capabilities
5. Special Write-STUP routine to permit bootstrapping into memory
6. Duplicate magtape

C. Miscellaneous operations

1. Fault condition display
2. External interrupt control
3. Inspect and Change
4. Store Q
5. Clear memory
6. Memory Search (pooch) with either Flex or TWX code punched tape output of "finds"
7. List area on on-line Teletype
8. Suppress and reinstate monitoring-printouts
9. Suppress and reinstate System Monitoring Panel display

STUP occupies core memory addresses 00140 to 03000. Addresses 03000 to 04000 are reserved for programs which make linkage with STUP. The entire area, 00140 - 03777, is written on magtape by a special write routine which permits it to be easily bootstrapped into memory.

STUP is separated into two sections, a Minimum package and the remainder of the package which attaches to the Minimum package. The Minimum package contains the following routines.

1. Bioctal, Flex, and TWX loads from paper tape
2. Octal load from magtape
3. Fault condition display
4. External interrupt control
5. Suppress and reinstate monitoring-printouts and SMP display

If desired the programmer may load into memory only the Minimum package.

Two separate STUP exist; one for use with File Computer Tape Units and the other for use with Service Test Tape Units. The two packages differ only in that they reference different tape units; the operations performed, formats, and the method of initiating commands of each package are identical.

B. OPERATION DEFINITIONS

I. PAPER-TAPE OPERATIONS

1. ABSOLUTE-BIOCTAL DUMP WITH ZERO SUPPRESSION OPTION

Absolute-bioctal format on punched paper tape is 1) leader, 2) 76 code, 3) initial address, 4) final address, 5) contents of included memory addresses, and 6) check sum.

If the zero suppression feature is selected the dump routine first scans the specified memory area and divides it into zero and nonzero segments. The nonzero segments are then dumped in a series of bioctal dump formats on one tape, separated by short leader (four frames of 00 codes). The entire series is loadable as if in one operation.

2. ABSOLUTE-FLEX DUMP WITH ZERO SUPPRESSION OPTION

Absolute-flex format on punched paper tape is 1) leader, 2) beginning codes, 3) each address and each content of the included memory addresses, 4) end codes, and 5) checksum.

All zero cells are ignored if the zero suppression feature is selected. The tape produced may be listed on the Flexowriter.

3. ABSOLUTE-TWX DUMP WITH ZERO SUPPRESSION OPTION

Absolute-TWX format on punched paper tape is 1) leader, 2) beginning codes 3) each address and each content of the included memory addresses, 4) end codes, and 5) checksums.

All zero cells are ignored if the zero suppression feature is selected. The tape produced may be listed on the Teletype.

4. ABSOLUTE BIOCTAL CHECK READ

This routine checks an absolute-bioctal punched tape with associated core-memory cells to assure good bioctal dumps. If the check read detects a discrepancy, the A register holds the word from tape and the Q register holds the memory word; the computer

stops. At this time the content of Q may be manually changed; on restarting the computer, the content of Q is returned to the address from which it was taken and the check read continues. At the operators' option a punched tape output of all discrepancies may be obtained in either Flexowriter code or Teletype code. The format on the tape is leader, format code, the address of the discrepancy, the memory word, and the word from tape. The output tape produced is able to be listed on the respective equipments; it can be reloaded into the computer with the memory word taking precedence over the tape word.

5. ABSOLUTE-BIOCTAL LOAD

A bioctal program must begin with a 76 (code for bioctal tape). Immediately after the 76, the initial and final address are loaded and the rest of the program is loaded without additional addresses. An automatic checksum computation is made. Additional programs are automatically read in if they are separated by short leader (four frames of 00 codes).

6. ABSOLUTE-FLEX LOAD

This routine enables the computer to accept instructions from Flex-coded punched paper tape. The Absolute-Flex must begin with a 45 code (carriage return), followed by a 60 code (8), followed by a 45 code. If checksums are required, two 60 codes indicate the appearance of checksums at the end of the Flex tape. The computer recognizes only the first 15 octal characters per instruction. The first five are the address, and the next ten are the instruction word. All other character codes including notes are ignored. A carriage return indicates the end of an instruction. If an address or instruction does not contain the maximum number of digits, an erroneous load occurs.

7. ABSOLUTE-TWX LOAD

This routine enables the computer to accept instructions from TWX - coded punched paper tape. The Absolute-TWX must begin with an 02, a 10 and a 33 code (carriage return, line feed, figures) followed by a 14 code (8), followed by an 02 and a 10 code. If checksums are required, two 14 codes (88) appear. Each instruction of a TWX tape is preceded by a five digit address, the addresses need not be sequential; the next ten octal-characters comprise the instruction word. All other characters including notes are ignored. The codes 02 and 10 (carriage return, line feed) indicate the end of the instruction. An end code of double period (..) terminates the load. If there is a checksum, it follows the double periods and is computed automatically by the load routine. When producing a TWX tape at the teletype the following considerations should be adhered to:

a) A carriage return must be followed by a "line feed"

b) Leader must be a "letters" code (37)

c) To stop Teletype reader two blank frames must follow the double periods.

8. RELATIVE-BIOCTAL LOAD

A Relative-bioctal program must begin with a 75 (code for Relative-bioctal tape). The entire program is written relative to zero. A numeric code preceding each programmed instruction tells the computer how to modify the instruction for storage at a modified address. A control digit of 7, followed by checksum, terminates the load and initiates an automatic checksum computation.

9. DUPLICATE PAPER TAPE

This routine identically duplicates a punched paper tape of any format. The tape is read in via the High Speed Reader and buffered to the High Speed Punch. Core memory addresses 75000 thru 77777 are used as a buffer storage area.

II. MAGTAPE OPERATIONS

1. OCTAL DUMP

A dump of a specified area of core memory on magnetic tape is given in octal format only. An internal program check is made to insure that the base address of the dump is 4000 or greater. An octal dump onto tape is preceded by a header; the operator may assign a name to the dump by manually entering into the A and Q registers the XS^3 coded name. A maximum of ten characters is permitted. If no name is entered, the title Memory Dump is placed in the header blockette. The operator may select the position on tape, relative to existing records on the tape, to place the octal dump.

Data are checksummed as transferred from core memory and the checksum word is written on tape. The record is automatically checkread to insure correct data transfer. If incorrect, the entire octal dump procedure is repeated. An error indication is given if the data are not able to be written correctly; the tape is repositioned to the beginning of the record.

2. OCTAL LOAD

This routine reads an octal program from magtape and stores it into its associated core memory area; the addresses for the load are obtained from the magtape. The Teletype prints out the name of the program and the base address of the load. If necessary two attempts at reading are made, if both attempts fail a checksum error is indicated. If either case, the tape is positioned at the end of the record requested.

3. RELATIVE LOAD

The relative program on magtape is written relative to zero. A numeric code associated with each instruction tells the computer how to modify the instruction for storage at a modified address. The starting address of the load is furnished at the console by the operator;

a Teletype printout warns the operator to enter the base address if he has not done so. The Teletype prints out the name of the program and the base address of the load. If necessary two attempts at reading are made; if both attempts fail a checksum error is indicated. In either case, the tape is positioned at the end of the record requested.

4. OCTAL CHECKREAD

This routine checks a program of octal format on magtape with associated core-memory addresses. If the checkread detects a discrepancy, the A register holds the word from magtape and the Q register holds the memory word; the computer stops. At this time the content of Q may be manually changed; on restarting the computer, the content of Q is returned to the address from which it was taken and the checkread continues. At the operators' option a punched paper tape of all discrepancies may be obtained in either Flexowriter code or Teletype code. The format on the tape is the address of the discrepancy, the memory word, and the word from tape. The output tape produced is able to be listed on the respective equipments; it can be reloaded into the computer with the memory word taking the precedence.

5. REWIND

This routine rewinds the tape on the specified tape unit to the beginning of the block.

6. PASS N BLOCKS

This routine positions the magtape at the record specified by the operator. Either the position factor **or** the call number can be used to express the position, relative to existing records on the tape, at which to stop the tape.

7. WRITE-STUP ONTO MAGTAPE

This routine provides the operator a means of writing STUP on magtape in a special format which will premit it to be bootstrapped into memory. Write-STUP rewinds the magtape on the specified tape unit and writes on tape the area 140 to 4000 as records 1 and 2.

8. DUPLICATE MAGTAPE

The Duplicate magtape routine identically duplicates from one magtape onto another the number of records specified by the operator. Both tapes must be positioned prior to initiation of the duplicate routine.

III. MISCELLANEOUS FUNCTIONS

1. FAULT CONDITION DISPLAY

The Fault routine is in memory at all times. Upon computer execution of a 00 or 77 instruction, computer control is automatically transferred to address 00000. A return jump instruction at address 00000 directs control to the fault return which re-establishes the fault condition and the computer stops. At this time register P holds the address of the faulting instruction and register U holds the instruction itself. All other registers are unchanged. The operator may manually enter a valid instruction in register U and restart the computer.

2. EXTERNAL INTERRUPT CONTROL

Before execution of each command requested of the Service Test Utility Package, this routine stores the contents of the External Interrupt addresses and flushes these addresses with a monitoring instruction. If an interrupt occurs while a function is being performed, STUP does not lose control but merely stores the channel on which the interrupt occured, drops the interrupt lockout, and continues its normal function. Upon completion of the function the interrupt addresses are restored to their original state.

3. INSPECT AND CHANGE

This routine causes the content of an inspection address to be entered in register A; register Q is cleared, and the computer stops. A manual change of the instruction may then be made in either register A or Q. On reinitiation of computer action, register Q is returned to the inspection address if it is nonzero, otherwise register A is returned. Contents of sequential addresses are processed with each performance of the Inspect and Change function. The inspection address may be changed at any time.

4. STORE Q

The Store Q function permits the computer operator to load a specified area of memory with a value which he manually enters into the Q register. If Q contains all zeros, the area is cleared.

5. CLEAR MEMORY

This routine clears all core storage above address 04000.

6. MEMORY SEARCH (POOCH)

The Memory Search routine permits the operator to search through a specified area for a memory word which contains a selected value; the operator manually enters at the console the value to be found and the mask on which to search. If delimiting addresses are not stated all of memory is searched. If a find occurs the computer stops; B^4 is set to the address of the find and register Q holds the entire contents of that address. At this time the content of Q may be manually changed; on restarting the computer, the content of Q is returned to the address from which it was taken and the Memory Search continues. At the operators' option a punched paper tape output of all finds may be obtained in

either Flexowriter code or Teletype code. The format on the tape is the address of the find, the memory word, and the mask. The output tape produced is able to be listed on the respective equipments; it can be reloaded into the computer with the memory word taking precedence over the mask.

7. LIST AREA

The List Area routine produces a printout on the on-line Teletype of the addresses and the contents of a selected area of memory. All zero cells are ignored if the zero suppression feature is selected.

8. MONITORING - PRINTOUT SUPPRESSION

All monitoring printouts directed to the on-line Teletype are suppressed by initiation of this routine. Successful completion of all STUP functions are shown by 4-stops. Error indications that would normally cause a Teletype printout are then shown by registers A and Q set/all 1's and the computer"jumps to itself"in a loop. On initial loading of STUP, monitoring - printouts are not suppressed.

9. REINSTATE MONITORING - PRINTOUT

This routine provides a method of re-establishing monitoring - printouts on the on-line Teletype. On intial loading of STUP monitoring - printouts are not suppressed.

10. SUPPRESS SMP DISPLAY

In all load and dump routines within STUP, each address as it is being processed is sent to the System Monitoring Panel as an external function word. This feature is suppressed by initiation of the Suppress SMP Display routine. On initial loading of STUP, SMP Display is suppressed.

11. REINSTATE SMP DISPLAY

This routine provides a means of re-establishing the address display at the SMP console. On intial loading of STUP, SMP Display is suppressed.

C. LOAD PROCEDURE

I. From paper-tape (for both ST. tape unit package and FC tape unit package)

1. Place STUP tape in reader
2. Master clear computer, set A sequence
3. Set Key 1 if desired to load only Minimum package
4. Press Automatic Recovery switch down

II. From FC tape unit

1. Place magtape on tape drive
2. Place bootstrap tape* in reader
3. Master clear computer, set A sequence
4. Set unit number in B^7, i.e., U0000
5. Set Key 1 if desired to load only Minimum package
6. Press Automatic Recovery Switch down

III. From ST tape unit

1. Place magtape on tape drive
2. Switch tape unit number to 1
3. Master clear computer, set A sequence
4. Set Key 1 if desired to load only Minimum package
5. Press Automatic Recovery switch down**

* The STUP bootstrap tape is read in via the wired-memory paper-tape bootstrap program. The tape contains a program to rewind the magtape, if necessary, and read in the Minimum package - the first record on the tape. The remainder of STUP, the second record on the magtape, is read in depending on the Key 1 setting.

** This activates the wired-memory magnetic-tape bootstrap program which rewinds the magtape, if necessary, and reads in the Minimum package - the first record on the tape. The remainder of STUP, the second record on the magtape, is read in depending on the Key 1 setting.

D. OPERATING INSTRUCTIONS

I. PAPER TAPE OPERATIONS (jump to STUP switcher is at address 00001)

B^7 Entry	Function	Other Parameters
00002	Absolute-bioctal dump - -	B^6 - base address
00012	(with zero suppression)	B^5 - last address
00003	Absolute-flex dump - - -	B^6 - base address
00013	(with zero suppression)	B^5 - last address
00004	Absolute-TWX dump - - -	B^6 - base address
00014	(with zero suppression)	B^5 - last address
00010	Absolute-bioctal checkread - -	None
00013	(With flex-code output)	
00014	(with TWX-code output)	
00000	Absolute-bioctal load - - -	None
00000	Absolute - flex load - - -	None
00000	Absolute- TWX load - - -	None
00000	Relative - bioctal load - - -	B^4 - base address
00055	Duplicate paper tape - -	None

II. MAGTAPE OPERATIONS (jump to STUP switcher is at address 00001)

B^7 Entry	Function	Other Parameters
U02CN*	Octal dump	B^6 - base address B^5 - last address B^4 - position factor* XS^3 - name in Q and A
U03CN*	Octal load	B^6 - position faction*
U03CN*	Relative load	B^6 - position factor* B^4 - base address
U0100 U0103 U0104	Octal checkread (with flex-output) (with TWX-output)	B^6 - position factor*
U0400	Rewind magtape	None
U0500	Pass n blocks	B^6 - position factor*
U1300	Write-STUP on magtape	None
00600***	Duplicate magtape	**B^6 - from unit **B^5 - to unit

* U is the tape unit number. CN is the call number of the program on tape and represents the position of the program (or record) on the tape. If CN is given the tape is rewound and passed forward CN-1 records. If CN is 0 then the position factor is interrogated. The position factor provides for forward or backward tape movement before execution of a tape function. It takes the form 400XX for backward movement and 000XX for forward movement. XX represents the number of blocks desired to pass. If both CN and the position factor are zero then no initial tape movement is indicated.

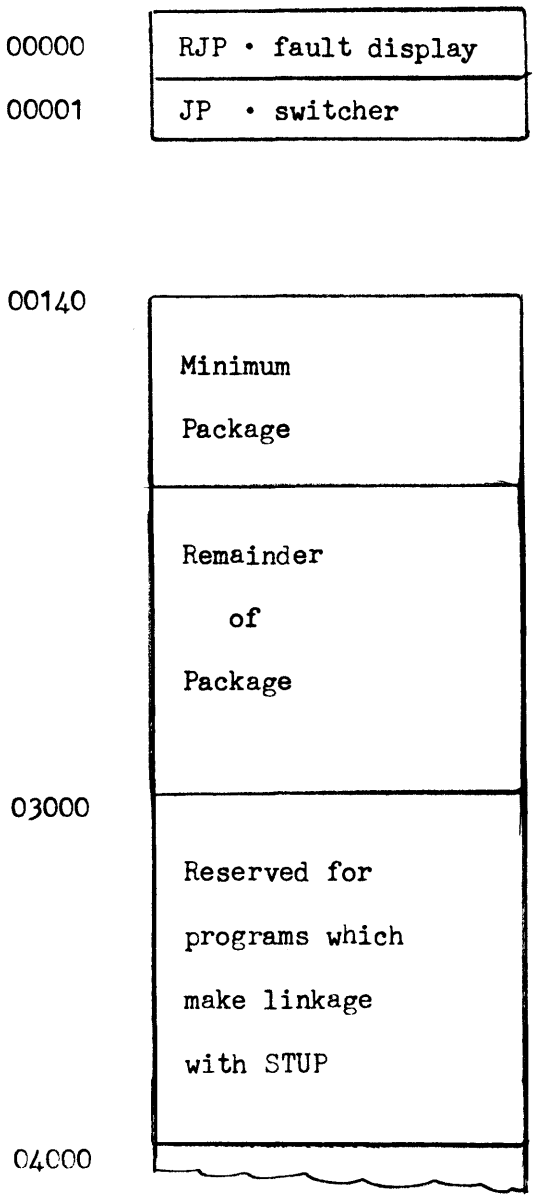
** Where unit number is given in the upper 3-bits, i.e., U0000.

*** Where the lower 2 digits represent the number of records to duplicate, i.e., 00605, indicates duplicate five records.

III. MISCELLANEOUS FUNCTIONS (jump to STUP switcher is at address 00001)

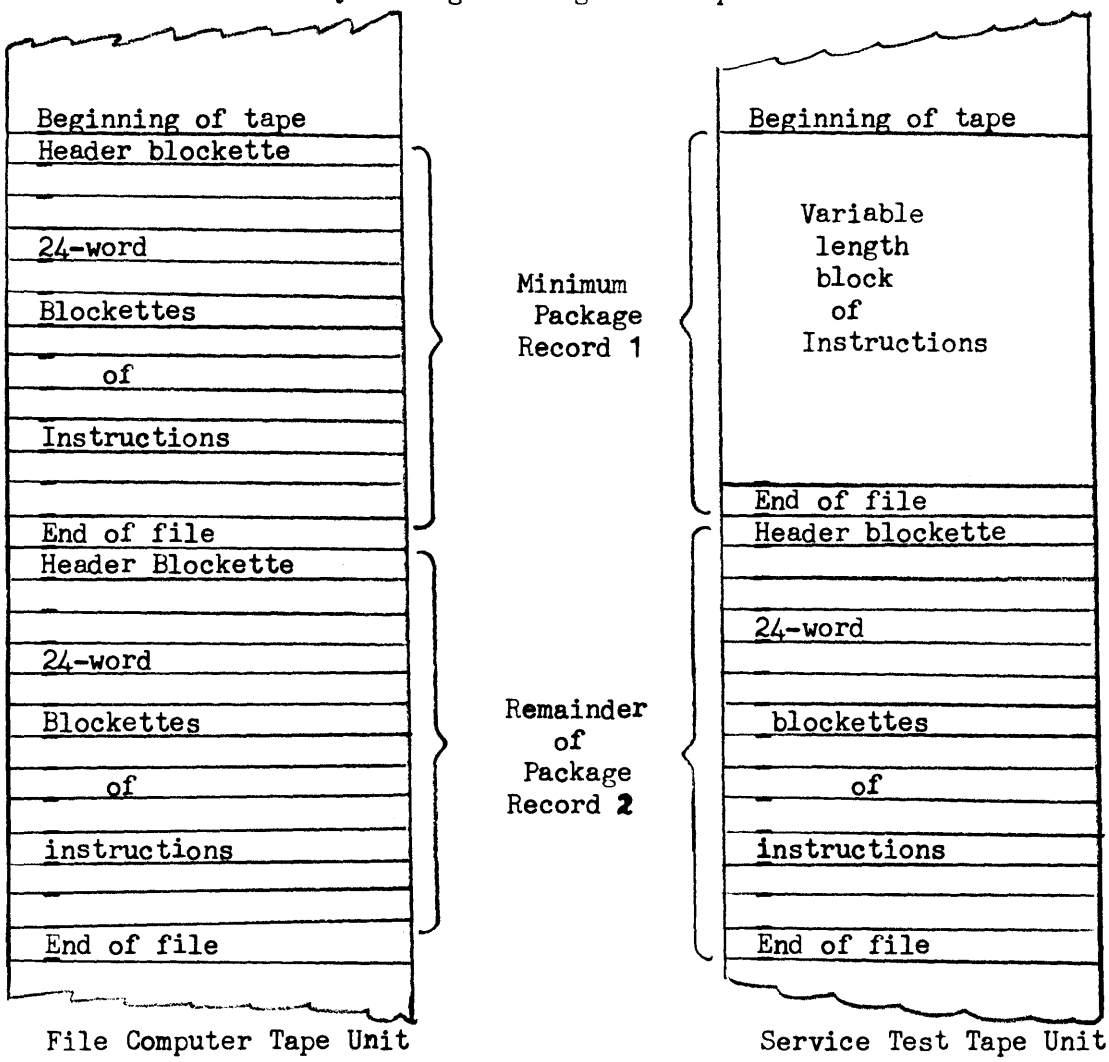
B^7 Entry	Function	Other Parameters
None	Fault condition display	None
None	External interrupt control	None
70707	Clear memory	None
01000	Memory search (pooch)	B^6 - base address of search
01003	(with flex-output)	B^5 - last address of search
01004	(with TWX-output)	Value in A
		Mask in Q
01100	Inspect and Change	B^4 - inspection address
01200	Store Q	B^6 - base address of area
		B^5 - last address of area
		word to be stored in Q
01400	List area	B^6 - base address of area
01410	(with zero suppression)	B^5 - last address of area
01600	Monitoring-printout suppression	None
01700	Reinstate monitoring-printout	None
02000	Suppress SMP display	None
02100	Reinstate SMP display	None

E. Position of Service Test Utility Package in Core Memory



The exact location in memory of a specific routine may be obtained from a current allocation output for the respective packages.

F. Service Test Utility Package on Magnetic Tape



Records 1 and 2 contain the instructions for core memory addresses 00140 thru 03777. The minimum package is read into memory by the respective bootstrap programs - for FCTU, a bootstrap program on paper-tape and for STTU, the wired-memory bootstrap program.

SECTION E1

NTDS UTILITY SYSTEM

CONTENTS

ILLUSTRATIONS

TABLES

NTDS UTILITY SYSTEM

1. BASIC INFORMATION

The NTDS Utility System is a collection of programs catalogued on one magnetic tape called the Utility Tape. The Utility System provides computer operators with easy access to frequently used programs. These programs may be:

1) Program diagnostic routines such as TRACE III, CS-1 debugging packages, and STORAGE PRINT
2) Computer maintenance and diagnostic routines
3) Standard load and dump routines for paper tape and magnetic tape
4) Duplication and verification routines for paper tape and magnetic tape
5) Any other programs frequently used by computer operators.

The Utility System is programmed for AN/USQ-17 and AN/USQ-20 Unit Computers and uses File Computer Magnetic Tape Units (4950). Because paper tape and magnetic tape formats are the same for AN/USQ-20 and AN/USQ-17 Unit Computers, Utility Tapes for the AN/USQ-20 Unit Computers will be prepared by the Utility Tape Update Program on the AN/USQ-17 Unit Computer until the program is modified to be operational on the AN/USQ-20 Unit Computer.

Two programs control the Utility Tape: 1) Utility Tape Load Program (UTLP) which brings various programs from the Utility Tape into the core memory and 2) Utility Tape Update Program (UTUP) which adds, deletes, and replaces programs on the Utility Tape. A Utility Tape Directory categorizes programs on the Utility Tape. This directory is built by the Utility Tape Update Program and is used by the Utility Tape Load Program when loading a program. These programs detect various errors and direct their type-out on the Flexowriter and also list pertinent information on the High Speed Printer.

The Utility System handles all data on magnetic tapes with check-sum computation to insure accuracy of data transfer.

Computer operators control the Utility Tape by communicating with the Utility Tape Load and Utility Tape Update Programs.

2. FORMAT DESCRIPTION

A. *UTILITY TAPE FORMAT*

Programs on Utility Tape appear in magnetic tape formats which are either relative (output No. 41) or absolute bioctal (output No. 40) outputs of the CS-1 Compiler. (See PX 1349, Vol. 2.)

Each program consists of physical units of 24 words each (blockettes). (See Figure E1-1.) The first blockette of each program is a header blockette which contains:

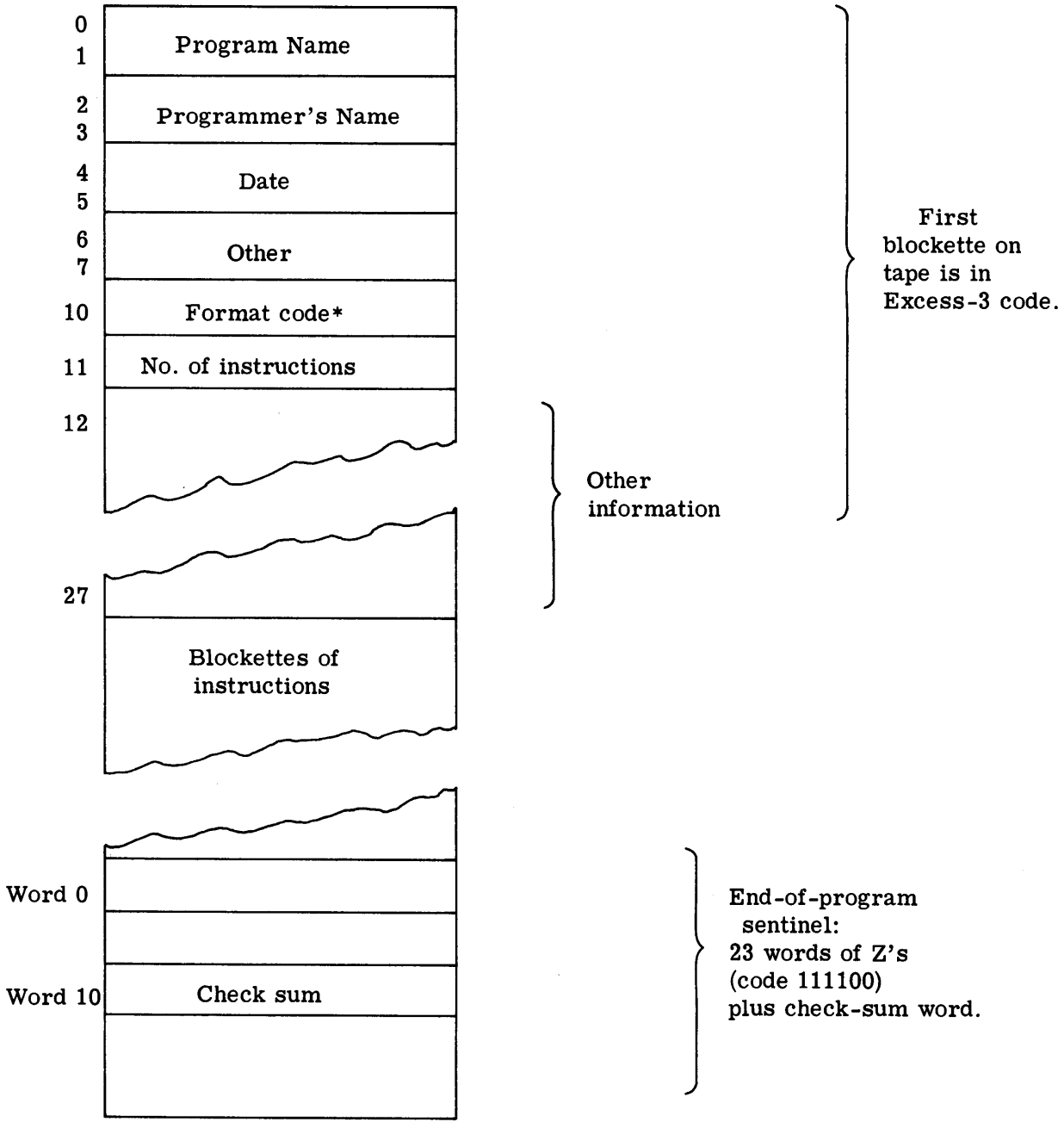
1) The program name, words 0 and 1
2) The programmer's name, words 2 and 3
3) The date, words 4 and 5
4) The format code, word 10, which specifies that the program be in absolute bioctal or relative format
5) The number of instructions, word 11.

All information in the header blockette is in *Excess-three* code. Blockettes of instructions in either absolute bioctal (see Figure E1-2) or relative (see Figure E1-3) follow the header. The program is terminated by an end-of-program sentinel — a blockette which, except for word 10, contains Z's (code 111100). Word 10 contains the check-sum word for the program (see Figure E1-1). The format for check-sum computation appears in Supplement A at the end of this section.

B. *UTILITY TAPE DIRECTORY*

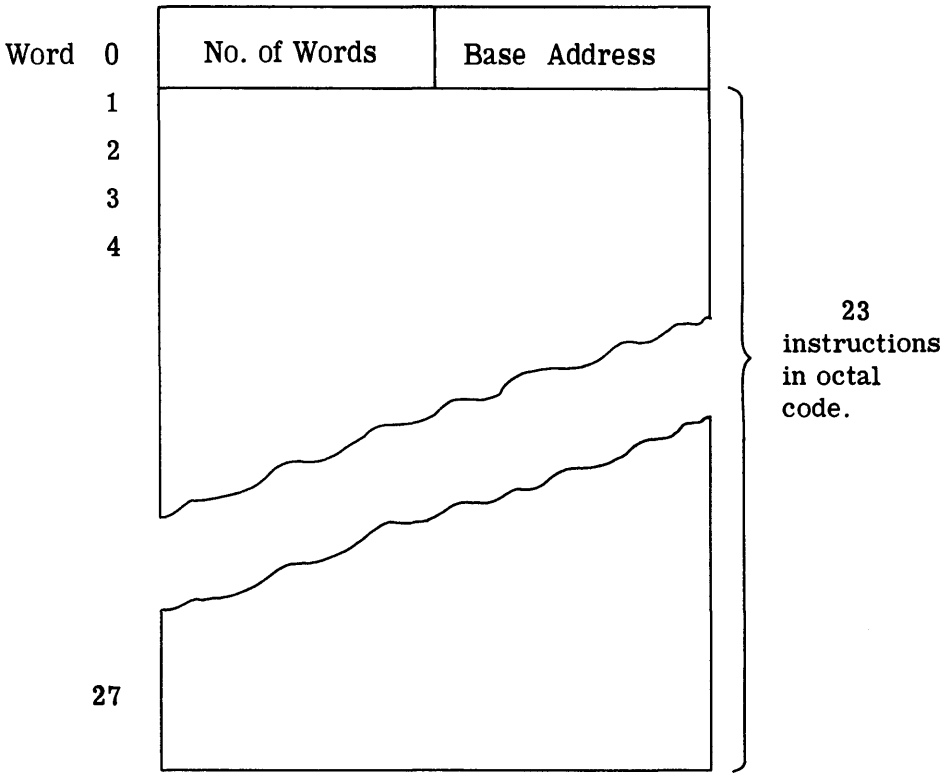
Names of all programs on the Utility Tape appear in the Utility Tape Directory. Each program name consists of two words in Excess-three code. Position of a program name in the directory corresponds to the position of that program on the Utility Tape (see Figure E1-4). The first word of the directory contains the current number of programs, N, listed in the directory.

The Utility Tape Update Program constructs the Utility Tape Directory each time it changes the Utility Tape. It places program names in the directory in the same order as it writes



*If format code is "OCTAL" format is bioctal.
If format code is "RELAT" format is relative.

Figure E1-1. Univac File Computer Magnetic Tape L_4 Format



Word 0 Lower half specifies the address at which word 1 is to be stored; word 2 follows sequentially.

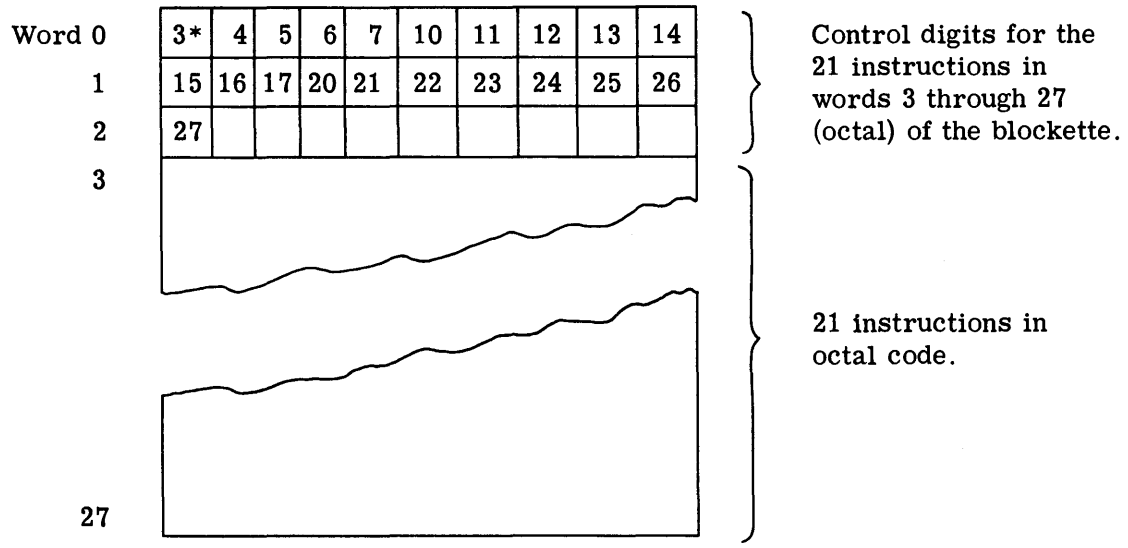
Word 0 Upper half specifies the number of actual instructions in the blockette, 1 through 23.

Figure E1-2. Bioctal Tape Format

them on the new Utility Tape. Names of the first two programs on the Utility Tape are CTRL PKG^{00}, which is contained in words 2 and 3, and $PTAPECTL^{00}$, which is contained in words 4 and 5.

C. *UTILITY TAPE LOAD PROGRAM*

After receiving instructions from the computer operator, the Utility Tape Load Program, aided by information from the Utility Tape Directory, locates the desired program on the Utility Tape. It then performs necessary modifications to program instructions while the program is transferred from the Utility Tape into core storage. The two tape formats used are: 1) relative and 2) octal (absolute). Twenty-four-word blockettes are transferred into core storage through a buffer area. Transfer is under control of the File Computer Magnetic Tape Control Program. See NTDS Technical Note No. 246, *Univac File Computer Magnetic Tape Control System Program.*



*Controls the address modification of the instruction in word 3.

Values 3 through 27 refer to the word numbers (in octal) in the blockette that contain the instructions to which the control digits apply. Each control digit consists of three bits whose range of values and meanings relative to the instruction are as follows:

0 - make no modification

1 - modify the lower half with the initial address

2 - modify the upper half with the initial address

3 - modify both upper and lower halves with the initial address

4 - perform the instruction, then continue loading

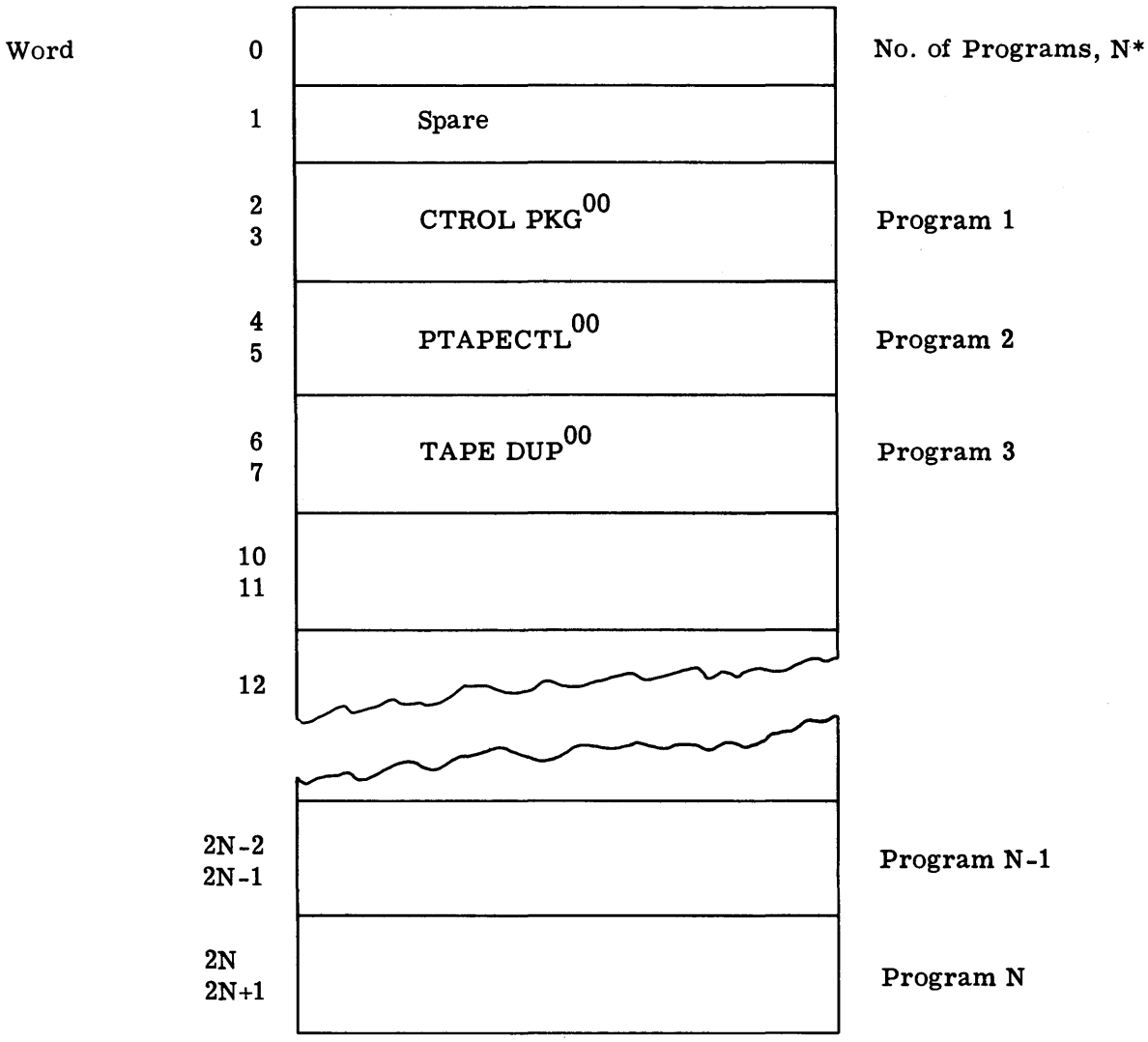
5 - modify the lower half with (B^5)

6 - modify the lower half with (B^6)

7 - terminate load on the blockette

One of these control digits actually occurs in each of the numbered locations of the blockette above.

Figure E1-3. Relative Tape Format



*N≤63

Figure E1-4. Directory Format - Excess-Three Code

(1) *Relative*

When the program format is relative, the computer modifies each instruction word on the basis of a 3-bit control digit assigned to it and then transfers the modified word to storage. The first three words of each blockette contain control digits sequentially assigned to the remaining 21 instruction words in the blockette. Significance of control-digit values is shown in Figure E1-3.

(2) *Octal*

When the program format is octal, the computer stores the first instruction of the blockette in core storage at the address given in the lower half of the first word in the blockette. Remaining words are stored consecutively (see Figure E1-2).

D. *UTILITY TAPE UPDATE PROGRAM*

The Update Program (see Figure E1-5) constructs the Utility Tape from programs supplied from either paper tape or magnetic tape and updates the Utility Tape Directory. A set of control and edit routines is always placed on the Utility Tape. It also includes those frequently used NTDS programs desired by the individual programmer. The Update Program detects various errors and limitations and directs their Flexowriter type-out. Lists of error and information type-outs appear in Subsection 5.E.

The Update Program writes all programs on the Utility Tape with a check sum. It uses this check sum as an accuracy check during later transfer of a program to a New Utility Tape. The check sum is also used when a program is read into the computer by the Control Package. Supplement A at the end of this section shows format for check-sum computation.

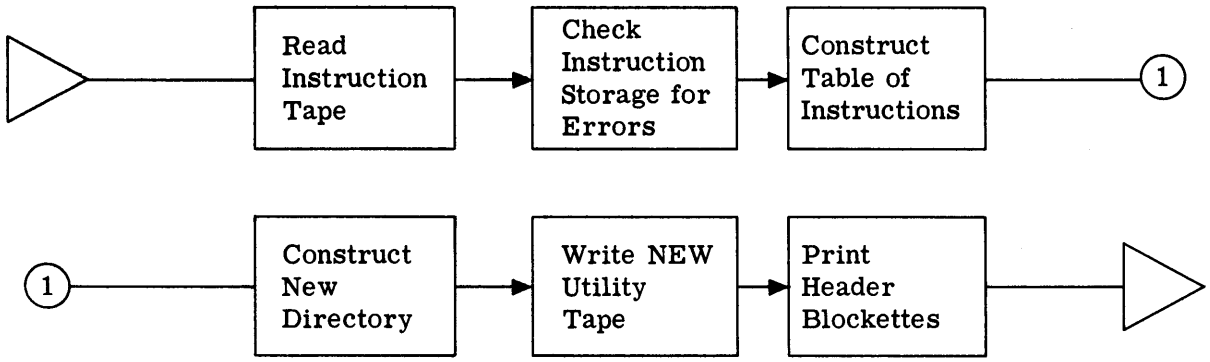


Figure E1-5. Block Diagram of Utility System Update Program

The Update Program performs the following general operations on the Utility Tape:

1) It moves a program to a different position on the tape

2) It replaces a program with a modification of the program

3) It adds new programs to the tape

4) It deletes programs from the tape.

A set of action operations placed on an independent instruction tape, to be described later, prescribes the above activities. The program permits as many as 30 of these operations (at the programmer's option).

Finally, the Update Program effects High Speed Printer print-out of header blockettes on the updated Utility Tape. Figure E1-6 illustrates performance of the Update Program.

(1) *Tape Formats*

The Update Program places programs on the Utility Tape in either of two formats (see Figure E1-7):

1) CS-1 Compiler Format, Absolute Bioctal

2) CS-1 Compiler Format, Relative.

Magnetic tape formats of programs that serve as inputs to the update process are also either the octal or relative format of the L_4 output of the CS-1 Compiler. Magnetic tape inputs retain the same format when transferred onto the Utility Tape; that is, relative remains relative and bioctal remains bioctal. Figures E1-1 through E1-3 give a more detailed description of these formats.

Five paper tape formats also acceptable as input to the Utility Tape include:

1) Absolute Flexcode

2) Relative Flexcode

3) Absolute Bioctal

4) Relative Bioctal

5) Upper Service Library Bioctal.

Formats of 2) and 4) above become relative on magnetic tape; formats of 1), 3), and 5) be-

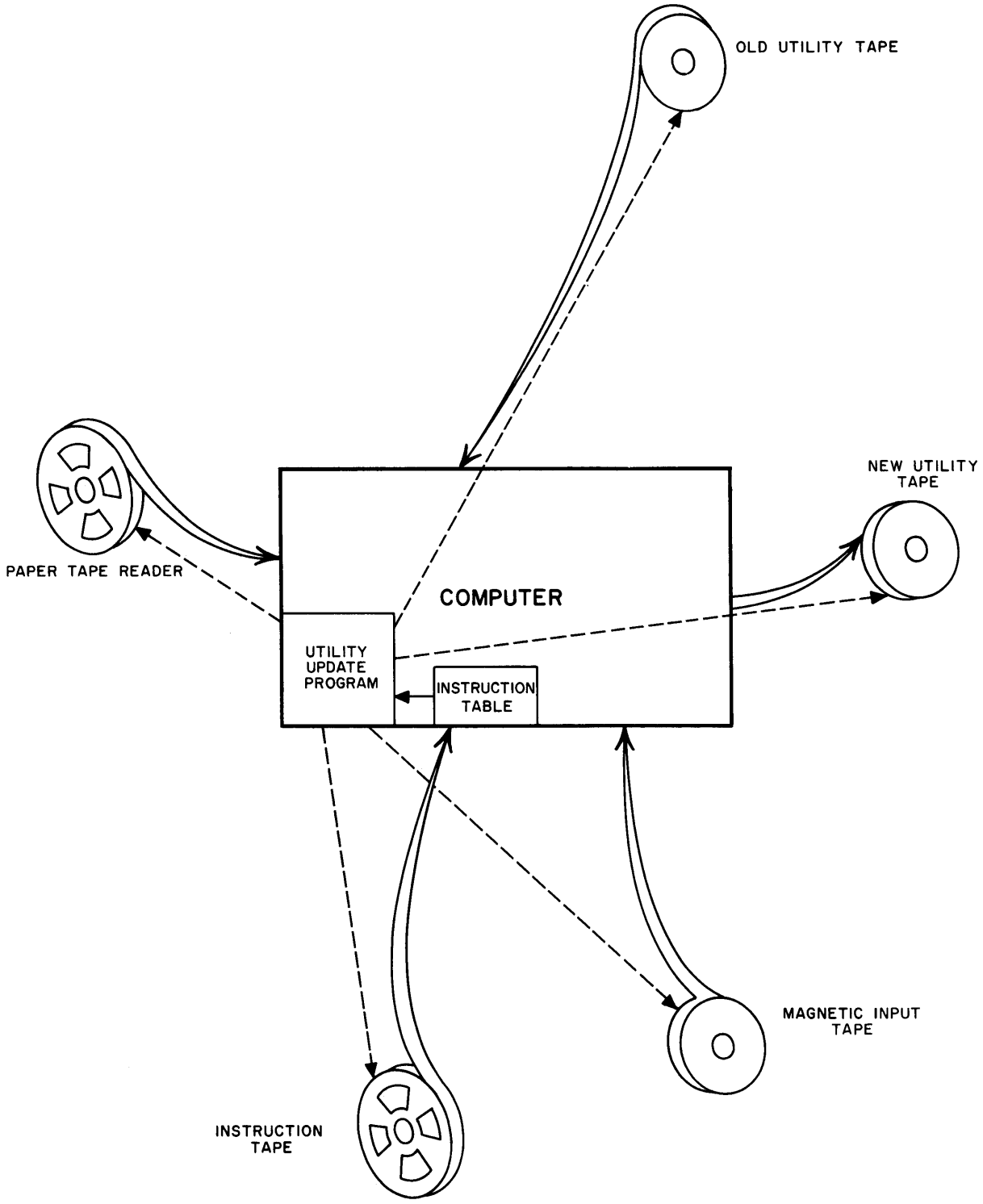


Figure E1-6. Performance of Utility System Update Program

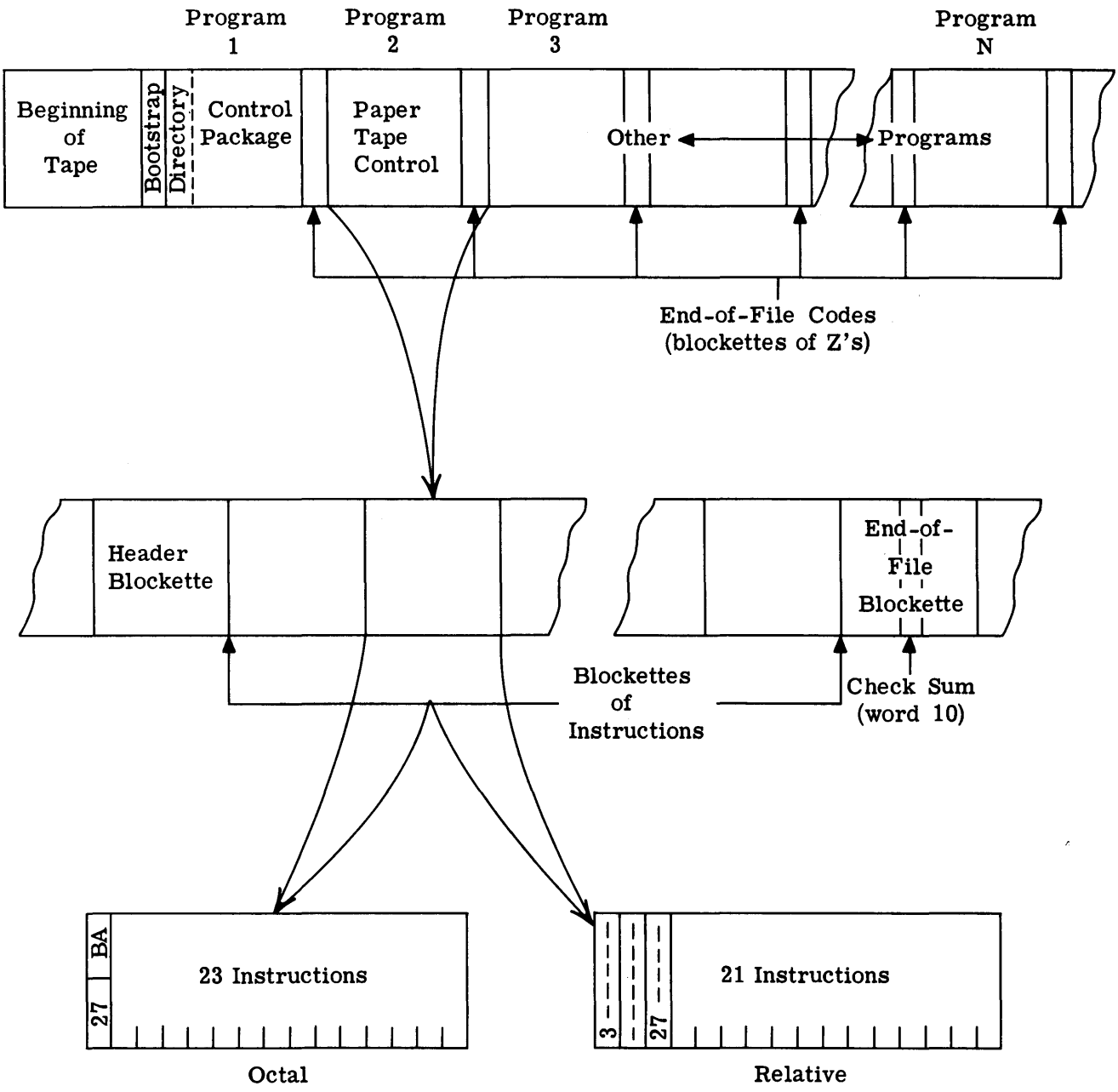


Figure E1-7. Arrangement and Format of Utility Tape

come absolute. Programs from paper tape are transferred to the Utility Tape under control of load routines contained within the Update Program itself.

A more detailed description of paper tape formats appears in Table E1-1.

(2) *Program Headers*

A header blockette must appear at the beginning of each program on magnetic tape. When the input is from magnetic tape, the header from the input tape serves as the program header on the Utility Tape. When the input is from paper tape, the header blockette consists of V_0 and V_1 operands from action operations (see Subsection 3.C, "Action Operations"). Several paper tapes may be combined under one program header, provided they are of the same basic format (relative or absolute). Setting a key on the console, as the information con-

TABLE E1-1. LOAD TYPES

Load Type	Character Code	Beginning Code	End Code	Automatic Check-Sum Computation*
Header	Flex	Upshift	Leader	No
Absolute (Flex)**	Flex	8	· ·	No
Absolute (Flex)**	Flex	88	· ·	Yes
Relative ***	Flex	9	7.	No
Relative ***	Flex	99	7.	Yes
Relative-Bioctal ***	Octal	75	none	Yes
Absolute-Bioctal ****	Octal	76	none	Yes

*Check-Sum Words (upper and lower sums) appear immediately following the End Codes.

**Five Flex codes must precede each instruction to indicate storage address.

***All addresses must be relative to zero. A control digit which tells the computer how to modify the instruction for storage precedes each instruction.

****Initial and final addresses are given following the beginning code.

tained on the paper tapes is transferred to the magnetic tape, accomplishes this combining operation.

(3) *Tape Unit Requirements*

The Update Program normally requires three tape units; this holds true when a NEW Utility Tape is built from an OLD Utility Tape and another INPUT tape. Other input combinations and corresponding tape unit requirements appear in Table E1-2.

The computer operator specifies, by means of tape unit number, tape units which are to be used; he places tape unit selection at the beginning of the Instruction Tape (see Subsection 3.B, "Allocation Operations", and Figure E1-8).

TABLE E1-2. UTILITY UPDATING REQUIREMENTS

Input Tape	Paper Tape	OLD Utility Tape	NEW Utility Tape	Instruction Tape	No. of Tape Units Needed	Applicable Action Operations
X	X	X	X	X	3	all
X	X		X	X	2	ADD only (initial tape)
		X	X	X	2	DELETE and MOVE
	X		X	X	1	ADD only (initial tape)
X			X	X	2	ADD only (initial tape)
	X	X	X	X	2	all
X		X	X	X	3	all

A-1206-1

TITLE Sample Instruction Tape
PAGE 1 of 1

NTDS CS-1 CODING FORM

PROGRAMMER R. Scholz
PLT. 5 EXT. MS 515
DATE 8-1-60

LABEL	OPERATOR	OPERANDS AND NOTES
	HEADER TYPE OLD	• 1
	NEW	• 2
	INPUT	• 3
2•0	UPDATE	• PTAPECTL[00] • RELATIVE • PTAPE
3•0	DELETE	• SIMP[02]
4•1	INSERT	• RET[01] • OCTAL • MTAPE
6•1	MOVE	• TRACE[03]
13•0	ADD	• GOOF[04] • OCTAL • PTAPE
14•0	MOVE	• STRPRINT[02]
		•
		•
		•
		•
		•
		•
		•
		•

Key:

[identifier]	[operator]	• [program name] • [format] • [medium]
		•
		•
		•
		•
		•
		•
		•

Figure E1-8. NTDS CS-1 Coding Form

3. INSTRUCTION TAPE

An important part of the Utility Tape Update Program is the *Instruction Tape*. With it, the operator gives all pertinent directions for performing desired updating activities. Primary features of the *Instruction Tape* are: 1) the identifiers, 2) the allocation operations, and 3) the action operations. The reader is requested to refer to the sample Instruction Tape in Figure E1-8 while reading the following discussion of these features.

A. *IDENTIFIERS*

An *identifier* is a set of octal digits which uniquely distinguishes a program that is either already on or is to be added to the OLD Utility Tape. It is comprised of two parts separated by a *point* separator. The part to the left of the *point* is termed *call number*; it identifies the numerical position of a program (on the OLD Utility Tape) after which a number of programs are to be inserted. The part to the right of the *point* is termed *insert number*; it indicates the order in which programs are to be inserted between the program defined by the *call number* and the next program on the OLD Utility Tape. Each part may have no more than two digits. The part to the left of the *point* cannot be 0; it ranges from 1 to 77 (octal). The part to the right of the *point* must be 0 whenever no other digit applies. Identifiers need not be listed either sequentially or consecutively; the Update Program arranges them sequentially and assigns consecutive values in the Directory of Routines.

The identifier is used on the Instruction Tape only with Action Operations; it appears in the LABEL portion of the operation.

Example of Identifier: 16 • 12

The program identified is program number 12 to be inserted on the OLD Utility Tape after program 16 and before program 17. (All values are octal.)

These inserted programs may come from: 1) an input tape, 2) a paper tape, or 3) another location on the Utility Tape itself. Insert digits apply only to INSERT and MOVE operations. Other identifier rules are:

1) An identifier with two zeros to the left of the *point separator* is not permitted
2) An identifier with a *call number* less than two is not permitted with INSERT or MOVE operators
3) An identifier with a *call number* less than three is not permitted with the DELETE operator.

The Control Package is permanently assigned a *call number* of one; Paper Tape Control Program is assigned a *call number* of two.

B. *ALLOCATION OPERATIONS*

Tape unit assignments are the first operations performed by the computer operator in preparing the Instruction Tape. Three tape unit assignments are employed:

1) OLD - states which tape unit contains the OLD Utility Tape

2) NEW - states which tape unit contains the blank tape which is to become the NEW Utility Tape

3) INPUT - states which tape unit contains compiler output magnetic tapes from which additions to the Utility Tape are to be made.

These tape unit assignments are called operators. Allocation operations do not require identifiers. Only one operand, V_0, accompanies an operator. This operand specifies the tape unit number (from 1 through 7) to which the particular tape is assigned. Allocation operations have the following general format (as written by the computer operator):

W V_0

→ [type of tape] • tape unit number)

Examples of allocation operations:

W V_0

a) → NEW • 2)

b) → INPUT • 3)

A NEW allocation operation is required each time a Utility Tape is prepared; the other two operations are included as they apply. Each operator may appear only once on an Instruction Tape. Only one magnetic tape unit is used for input during the updating process; different tapes, however, may be placed on the given tape unit.

C. *ACTION OPERATIONS*

(1) *General Description*

Five operators prescribe the update functions required to build a NEW Utility Tape. These are:

1) ADD

2) DELETE

3) UPDATE

4) INSERT

5) MOVE.

A maximum of 30 action operations may be placed on one Instruction Tape. An action operation has the following standard format:

L W V_0 V_1 V_2

[identifier] ⟶ [type of action] • [program name] • OCTAL or RELATIVE • PTAPE or MTAPE

L consists of the identifier in two parts as described in Subsection 3.A, "Identifiers." Only the operators MOVE and INSERT require a value other than 0 in the insert-number portion to the right of the point separator. The remaining operators require either one or two 0's in this location.

W names the action operator to be used to perform the desired updating action. These operators are named above.

V_0 gives the name of the program being inserted, added, deleted, replaced, or moved. This operand may consist of any combination of up to 10 alphanumeric characters.

V_1 states whether the program specified by V_0 is in relative (RELATIVE) or absolute bioctal (OCTAL) format.

V_2 states whether the program specified by V_0 is on paper tape (PTAPE) or magnetic tape (MTAPE).

V_0, V_1, and V_2 operands are all required for all action operations except MOVE and DELETE.

(2) *Individual Descriptions*

(a) *ADD Operation*

L W V_0 V_1 V_2

[identifier] ⟶ ADD • [program name] • OCTAL or RELATIVE • PTAPE or MTAPE

The ADD operation places a new program from paper tape or INPUT magnetic tape after the last previously stored program on the OLD Utility Tape. The identifier must have a call number greater than the call number of the last program on the OLD Utility Tape. The Update Program arranges these values sequentially and assigns them consecutive values in the Directory of Routines.

Example:

$$16 \bullet 0 \longrightarrow \text{ADD} \bullet \text{PRINTER}^{04} \bullet \text{RELATIVE} \bullet \text{MTAPE}$$

(b) *INSERT Operation*

$$\underset{[\text{identifier}]}{L} \longrightarrow \underset{\text{INSERT}}{W} \bullet \underset{[\text{program name}]}{V_0} \bullet \underset{\text{OCTAL or RELATIVE}}{V_1} \bullet \underset{\text{PTAPE or MTAPE}}{V_2}$$

The INSERT operation causes a program from either paper or magnetic tape to be placed between two programs already on the OLD Utility Tape. The identifier must have a call number which is less than the call number of the last program on the OLD Utility Tape. If more than one program is to be inserted between two programs, the programmer must provide each with a unique octal *insert number* to the right of the *point* separator. These numbers need not be consecutive.

Example:

$$5 \bullet 1 \longrightarrow \text{INSERT} \bullet \text{PACKER} \bullet \text{OCTAL} \bullet \text{PTAPE}$$

(c) *UPDATE Operation*

$$\underset{[\text{identifier}]}{L} \longrightarrow \underset{\text{UPDATE}}{W} \bullet \underset{[\text{program name}]}{V_0} \bullet \underset{\text{OCTAL or RELATIVE}}{V_1} \bullet \underset{\text{PTAPE or MTAPE}}{V_2}$$

The UPDATE operation replaces a program already on the OLD Utility Tape with a new version of the same program. The identifier must have a call number that is equal to the call number of the program to be replaced. The *insert digits* must consist of one or two zeros. Operands V_1 and V_2 must be present.

Example:

$$14 \bullet 0 \longrightarrow \text{UPDATE} \bullet \text{RESE}^{01} \bullet \text{OCTAL} \bullet \text{MTAPE}$$

(d) *DELETE Operation*

L *W* V_0

[identifier] → DELETE • [program name] ↵

The DELETE operation removes a program from the OLD Utility Tape. The identifier has a call number that is equal to that of the program to be deleted. The insert number must consist of one or two zeros. A V_0 operand, which names the program as it appears on the OLD Utility Tape, is required.

Example:

3 • 0 → DELETE • TRACE^{03} ↵

(e) *MOVE Operation*

L *W* V_0

[identifier] → MOVE • [program name] ↵

The MOVE operation transfers a program from its current position on the OLD Utility Tape to a new position on the NEW Utility Tape. If the program is to be placed between two programs already on the OLD Utility Tape, the programmer establishes the identifier according to rules for the INSERT operation. To place the program at the end of programs already on the OLD Utility Tape, the identifier must have a call number that is greater than the call number of the last program on the OLD Utility Tape. Only a V_0 operand is required.

Examples:

1) 22 • 0 → MOVE • COMSOL ↵

2) 5 • 3 → MOVE • PAKIT ↵

4. UTILITY TAPE CONTENTS

The Utility Tape contains the following routines:

1) A Bootstrap routine for loading the Control Package (for AN/USQ-17 Unit Computer).

2) The Control Package consisting of a set of control, editing, and debugging routines

including the Utility Tape Directory. The Control Package is permanently assigned the first program location immediately following the Bootstrap and Directory.

3) The Paper Tape Control Program which is permanently assigned the second program location on the Utility Tape.

4) The Utility Tape Update Program.

5) Other special-use programs optionally selected by the programmer or computer operator.

Routines 1), 2), 3), and 4) above are standard routines always included on the Utility Tape. Locations of 1), 2), and 3) are fixed. The Update Program, together with the optional special-use programs, can be arranged in any order on the Utility Tape. Figure E1-7 illustrates arrangement and format of the Utility Tape.

A. *BOOTSTRAP ROUTINE* (for AN/USQ-17 Unit Computer)

The Bootstrap which loads the Control Package from the Utility Tape into core memory consists of one 24-word blockette of instructions. The computer operator reads the Bootstrap itself from the Utility Tape into memory by entering a manual Bootstrap into the computer via the console (see Subsection 6.B, "Control Package").

B. *CONTROL PACKAGE*

The Control Package contains the following:

1) The Utility Tape Directory - an area of six blockettes reserved at the beginning.

2) The Utility Tape Load Program - performs reading of programs from the Utility Tape into core memory.

3) The General Tape Control Program - performs Read and Write operations on tapes other than the Utility Tape.

4) Miscellaneous editing, debugging, and special-purpose routines. These are listed and described in subsequent text.

The Control Package is the first program stored on the Utility Tape. From this tape, it is brought into the computer by means of a Bootstrap routine. The Control Package assumes

control of all subsequent operations and all communication with tape units. It always retains its position as the first program on the Utility Tape.

The Control Package primarily controls the Utility Tape; it handles the tape during reading of programs from the tape. In addition, it provides for complete control of paper tape load and dump routines and for reading from or writing on magnetic tapes which are on tape units other than the one that houses the Utility Tape.

Of the remaining routines in the Control Package, some have been extracted from the CS-1 Utility Package. See Section E2 of this document or NTDS Technical Note No. 220, CS-1 *Utility Package (Paper Tape)*. The Control Package, therefore, supplants the CS-1 Utility Package although the latter is retained on paper tape for computer operators who wish to use it. Subsection 5.C, "Control Package", describes how the computer operator initiates these routines once they are transferred from the Utility Tape to the computer memory. The following paragraphs describe the purpose of routines available in the Control Package.

(1) *Directory*

The Utility Tape Directory is not a routine, but an area reserved for recording names and numbers of routines on the Utility Tape.

(2) *Utility Tape Load Program*

The Utility Tape Load Program performs modifications to program instructions while a program from the Utility Tape is being transferred into core storage. The two formats on the tape are relative and octal (absolute). See Figures E1-2 and E1-3 for examples of these formats. Twenty-four-word blockettes are *read* in through a buffer area under control of the File Computer Tape Control Program.

(3) *General Tape Control Program*

The General Tape Control Program directs *read* and *write* operations for tapes other than the Utility Tape. Blockettes of 24 words are transferred through a buffer. Transfer occurs via File Computer Tape Control Program. The following operations are performed:

1) Relative Read - performance is the same as for the Utility Tape

2) Octal Read - same as for Utility Tape

3) Octal Dump - in an octal dump operation, the computer transfers to tape (in L_4 octal compiler format) information contained in an area of core storage.

4) Excess-Three (XS-3) from cards (for AN/USQ-17 Unit Computer) - this operation converts information transferred to magnetic tape from punched cards in XS-3 code to octal code. It *reads* information into the computer in 24-word blockettes via File Computer Tape Control Program.

(4) *Miscellaneous Routines*

Editing and debugging routines in the Utility System Control Package include the following:

1) *Store Q** - Permits the computer operator to manually enter a constant value into the Q register and then to store this value in all storage locations within initial and final limits specified by entries in B^6 and B^5, respectively.

2) *Inspect and Change** - Enables the computer operator to bring the content of any memory location which he specifies in B^4 into Q and A registers.

3) *Clear Memory* - Enables the computer operator to direct the computer to clear all core storage above the area occupied by the Control Package and the Paper Tape Control Program.

4) *Dump Registers* (for AN/USQ-17 Unit Computer) - Effects Flexowriter type-out of the contents of various registers.

5) *Directory Print* - Directs the Flexowriter to type a list of all programs in the Directory of Routines.

C. *PAPER TAPE CONTROL PROGRAM*

The Paper Tape Control Program (PTCP), which follows the Control Package on the Utility Tape, is a collection of load/dump routines for paper tape. As a part of the Utility System, it is loaded into core memory from magnetic tape; all commands to the Paper Tape Control Program are directed to it by the Utility System Control Package. The computer operator does this by entering the appropriate function code into the B register reserved for this purpose. Paper Tape Control Program, once loaded in memory, may be used independently if desired.

The Paper Tape Control Program for the AN/USQ-17 Unit Computer is in Relative-Load format; hence, it can be loaded into any available area in memory. Its normal position is memory locations 02600 to 04000. Paper Tape Control Program for AN/USQ-20 Unit Computer is in Octal-Load format and is permanently assigned memory locations 00140 to 01000.

*For the AN/USQ-20 Unit Computer, *Inspect and Change* and *Store Q* routines are functions of the Paper Tape Control Program.

The Paper Tape Control Program loads and dumps programs or data in the NTDS CS-1 assembled program-output formats. Codes punched on the paper tape are either Flex code or octal. Each of the formats mentioned previously is identified by a code number at the beginning of the tape. Code numbers and corresponding formats appear in Table E1-1.

D. *THE UTILITY TAPE UPDATE PROGRAM*

The Utility Tape contains the Utility Tape Update Program already described in Subsection 2.D.

E. *OTHER PROGRAMS*

The programmer selects programs to include on the Utility Tape on the basis of individual requirements. He specifies the order of these programs to the Update Program by means of the Instruction Tape.

5. METHOD OF OPERATION

This subsection describes what actions are required by the computer operator in operating the NTDS Utility System and how the system performs its functions. Also described are error and fault type-outs with instructions pertinent to their correction, and information and status type-outs with subsequent instructions.

A. *INITIAL UTILITY TAPE*

(1) *Requirements*

The first Utility Tape requires the following for its formation:

1) An Instruction Tape
2) A blank magnetic tape on one of the File Computer tape units
3) The Update Program in core storage
4) The File Computer Tape Control Program (FCTCP) at address 1600 in core storage
5) Input tapes (punched paper tapes and/or magnetic tapes) containing programs to be placed on the Utility Tape.

(2) *The Instruction Tape*

The computer operator prepares the Instruction Tape by first stating the tape-allocation operations. These consist of: 1) a NEW operation with a V_0 operand stating the tape unit number on which the NEW Utility Tape will be placed and 2) an INPUT operation stating the

tape unit number on which an input magnetic tape is placed if such a tape exists. If none of the input programs are on magnetic tape, the INPUT operation is unnecessary; in that case, only the NEW operation appears.

Following tape-allocation operations, the computer operator lists a series of ADD operations. Only ADD operations are applicable because no OLD Utility Tape exists on which alterations could be made.

The computer operator specifies the order of programs on the Utility Tape by means of the identifier in the Label position of the operation. The computer operator assigns identifier values 1·0 and 2·0 to the Control Package and the Paper Tape Control Program, respectively; they will always be the first two programs on the Utility Tape. The computer operator determines and assigns other programs in any location order he desires.

(3) *Building the Tape*

The following steps are used in building the tape:

1) The computer operator first *reads* into the computer, presumably from paper tape via the CS-1 Utility Package, the programs needed to construct the Utility Tape. These programs are the Utility Tape Update Program and the File Computer Tape Control Program.

2) The computer operator places the NEW Utility Tape on the appropriate tape unit.

3) The computer operator next places the Instruction Tape in the reader. He then initiates the Update Program which reads in the Instruction Tape and arranges the identifiers in sequential order. The Update Program also checks, during *read-in*, for various errors and fault conditions and enables the Flexowriter to type out the error. A listing of errors, together with remedial instructions appears in Subsection 5.E "Error and Information Outputs." If no errors occur, the updating process continues.

4) The Update Program then transfers the 24-word blockette of instructions that make up the internal Bootstrap of the Control Package to the appropriate position on the Utility Tape.

5) With Key 3 set, the Update Program establishes and clears the area reserved

for storage of the Utility Tape Directory. The Key-3 setting is applicable only during construction of the *first* Utility Tape. If Key 3 is not set, a fault condition will occur.

6) The Update Program next constructs the directory from the Instruction Tape and transfers it to the Utility Tape.

7) After the directory is completed, the process of writing programs onto the Utility Tape begins. The Update Program checks the directory to find the name of the first program to be transferred to the Utility Tape. It also determines from the Instruction Tape whether the program is on paper tape or magnetic tape and whether the format is relative or octal. The Flexowriter then types out a set of instructions to inform the operator to load either the paper tape or the magnetic tape that contains the designated program on the appropriate input device.

(a) *Paper Tape Loading*

The Update Program reads instructions from paper tape, assembles them into 24-word blockettes in the prescribed format, and stores the assembled blockettes in memory beginning at address 12000. When it has completed this process, the Update Program initiates the File Computer Tape Control Program which transfers the blockettes to the Utility Tape.

Each paper tape program repeats the above process. Several tapes can be combined into one program provided they all are in the same format. Key 2 must be set for loading paper tapes. This suppresses placing a blockette of end-of-file codes on the magnetic tape until Key 2 is released.

(b) *Magnetic Tape Loading*

The computer operator puts the magnetic tape that contains the designated program on the tape unit specified for input tapes. The Update Program searches for and transfers the program from the input tape to core storage and then to the Utility Tape.

(c) *Directory Print-out*

After the Utility Tape is completely written, the Flexowriter types an instruction for the computer operator to connect the High Speed Printer. When the computer is started, the Update Program searches out the header blockette of each program on the Utility Tape. The High Speed Printer prints the contents of each header blockette as it is found.

B. *REVISION OF A UTILITY TAPE*

Construction of a revised Utility Tape from an OLD Utility Tape is somewhat different than construction of the initial Utility Tape. In addition to the requirements listed for the Initial Tape in Subsection 5.A, a tape unit that contains the OLD Utility Tape is required. Programs needed to perform the actual updating can be read into the computer memory according to procedures described in Subsection 5.C. Revision of a Utility Tape is illustrated in Figure E1-9.

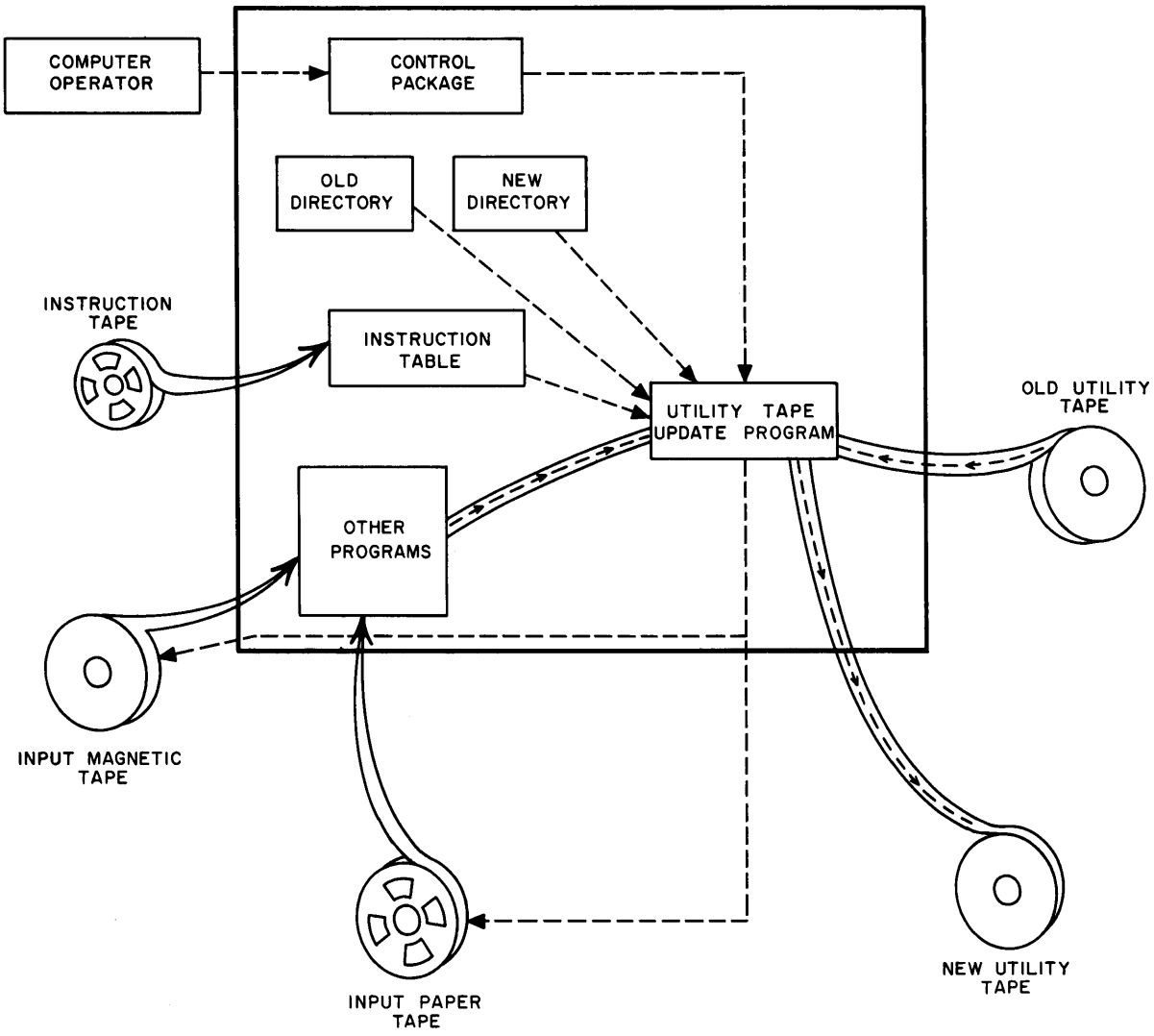


Figure E1-9. Revision of Utility Tape

(1) *The Instruction Tape*

Allocation operators needed for updating a Utility Tape are:

1) NEW, with the tape unit number as the V_0 operand
2) OLD, with the tape unit number as the V_0 operand
3) INPUT (if an input magnetic tape exists), with a V_0 operand giving the tape unit number.

The computer operator uses any or all of the repertoire of action operations, INSERT, ADD, DELETE, UPDATE, and MOVE, in any combination and in any order. He may use any action operator as often as he desires with a combined total of 30 operations permitted. These operations are discussed in Subsection 3.

A sample Instruction Tape appears in Figure E1-8. Figure E1-10 shows magnetic tape updating that results from the sample Instruction Tape.

(2) *Building the New Tape*

The following steps are used in building the new tape:

1) The computer operator records the OLD and NEW Utility Tapes and the assigned units on the Instruction Tape. He then initiates the Update Program which *reads in* the Instruction Tape and checks for several types of errors. The Flexowriter types out any errors detected. The computer operator corrects these errors and then reinitiates the Update Program.

2) The Update Program examines the Instruction Tape and sets indicators in a table (Table of Instructions). These indicators tell which programs are to be deleted from the tape and where insertions are to be made.

3) The Update Program constructs the NEW Utility Tape Directory from the table of indicators, the OLD Utility Tape Directory, and the Table of Instructions.

4) The Update Program transfers the blockette of internal Bootstrap instructions (for AN/USQ-17 Unit Computer) and the NEW updated directory (six blockettes) to the proper location for inclusion in the Control Package on the NEW Utility Tape.

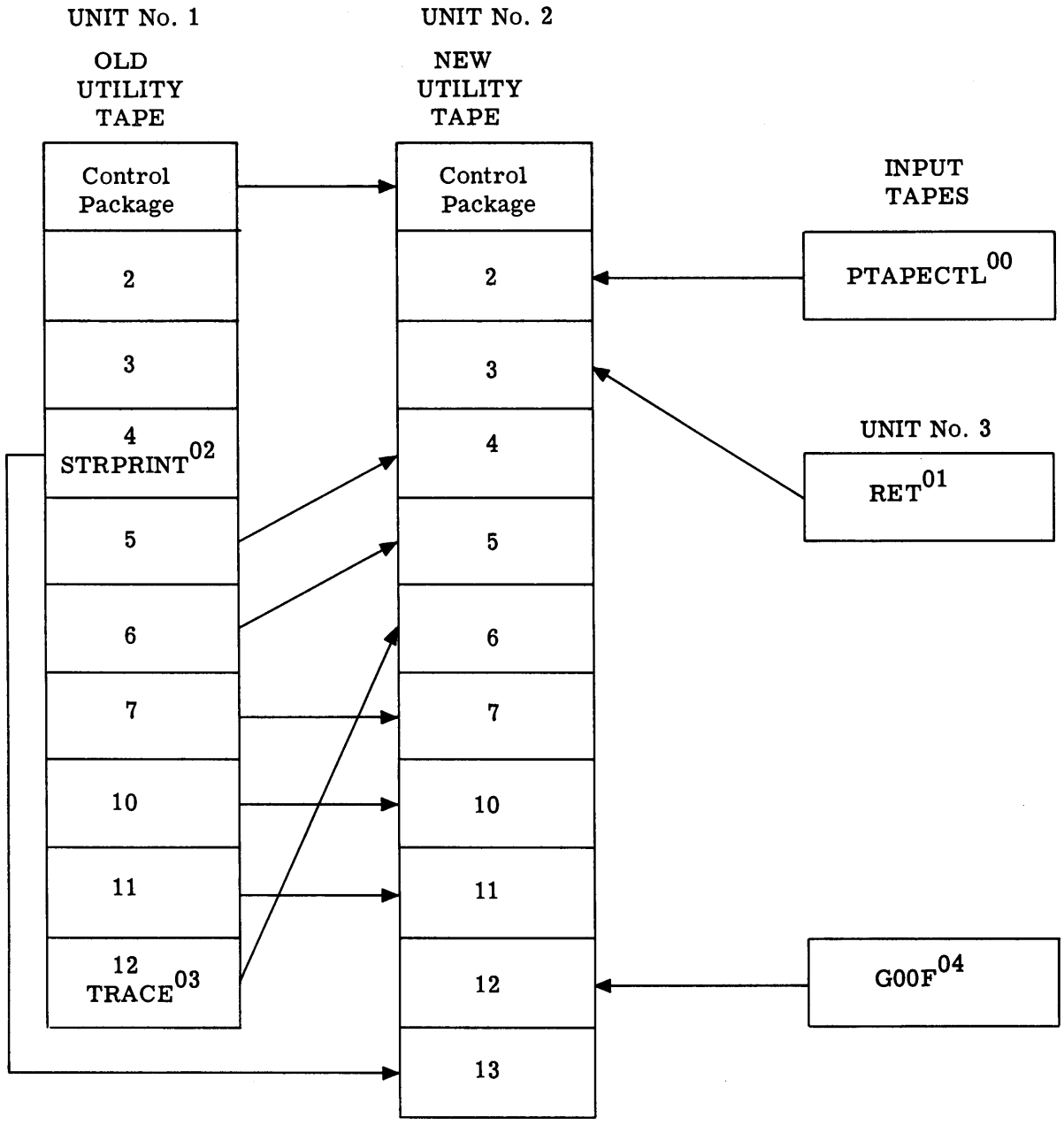


Figure E1-10. Updating Process Resulting from Use of Sample Instruction Tape

5) The table of indicators and OLD Utility Tape Directory then supply the information to complete the writing of the NEW Utility Tape. The Update Program directs the transfer of the first two programs (the Control Package and Paper Tape Control Program) from the OLD Utility Tape to the NEW Utility Tape. It bypasses the internal Bootstrap (for AN/USQ-17 Unit Computer) and the directory and begins transfer of the Control Package to the segment immediately following the directory.

6) Transfer of programs from the OLD Utility Tape to the NEW Utility Tape continues until the Update Program reaches a point where an input tape is required. The program directs the Flexowriter to type out instructions telling the computer operator which tape to load and whether it is paper tape or magnetic tape. The computer operator loads the requested program and the Update Program continues according to the paper tape or magnetic tape loading process described for the Initial Utility Tape. Each input tape is requested as needed. No action by the computer operator is required for programs on the OLD Utility Tape; they are transferred by program control in the sequence prescribed by the internal tables until a program on another input tape is to be inserted or added.

7) When the NEW Utility Tape is completely written, it is rewound. The Flexowriter types a request to connect the High Speed Printer.

(3) *Directory Print*

After a NEW Utility Tape has been constructed, the Update Program automatically initiates a complete print-out of the Directory of Routines. All information in the 24-word header blockette of each program on the NEW Utility Tape is printed out by the High Speed Printer. This is not the same as the Directory Print described on either Page E1-21 or Page E1-36 of this report.

The computer operator connects the High Speed Printer and initiates program action. The program searches out the header blockettes on the tape and directs print-out of their contents on the High Speed Printer. Figure E1-1 shows a sample of a typical header blockette.

C. *CONTROL PACKAGE*

Once the Utility Tape is constructed, it is under control of the Control Package. Programs on the Utility Tape are brought into the computer under the control of programs in the Control Package. Since the Control Package is itself a program on the Utility Tape, a special Bootstrap routine initiates its transfer into the computer (see Subsection 6.B). The Control Package is illustrated in Figure E1-11.

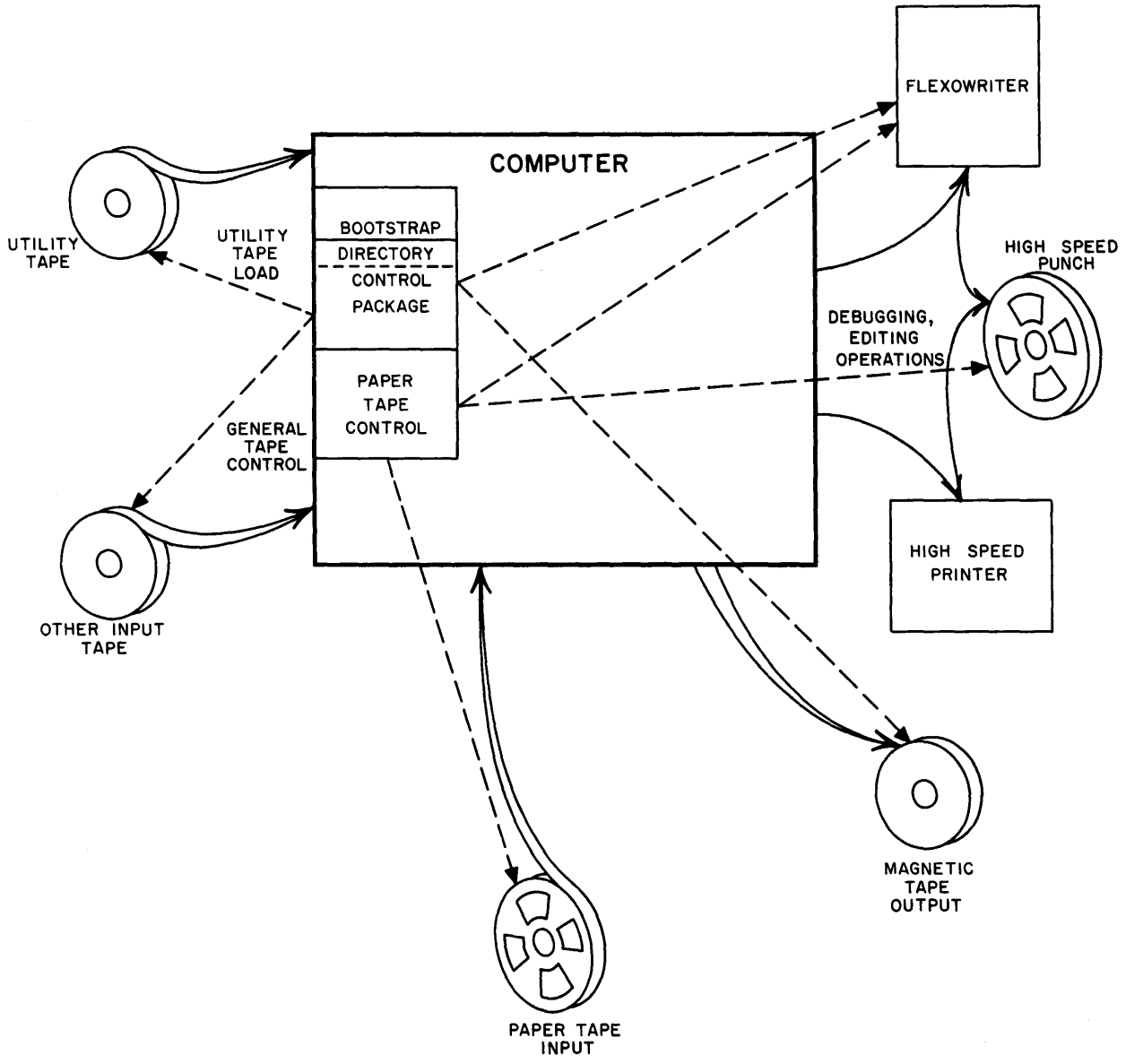


Figure E1-11. Control Package
(Showing also Paper Tape Control Program)

(1) *Bootstrap Blockette* (for AN/USQ-17 Unit Computer)

The Bootstrap, which is entered manually at the console, loads a 24-word blockette of instructions into core-storage locations 00032 to 00061. This blockette constitutes another internal Bootstrap which completes *read-in* of the Control Package into locations 00062 to approximately 02600. This includes the current Utility Tape Directory in core-storage locations 00062 to 00261 (see Figure E1-12).

(2) *Bootstrap Tape* (for AN/USQ-20 Unit Computer)

The Bootstrap tape is read into the computer by means of the wired-memory Bootstrap for paper tape. This tape constitutes a tape read-in program which automatically reads the Control Package, the Utility Tape Directory, and the Paper Tape Control Program into memory locations 00140 to 03000 (see Figure E1-12).

(3) *The Control Word*

The computer operator uses B-register-7 entry to indicate which Control Package function is to be performed. Table E1-3 shows the B-register-7 entries and describes the functions of the Control Package. The B-register-7 entry is called the *Control Word*. The 15-bit Control Word is composed of three groups of octal digits; each group is assigned a special identifying function. Figure E1-13 shows the format of the Control Word and the groups of which it is composed.

TABLE E1-3. CONTROL PACKAGE FUNCTIONS

Control Word in B^7	Function
U00CN	Utility Tape Load
00200	Paper Tape Control
U03CN	General Tape Control
00400	Inspect and Change*
00500	Directory Print
00600	Store Q*
70707	Clear Memory
-----	Dump Registers (for AN/USQ-17)

Note: For a more complete description of Control Package Functions, see Subsection 6.B.

*For the AN/USQ-20 Unit Computer, Paper Tape Control Program performs these functions.

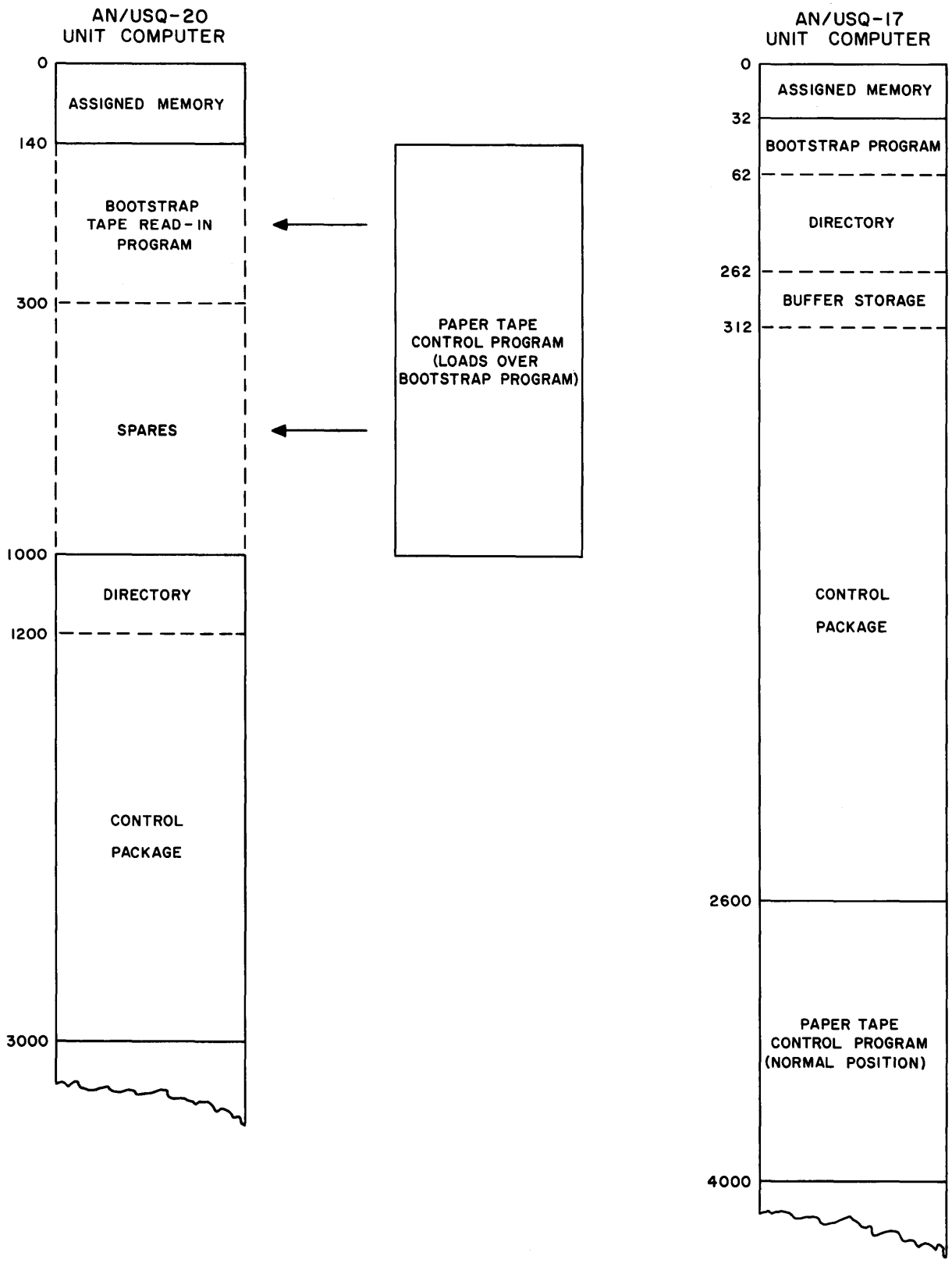


Figure E1-12. Locations of Control Package and Paper Tape Control Program in Core Memory

14 13 12	11 10 09 08 07 06	05 04 03 02 01 00
U Tape Unit Number	FC Function Code	CN Routine Call Number

Figure E1-13. Control Word Format

U - The tape unit number, bit-positions 12 through 14 of B^7, specifies the tape unit that contains a magnetic tape for those operations which involve magnetic tapes. Two operations apply: 1) Utility Tape Load Program and 2) General Tape Control.

FC - The function code, in bit-positions 6 through 11 of B^7, indicates which Control Package routine is to be initiated. Function codes and associated functions are shown in Table E1-3.

CN - The routine call number, in bit-positions 0 through 5 of B^7, specifies which program on a magnetic tape is to be transferred from a magnetic tape to computer memory.

The first three functions listed in Table E1-3 are tape control functions; the remaining five perform diverse debugging and editing activities. The Utility Tape Load Program and General Tape Control Program are the only functions requiring a tape unit number, U, and a call number, CN; they are the only routines involved in transferring programs from magnetic tape to core memory.

(4) *Control Package Functions*

(a) *General*

To execute any of the Control Package functions at the console, the computer operator enters the control word into B^7 and sets the other parameters, if necessary. The Control Package interprets the function, performs the prescribed operation, and stops. It is then ready to accept the next command. At this point, the Flexowriter types the instruction, READY. A complete list of error checks and type-outs for the Control Package appears in Subsection 5.E.

(b) *Individual Function Descriptions*

Utility Tape Load

The Utility Tape Load Program is used to direct the Utility Tape and to read into core storage any of the programs stored on the Utility Tape. The computer operator enters the control word U00CN into B^7.

U - number of the tape unit on which the Utility Tape is stored

00 - function code for Utility Tape Load

CN - call number of the particular program on the Utility Tape

Example: 30011 entered into B^7 means that program No. 11 from the Utility Tape, which is on tape unit No. 3, is to be read into memory.

The Control Package reads and interprets the control word and directs the Utility Tape to the desired program. After positioning the tape at the beginning of this routine, the Utility Tape Load Program reads in the header blockette and checks format code (word 10_8) to determine which type of load it is requested to perform. If the format code is RELAT, indicating a relative routine, the Utility Tape Load Program checks B-register 4 to see if the *base address* at which the program is to be stored has been furnished. If no *base address* is entered in B^4, the computer stops and the program causes the Flexowriter to print, SET BASE ADDRESS IN B^4. The computer operator then enters the base address in B^4 and resumes the program.

Exception: For the AN/USQ-17 Unit Computer, a base address for the Paper Tape Control Program is optional. *Zeros* in B^4 will load this program into core storage, immediately after the Control Package, starting at address 02600.

The Utility Tape Load Program sets a counter using this base address as the starting address of the load. The Flexowriter prints out the name of the program and the *base address.*

Example Print-out:

TAPE TO CORE
TRACE 03 BASE ADDR 50000

The program then reads the next blockette from tape to begin the loading procedure. It interprets each control digit (see Figure E1-3) and modifies accordingly the word in the blockette to which it pertains. It then stores the word at the address given in the counter and advances

the counter by one. Successive blockettes are read in and processed until the end-of-file blockette is reached indicating termination of the load. The Flexowriter then types, READY.

If the format code is OCTAL, the Utility Tape Load Program finds the address in the lower half of the first word of each blockette. It stores the first word at that address and the remaining words in successive addresses until all available words have been stored. The next blockette is processed in a like manner. The procedure continues until the end-of-file blockette is reached; the Flexowriter types READY.

As a program is read in from the Utility Tape a check-sum word is computed. This word is then compared with the check-sum word which is stored for the program on the tape. If the two words do not compare, the Utility Tape is passed back to the beginning of the program and a second *read* operation is attempted. If the *read* is again unsatisfactory, an error print-out occurs and the computer stops. See Supplement A at the end of this section for check-sum format.

Paper Tape Control

The computer operator uses the Paper Tape Control Program to direct performance of all paper tape functions. If the Paper Tape Control Program is not already in core memory, the computer operator must bring it in from the Utility Tape. He does so by placing U0002 in B^7; U is the tape unit on which the Utility Tape is placed, 00 is the function code for Utility Tape Control, and 02 is the program call number for the Paper Tape Control Program. To use the Paper Tape Control Program, the computer operator must perform the following operations:

1) Place 00200 in B^7
2) Enter B-register parameters and/or make Key settings if needed
3) Place the paper tape in the reader for a *read* operation
4) Start the computer.

On recognizing function code 02 in B^7 as a paper tape request, the Control Package yields control to the Paper Tape Control Program Switcher. After the requested operation is completed, control is returned to the Control Package.

Individual operations performed by the Paper Tape Control Program are explained in Subsection 5.D.

General Tape Control

Read from General Tape - By entering control word U03CN in B^7, the computer operator specifies that a program on magnetic tape (other than the Utility Tape) be read into core memory. The General Tape Control activates the tape unit specified by U, directs it to the program specified by CN, and performs the *read-in.* If the program on tape is in *relative* format, the *base address* in core storage must be set in B-register 4. If no entry is made in B^4, the computer stops and causes the Flexowriter to type, SET BASE ADDRESS IN B^4. When this has been done, depressing the HIGH SPEED button will reinitiate action.

The tape is not rewound after a *read* operation. If call number 01 is entered into B-register 7, the record which has its beginning positioned under the *read head* is transferred to core memory; hence, consecutive records may be read into memory with minimum delay.

Check sum computation for General Tape Control is the same as that outlined for Utility Tape Load except, of course, that the program is read from a tape other than the Utility Tape.

Octal Dump onto Magnetic Tape - Dump of a specified area of core memory is given on magnetic tape in octal format only (see Figure E1-2). The required parameters are:

1) Control word in B^7
2) Base address in B^6
3) Last address in B^5
4) Key 2 set.

An internal program check is made to insure that the B^5 entry is larger than the B^6 entry. Dump of memory onto tape is preceded by a header; the operator may assign a name to the dump by manually entering into the A and Q registers the XS-3 coded name. A maximum of 10 characters is permitted. If no name is entered, the title *Memory Dump* is placed in the header blockette.

Data are check-summed as transferred from core memory. The check-sum word is stored as word 10 (octal) in the end-of-file blockette which is the last blockette of the dump.

Excess-Three from Card Load (for AN/USQ-17 Unit Computer) - Excess-three from Card Load is used to load *Flex errata* via magnetic tape. This load is detected by the header format of BEGIN and is terminated by an END code given on a card in the position normally occupied by a Label. The format is as follows:

BEGIN	-	card 1
Instruction No. 1	-	card 2
.		.
.		.
.		.
.		.
Instruction No. N	-	card N+1
END	-	card N+2

There is no check-sum computation associated with this type of load. If more than one Flex program per tape is desired, it is necessary to separate the programs with an end-of-file blockette (24 words of Z's). The B^7 entry must contain a call number as it must for any other program read from tape.

(5) *Miscellaneous Control Package Functions*

(a) *Inspect and Change*

The Inspect and Change function initializes essentially the same performance as the Inspect and Change routine in the CS-1 Utility Package. Setting B^4 to the address to be inspected causes the contents of B^4 address to be entered into registers A and Q. Content of Q may then be changed manually with A serving as a visual aid. Content of Q is then returned to the address from which it was taken. The inspection address need be entered in B^4 only the first time; thereafter, contents of sequential addresses will be brought into A and Q with each successive performance of the Inspect and Change function. Should the computer operator or programmer wish to inspect the contents of some address other than the next sequential address, he may do so by setting the new address in B^4 before returning the content of Q.

(b) *Directory Print*

The Directory Print function permits the computer operator to obtain a Flexowriter print-out of names and call numbers of all programs in the directory. The order of programs is the same in the directory as on the tape.

Example Directory Print-out:

ROUTINE NO. 1
CTRL PKG^{00}
ROUTINE NO. 2, etc.

Entering the control word, 00500, into B^7 and initiating High Speed is all that is required to obtain performance of this function. This is independent of the directory print-out obtained in the Update Program.

(c) *Store Q*

The Store Q function permits the computer operator to load an area of memory with a value which he manually enters into the Q register. He enters the first and last addresses of the area he wishes to fill in B-registers 6 and 5, respectively; he then initiates High Speed. If Q contains all *zeros*, the area is cleared.

(d) *Clear Memory*

By entering the value 70707 in B^7, the computer operator initiates a routine which enables the computer to clear all core storage above the area occupied by the Control Package and the Paper Tape Control Program.

(e) *Dump Registers* (for AN/USQ-17 Unit Computer)

Dump Registers function initiates a routine which directs Flexowriter type-out of the contents of registers A, Q, B^1 through B^7, and C^0 through C^7. No entry is made in B^7. In case of a fault, the computer operator stops the computer with the SEQ STEP key and then clears the *n* designator to extinguish the FAULT light. In all cases, the B Sequence must be set on the console and a Return Jump (65000) to Dump Registers subroutine manually entered in the U register. This is a console debugging feature which enables the computer operator to examine various instruction parameters to determine the fault.

D. *PAPER TAPE CONTROL PROGRAM*

The Paper Tape Control Program is a collection of load/dump routines which provides the computer operator with a ready means of handling paper tape operations. The routines included are:

1) Header print-out and punch-out (for AN/USQ-17 Unit Computer)
2) Relative-Flex load with or without check sum (for AN/USQ-17 Unit Computer)
3) Absolute-Bioctal dump with check sum
4) Relative-Bioctal load with check sum
5) Absolute-Bioctal load with check sum
6) Absolute-Flex load with or without check sum
7) Absolute-Bioctal check read

8) Absolute-Flex dump with check sum (for AN/USQ-20 Unit Computer).

All Paper Tape Control operations are initiated by function code 02 in B-register 7. Other parameters are entered into B registers, and key settings are made as determined by the requested operation. These are listed in Subsection 6.C. A description of each operation follows.

(1) *Header Print-out and Punch-out* (for AN/USQ-17 Unit Computer)

This routine enables the Flexowriter to print out the program header area of CS-1 Compiler output tapes prior to the actual program load into the computer. No header print-out, however, precedes the bioctal check-read operation. Relative-load operations produce a print-out of the *base address* and the *last address* of the load.

A header punch-out stating limits of a dump precedes the bioctal-dump operation. Header print-out and punch-out are bypassed by setting Key 1.

(2) *Relative-Flex Load* (for AN/USQ-17 Unit Computer)

Relative-Flex Load is a subroutine that enables loading of a program into any designated area of core storage. A Flex-coded 9 or 99, preceded and followed by a Flex-coded carriage return, precedes the program to be loaded and identifies it as *relative* input.

The entire program must be written relative to zero. A numeric code preceding each programmed instruction tells the computer how to modify the instruction for storage at a modified address. The modification code precedes the period in the line of an instruction. Code meanings are given in Figure E1-3.

Before loading a *relative* program, the address at which storage is to begin must be entered manually in B^4. Content of B^4 enters automatically into B^1 to effect program-address modification; B^4 remains as the address counter. Value in B^5 or B^6, if code 5 or 6 is used in coding the tape, must be set manually for desired modification.

The Relative-Flex Load protects core-memory addresses less than 04000. If B^4 is less than 04000 or not set at all, the Flexowriter types RESET B^4. To load a program in addresses less than 04000, the computer operator clears register A after print-out and starts the computer. Other limitations of Relative-Flex Load include:

1) B^4 may not be set to 77777

2) Y, the lower half of an instruction, may not be 77777 if it is to be modified

3) Address 77777 may not be formed in Y.

If the 9 or 99 identifier code is incorrect or missing, the Flexowriter types FORMAT ERROR. When the computer incorrectly reads or interprets the program, the Flexowriter types CHECK-SUM ERROR. Following all error print-outs, the Relative-Flex Load stops the computer at the beginning of the subroutine. Table E1-4 illustrates the format of programs using the Relative-Flex Load.

TABLE E1-4. FORMAT OF PROGRAMS USING THE RELATIVE-FLEX LOAD SUBROUTINE

CHARACTERS	EXPLANATION
1. Without Check Sum	
	carriage return
9	start code, no check sum required
	carriage return
*C. X_1, X_2, ----- X_{10}	first data word
C. nX_1, nX_2, ---- nX_{10}	nth data word
7. . .	stop code
	carriage return
2. With Check Sum	
	carriage return
99	start code, check sums required
	carriage return
C. X_1, X_2, ----- X_{10}	first data word
C. nX_1, nX_2, ---- nX_{10}	nth data word
7.	end of data, start check-sum print-out
	carriage return
nnnnn nnnnn	upper check sum
nnnnn nnnnn	lower check sum
7.	stop code
	carriage return

*"C" denotes the code for instruction modification (values 0 - 7).

(3) *Absolute-Bioctal Dump*

To obtain a bioctal dump from a specified core-storage area, the computer operator sets Key 2 and places the *first address* in B^6 and the *last address* in B^5. Output on punched paper tape is: 1) leader, 2) 76 code, 3) initial address, 4) final address, 5) contents of included memory addresses, and 6) check sum (see Supplement B at the end of this section).

(4) *Relative-Bioctal Load*

A relative-bioctal tape may be loaded into any available area of core storage by the use of Relative-Bioctal Load. Program data of a relative-bioctal tape must begin with a 75 code followed by: 1) 10-character computer words, each preceded by a modification digit and 2) the check sum. Modification digits have the same significance to program contents as those described for relative-Flex tapes (see Supplement B).

If a check-sum error occurs, the computer operator may replace the tape in the reader, set the *base address* in B^4, and depress HIGH SPEED for another attempt.

1) AN/USQ-17 Unit Computer - The Relative-Bioctal Load protects core-memory addresses less than 04000; if B^4 is less than 04000 or not set at all, the Flexowriter types RESET B^4. To load a program in addresses less than 04000, the computer operator clears the A register after print-out and starts the computer.

2) AN/USQ-20 Unit Computer - The Relative-Bioctal Load protects core-memory addresses less than 01000.

(5) *Absolute-Bioctal Load*

A Bioctal Program must begin with a 76 (code for bioctal tape). Immediately after the 76, the initial and final addresses of the program are loaded, and the rest of the tape is loaded without additional addresses in the program. Because of the delimiting addresses, the load will stop immediately after an automatic check sum is computed without an end code.

(6) *Absolute-Flex Load*

Absolute-Flex Load enables the computer to accept instructions from Flex-coded punched paper tape. The Absolute Flex must begin with a 45 code (carriage return), followed by a 60 code (8), followed by a 45 code. If check sums are required, two 60 codes follow the initial code. The two 60 codes indicate the appearance of check sums at the end of the Flex tape. A format error is indicated if the first 45 code does not appear.

The computer recognizes only the first 15 octal characters per instruction. The first five are address, and the next ten are the instruction word. All other character codes including notes are ignored. A carriage return identifies the end of an instruction. If an address or instruction does not contain the maximum number of digits, an erroneous load occurs.

(7) *Absolute-Bioctal Check Read*

The computer is able to check an absolute-bioctal punched tape with associated core-memory cells by means of the Absolute-Bioctal Check Read. Objectively, the task assures good bioctal dumps. If the check read detects a discrepancy, the A register holds the word from the tape and the Q register holds the memory word; the computer stops. At this time, the computer operator may check to determine the cause of the discrepancy; he is able to continue the check read from this point.

1) AN/USQ-17 Unit Computer - The Flexowriter prints CONFIRMED for an affirmative check-read operation. The check-read operation is initiated by a Key 3 setting.

2) AN/USQ-20 Unit Computer - The check-read operation is initiated by a Key 1 setting.

(8) *Absolute-Flex Dump* (for AN/USQ-20 Unit Computer)

To obtain a Flex dump from a specified core-storage area, the computer operator sets Key 3 and places the first address in B^6 and the last address in B^5. The "88" format, with the check sum at the end, is the only one dumped. The output on punched paper tape is: 1) leader, 2) beginning codes, 3) each address and each content of the included memory addresses, 4) end code, and 5) check sum.

Note: For the AN/USQ-20 Unit Computer, the Paper Tape Control Program performs the *Inspect and Change* and *Store Q* operations. These routines are described in the preceding subsection.

E. *ERROR AND INFORMATION OUTPUTS*

(1) *Error Outputs*

(a) *Utility Tape Update Program*

The following typewriter-error print-outs apply to the Utility Tape Update Program:

1) INST TAPE FORMAT ERROR

Indicates that the Instruction Tape does not start with a carriage return. Check the Instruction Tape and restart.

2) ID EXCEEDS MAX

Indicates that an identifier has more than two characters on either side of the *point* separator, or an 8 or 9 was found in the identifier.

3) PROG ALREADY ON TAPE

Indicates that a program given with an ADD, INSERT, or UPDATE operator is already on the Utility Tape. This makes it impossible for the same program to appear more than once.

4) PROG NAME EXCEEDS TEN CHAR

Indicates that the name of a program as given on the Instruction Tape exceeded 10 characters.

5) INST TBL OVERFLOW

Indicates that the table used to store instructions *read in* from the Instruction Tape is full. At this point, the program may be continued by depressing the HIGH SPEED button. This will omit the instructions which were not read in. The Instruction Tape may also be checked and the program restarted. The table holds 30 instructions.

6) TAPE UNIT NOT DEFINED

Indicates that one or more operators which define the tape units are not of the correct format or that the unit number specified on the Instruction Tape is zero.

7) ILLEGAL OPERATOR

Indicates that an operator is not one of the five acceptable. Possibly a mispelling or typing error has occurred.

8) PROG NOT ON OLD UTIL TAPE

Indicates that the program given with a MOVE operator is not listed in the directory of the OLD Utility Tape. Possibly a mispelling or typing error has occurred.

9) ID ERROR

Indicates that the identifier given with an operator is not acceptable. May also indicate that the name of the program to be deleted (V_0 operand) given with a DELETE operator does not agree with the identifier given with the DELETE operator. See the description of operators to ascertain what is wrong with the identifier.

10) FORMAT OR MEDIUM ERROR

Indicates that the format of a tape (relative or octal) or the type of tape (magnetic or paper tape) is not given correctly on the Instruction Tape.

11) DIRECTORY EXCEEDED

Indicates that the maximum capacity of the directory has been exceeded. A check will have to be made to determine which programs should be deleted to avoid exceeding the directory limits (63 programs can be listed in the directory).

12) PAPER TAPE FORMAT ERROR

Indicates that the beginning code on a paper tape does not meet one of the accepted formats. Check the paper tape, replace tape in reader when difficulty is corrected, and start computer.

13) PAPER TAPE CHECK-SUM ERROR

Indicates that a paper tape did not read into the computer correctly. Replace this tape in the reader and depress HIGH SPEED.

14) NAME OF PROGRAM NOT FOUND ON MAGTAPE

Indicates that the input magnetic tape which was searched did not contain the program called for. Determine why the program was not found on the magnetic tape, load the correct tape on the unit, and proceed.

15) DUPL ID

Indicates that an identifier appears twice on the Instruction Tape.

16) DUPL PROG NAME

Indicates that the name of a program (V_0 operand) appears twice on the Instruction Tape.

17) UNIT [No.] READING CK SUM ERROR

Indicates that a magnetic tape program did not read into the computer correctly. The program was either on the OLD Utility Tape or on input magnetic tape. Check the tape unit for difficulties.

18) UNIT [No.] WRITING CK SUM ERROR

Indicates that a program cannot be written on the NEW Utility Tape correctly. Check the tape unit for difficulties.

(b) *Control Package*

The following type-outs apply to the Control Package:

1) UTILITY SYSTEM - PROGRAM NOT FOUND

Indicates that a program which is requested from the Utility Tape is not listed correctly or is not included in the directory which is retained in core memory. This print-out also occurs if, after the Utility Tape has been positioned to the requested program, a valid header blockette is not found.

2) ILLEGAL B^7 ENTRY

Indicates that a discrepancy is found in the Control Word which was entered into B-register 7. The discrepancy may be in the function-code or call-number selection.

3) ENTER BASE ADDRESS IN B^4

Indicates that the *base address* was not given for a *relative* load; the computer comes to a *4-stop*.

4) CK SUM ERROR

Indicates that (after two attempts at reading a program from magnetic tape) the program has not been transferred to core memory correctly. This print-out also occurs if (after two writing attempts) a dump of memory onto magnetic tape is not correct.

(2) *Information Type-Outs*

(a) *Utility Tape Update Program*

The following type-outs instruct the computer operator concerning the Utility Tape Update Program:

1) CORRECT ERRORS

Correct errors on the Instruction Tape which were typed out by the Flexowriter and restart the updating.

2) PLACE MAGTAPE [program name] ON TAPE UNIT

Place the tape called for in the type-out on the tape unit designated for input tapes and depress HIGH SPEED.

3) PLACE PAPER TAPE [program name] IN READER

Place the paper tape called for in the type-out in the paper tape reader and set Key 2. Depress HIGH SPEED to load the paper tape.

4) SET KEY 2

Set Key 2 while loading paper tapes.

5) CONNECT PRINTER

Connect the Printer to the computer in preparation for header-blockette print-out. When the Printer is connected, depress HIGH SPEED.

(b) *Control Package*

The following type-outs pertain to the Control Package:

1) ROUTINE NO. nn

(Name of program)

A directory print-out gives this information for each program on the Utility Tape.

2) TAPE TO CORE

(Name of program) BASE ADDRESS (nnnnn)

This print-out precedes the actual transfer of the named program from the Utility Tape into core memory. EXAMPLE: SINX BASE ADDRESS 21050.

3) INSPECT AND CHANGE (for AN/USQ-17 Unit Computer)

This print-out occurs on the initiation of the Inspect and Change function.

4) CLEAR MEMORY

This print-out occurs on the initiation of the Clear Memory function.

5) STORE Q (for AN/USQ-17 Unit Computer)

This print-out occurs on the initiation of the Store Q function.

6) DUMP REGISTERS (for AN/USQ-17 Unit Computer)

This print-out occurs on the initiation of the Dump Registers function.

7) READY

This print-out follows successful completion of each operation requested of the Control Package.

6. OPERATING INSTRUCTIONS

A. *UTILITY TAPE UPDATE PROGRAM*

An Instruction Tape should be typed and the paper tapes and magnetic tapes to be added to the Utility Tape procured. Then proceed as follows:

1) Place OLD Utility Tape on proper tape unit.

2) Place blank tape upon which the NEW Utility Tape will be written on the proper tape unit.

3) Load the Utility Tape Load Program into the computer.

4) Load the Utility Tape Update Program into the computer with the Utility Tape Load Program.

5) Place the Instruction Tape in the paper tape reader.

6) *Master Clear* the computer and set the A Sequence.

7) Place 4000 in P.

8) Set Key 7 if it is desired to change tape unit number assignment.

9) Set Key 3 if no OLD Utility Tape exists — all Instruction Tape operators must be ADD.

10) Depress HIGH SPEED

11) If 7-STOP occurs make tape unit number changes as follows:
B^7 = OLD
B^6 = NEW
B^5 = INPUT

12) If Instruction Tape errors are typed out, correct the Instruction Tape and restart at step 5).

13) Place the input magnetic tapes and paper tapes in the proper input units as they are called for. Set Key 2 for paper tape programs. Release Key 2 when all tapes of one program are loaded.

14) If a Check-Sum Error in *reading* occurs, the following action should be taken:

a) Check tape unit for troubles.

b) Depress HIGH SPEED to reread program from same tape. If it does not read successfully, continue with step c).

c) Place another copy of the tape (if one exists) on the proper unit.

d) If the tape is the Utility Tape, position it past the first block.

e) Depress HIGH SPEED.

f) If the program will not read in from the second tape, a maintenance man should be consulted.

15) If a Check-Sum Error in *writing* is indicated on the NEW Utility Tape, the following action should be taken:

a) Check tape unit for troubles. *Do not move* the tape.

b) Depress HIGH SPEED to rewrite on the same tape. If the write is not accomplished successfully, continue with step c).

c) Place a NEW tape on the proper unit and restart the updating at step 5).

d) If programs cannot be written on the NEW Utility Tape, a maintenance man should be consulted.

16) Connect the Printer when it is called for and start the computer. A complete print-out of each header blockette is printed by the High Speed Printer.

17) After the first directory print-out is finished and the computer has stopped, additional directory print-outs are obtained by again depressing the HIGH SPEED button.

ADDITIONAL OPERATIONAL INSTRUCTIONS

In event of hang-up while the new tape is being written, stop the computer with the OP STEP switch, clear the applicable tape unit, *Master Clear* the computer,

and set the A Sequence. Set P to the address of *Recover^{0} and depress HIGH SPEED.

B. *CONTROL PACKAGE*

(1) *Load Procedure*

(a) *AN/USQ-17 Unit Computer*

1) Place Utility Tape on the tape drive and position to read the first block.

2) Manually enter the following Bootstrap:

00000	13000	40174	C^{7} in C^{2}, C^{2} out
00001	13207	01400	read command
00002	61400	00003	delay
00003	75730	00007	set up buffer
00004	13207	10400	transfer in command
00005	62200	00005	wait on buffer
00006	61000	00032	jump to program
00007	00062	00032	buffer addresses

3) Enter the tape-unit number in B^{7} (binary position where bit-position 0 indicates unit 1; bit-position 1 is unit 2, etc).

4) Depress HIGH SPEED - after the first *4-stop* occurs, delay approximately two seconds until the tape movement ceases and again depress HIGH SPEED.

The above procedure will *read* into the computer the first blockette (24 words) of instructions from the Control Package which is stored as the first program on the Utility Tape. These instructions will load the remainder of the Control Package into the lower part of core memory (addresses 32 to 02600).

(b) *AN/USQ-20 Unit Computer*

1) Place Utility Tape on the tape drive (need not be positioned).

2) Place Bootstrap tape in the paper tape reader.

3) Enter octal tape unit number in B^{7} (normalized right).

4) Depress AUTOMATIC RECOVERY switch.

The wired-memory Bootstrap loads the Tape Read-In Program from the Bootstrap tape and transfers program control to it. This program rewinds the Utility Tape, if necessary, and reads the Control Package and the directory into memory locations 01000 to 03000. The pro-

*From any allocation listings obtained through compilation.

gram then gives program control to the Control Package which automatically loads the Paper Tape Control Program into memory locations 00140 to 01000.

(2) *Operating Procedure*

After the Bootstrap routine has performed its function, all communications with the tape units is accomplished by use of File Computer Tape Control Program which is retained within storage locations allotted the Control Package. To execute a given function, the control word is entered into B-register 7 and the other parameters, if any, are set. The Control Package interprets the function request, performs the prescribed operation, and stops. It is then ready to accept the next command; all registers used for entrance parameters are cleared and register P is preset. Table E1-5 gives more detailed information on Control Package functions than Table E1-3.

(a) *AN/USQ-17 Unit Computer*

A Jump to the Control Package switcher is placed at address 00000. To initiate an individual Control Package function, the computer operator *Master Clears* the computer, sets entrance parameters, and depresses HIGH SPEED. If the function pertains to paper tape, it is to be assumed that the Paper Tape Control Program has been read from the Utility Tape into core memory.

(b) *AN/USQ-20 Unit Computer*

A Jump to the Control Package *switcher* is placed at address 00001. To initiate an individual Control Package function, the computer operator *Master Clears* the computer, sets register P to 00001, sets the entrance parameters, and depresses HIGH SPEED.

C. *PAPER TAPE CONTROL PROGRAM*

(1) *Load Procedure*

(a) *AN/USQ-17 Unit Computer*

The computer operator loads the Paper Tape Control Program into core memory by placing U0002 in B^7; U is the tape unit on which the Utility Tape is placed, 00 is the function code for Utility Tape Control, and 02 is the program call number for Paper Tape Control Program. The computer operator may place a *base address* in B^4. If this is not done, the program will be placed in core storage in its normal position following the Control Package (addresses 02600 to 04000).

TABLE E1-5. CONTROL PACKAGE FUNCTIONS

AN/USQ-17 Unit Computer

Control Word in B^7	*Function*	*Parameters*
U00CN	Utility Tape Load	
	1) Relative Load	Base Address in B^4
	2) Octal Load	
00200	Paper Tape Control	
	1) All Relative Loads	Base Address in B^4
	2) Bioctal and Flex Load	
	3) Bioctal Dump	Key 2 Set
		First Address in B^6
		Last Address in B^5
	4) Check Read	Key 3 Set
	5) Suppress Header Print-out or Punch-out	Key 1 Set
U03CN*	General Tape Control	
	1) Relative Load	Base Address in B^4
	2) Octal Load	
	3) XS-3 From Card Read	
	4) Octal Dump	Tape Must Be Positioned
		Key 2 Set
		XS-3 Name in A and Q
		First Address in B^6
		Last Address in B^5
00400	Inspect and Change	Inspection Address in B^4
00500	Directory Print	

*If the call number, CN, is 01 in the General Tape Control Word, the program read in is the *next* sequential program on the specified tape unit, U. A call number is not required to obtain an Octal Dump from core to tape unit U.

TABLE E1-5. CONTROL PACKAGE FUNCTIONS (Cont.)

Control Word in B^7	*Function*	*Parameters*
00600	Store Q	Word to be Stored in Q First Address in B^6 Last Address in B^5
70707	Clear Memory	
- - -	Dump Registers	Sequence Step if Computer is Running Clear *n* Designator for Fault Set B Sequence 65000 DUMP REG* in U Register

AN/USQ-20 Unit Computer

U00CN	Utility Tape Load	
	1) Relative Load	Base Address in B^4
	2) Octal Load	
00200	Paper Tape Control	
	1) Relative-Bioctal Load	Base Address in B^4
	2) Bioctal and Flex Load	
	3) Bioctal Dump	Key 2 Set First Address in B^6 Last Address in B^5
	4) Check Read	Key 1 Set
	5) Flex Dump	Key 3 Set First Address in B^6 Last Address in B^5
	6) Store Q	Word to be Stored in Q First Address in B^6 Last Address in B^5 Set P to 750
	7) Inspect and Change	Inspection Address in B^4 Set P to 760

*From any allocation listings obtained through compilation

TABLE E1-5. CONTROL PACKAGE FUNCTIONS (Cont.)

Control Word in B^7	*Function*	*Parameters*
U03CN*	General Tape Control	
	1) Relative Load	Base Address in B^4
	2) Octal Load	
	3) Octal Dump	Tape Must be Positioned
		Key 2 Set
		XS-3 name in A and Q
		First Address in B^6
		Last Address in B^5
00500	Directory Print	
70707	Clear Memory	

*If the call number, CN, is 01 in the General Tape Control Word, the program read in is the *next* sequential program on the specified tape unit, U. A call number is not required to obtain an Octal Dump from core to tape unit U.

(b) *AN/USQ-20 Unit Computer*

The Paper Tape Control Program is automatically loaded into core memory as a part of the Bootstrap process. The Bootstrap program, after executing the read-in of the Control Package, sets up the parameters necessary to call the Paper Tape Control Program from the Utility Tape. A Jump to the Control Package effects the load of Paper Tape Control Program into its assigned memory area (addresses 00140 to 01000).

(2) *Operating Procedure*

To use the Paper Tape Control Program, the computer operator performs the following operations:

1) Places 00200 in B^7
2) Enters B register parameters and/or makes key settings if needed (see Table E1-5)
3) Places paper tape in the reader for a *read* operation
4) Starts the computer.

On recognizing External Function code 02 in B^7 as a paper tape request, the Control Package yields control to the Paper Tape Control Program *switcher*. After the requested operation is completed, control is returned to the Control Package.

Exception: For the AN/USQ-20 Unit Computer, *Inspect and Change* and *Store Q* routines are initiated by setting register P.

(3) *Error Detection*

(a) *AN/USQ-17 Unit Computer*

Errors encountered in Paper Tape Control Program are indicated by Flexowriter print-outs. The computer stops.

(b) *AN/USQ-20 Unit Computer*

Errors encountered in Paper Tape Control Program are indicated by registers A and Q set to all ones and the FAULT light on the reader console lighted. The computer jumps into a loop.

7. BLOCK DIAGRAMS

The Utility Tape Update Program described in Subsection 2.D is illustrated by the block diagram in Figure E1-14. This figure is an expansion of the *Write New Utility Tape* portion of Figure E1-5.

Figure E1-15 shows the Control Package flow diagram. Miscellaneous editing and debugging routines which are a part of this package are shown as branches. Any of these branches may be selected by interpretation of a specific Control Word in B-register 7. This is described in Subsection 5.C.

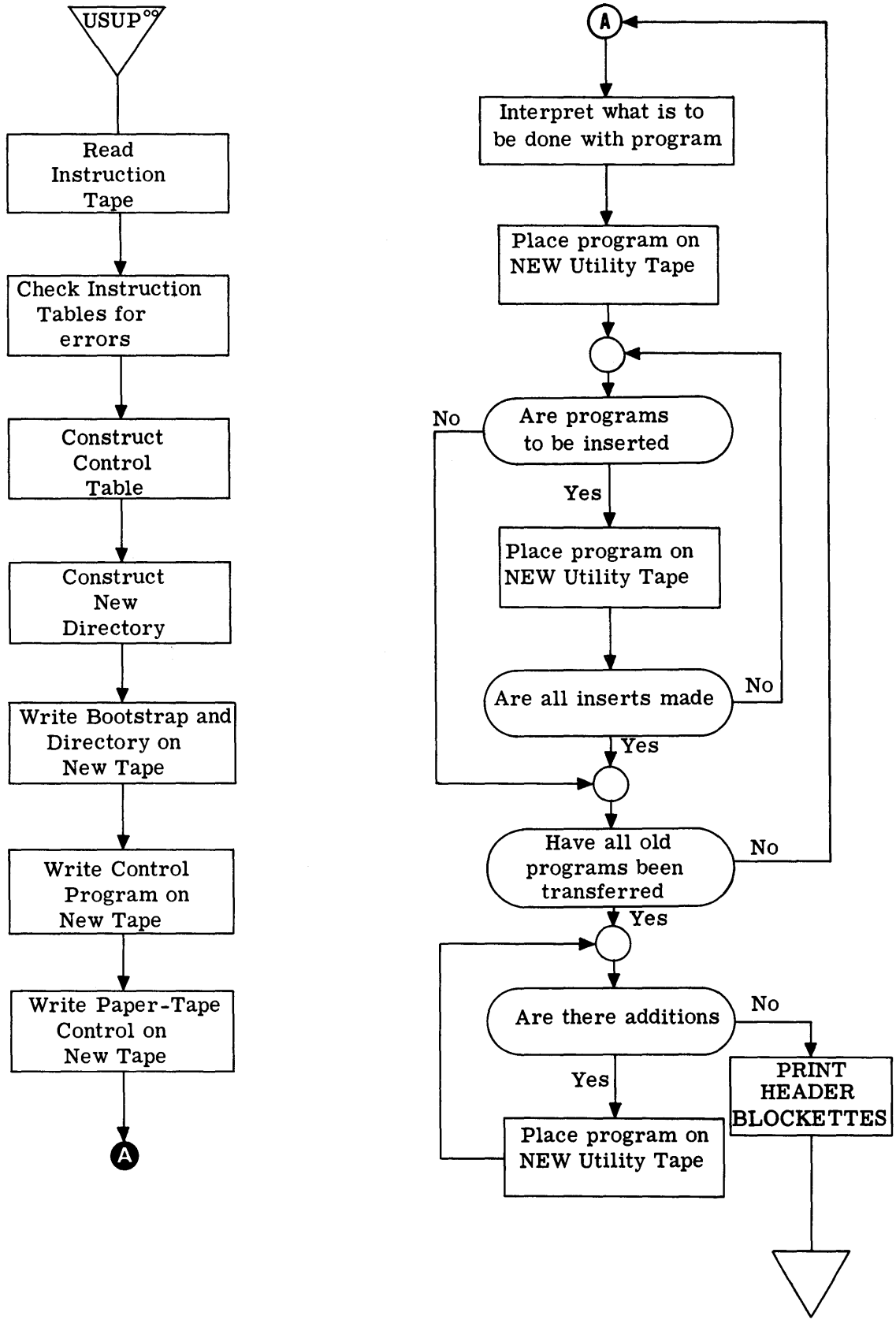
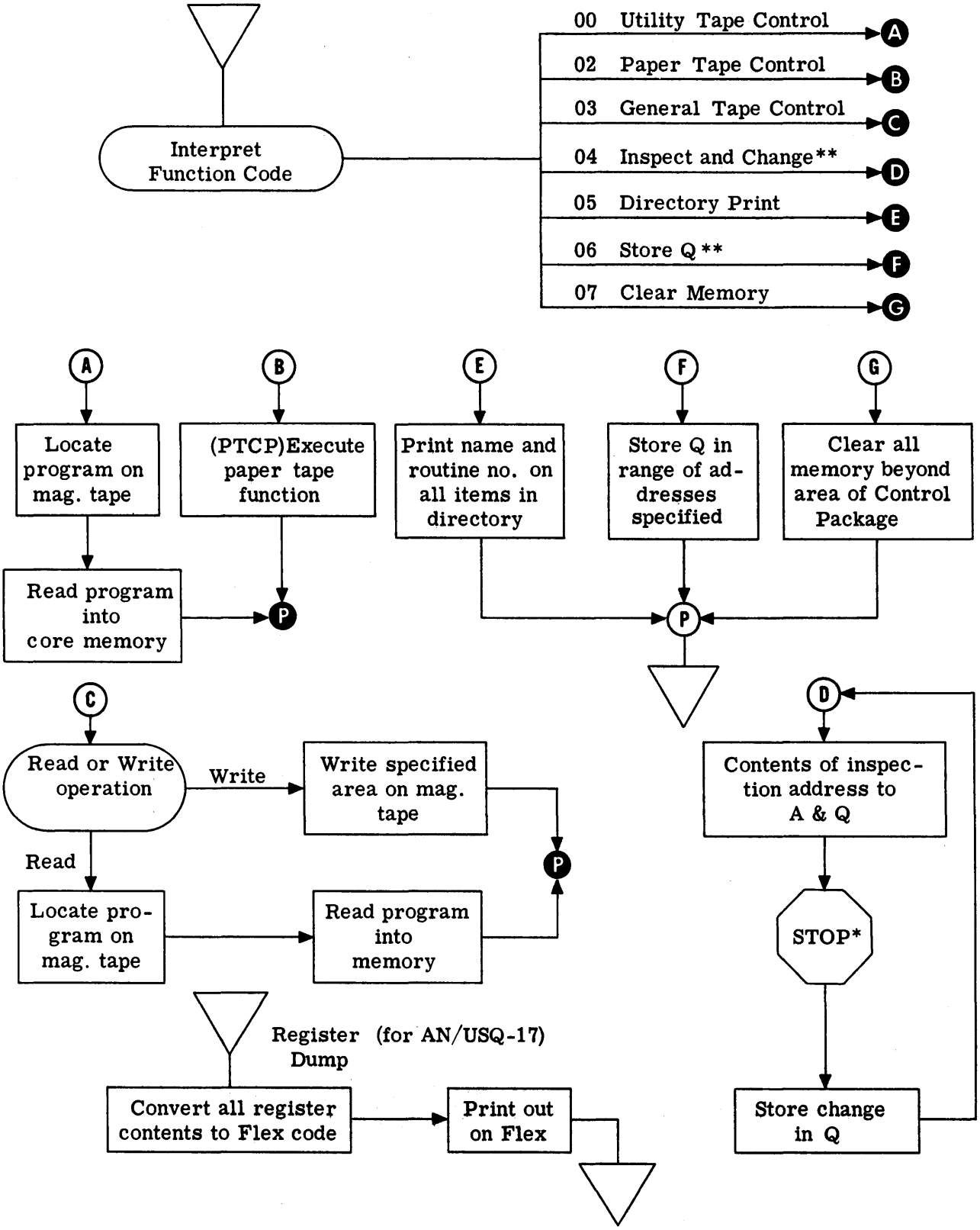


Figure E1-14. Utility Tape Update Program



*Computer operator makes desired change in register Q and restarts computer.

**For the AN/USQ-20 Unit Computer, Paper Tape Control Program performs these functions.

Figure E1-15. Control Package

SUPPLEMENT A

MAGNETIC TAPE OPERATIONS

1. FORMAT FOR CHECK-SUM COMPUTATION ON MAGNETIC TAPE

1) Header blockette not included in computation

2) Characteristic words for both octal and relative are included in computation

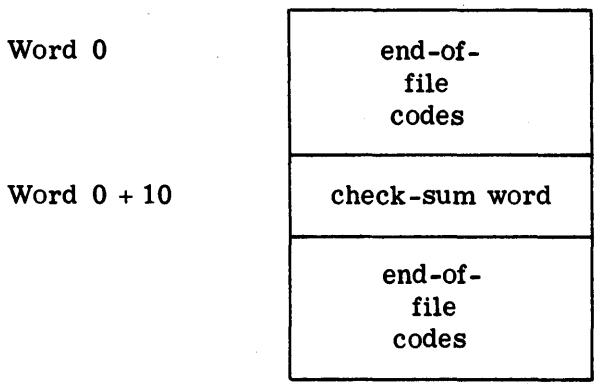
3) End-of-file blockette not included in computation

4) Check sum the buffer area for each blockette of data or instruction words and preserve count until end-of-file is reached

   a) Separate count for upper and lower parts of words

   b) The two are added together and the sum is found in word 0 + 10 of each end-of-file blockette.

End-of-File Blockette



2. READ OPERATION WITH CHECK SUM

1) *Read Forward* if check-sum error occurs, then

2) *Pass Backward* and

3) *Read Forward* again

3. WRITE OPERATION WITH CHECK SUM

1) *Write Forward*; then

2) *Pass Backward* and

3) *Read Forward*; compute check sum but do not store data; if error occurs, then

4) *Pass Backward* and

5) *Read Forward* as before; if error occurs, then

6) *Pass Backward* and

7) *Repeat* the *Write* Operation one time

4. ERROR PROCEDURE

1) Print-out on Flexowriter

2) Check sum as on tape to A register

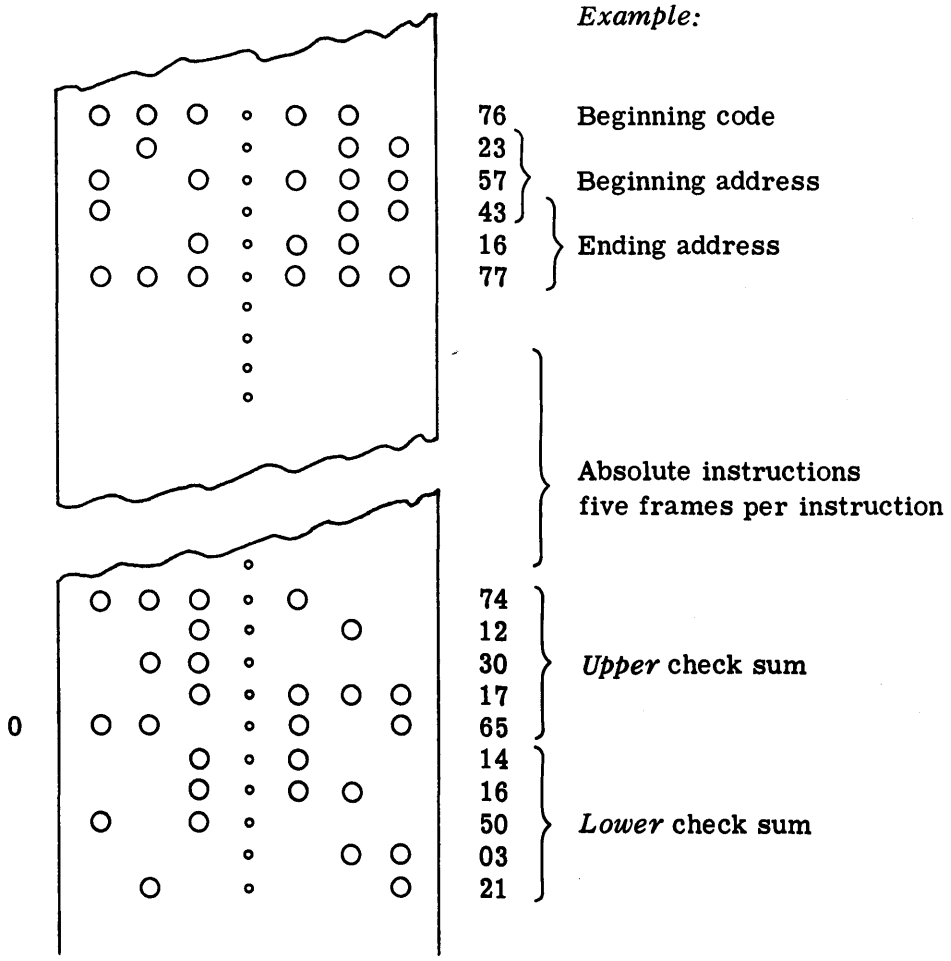
3) Check-sum word as computed to Q register

SUPPLEMENT B

PAPER TAPE FORMAT

1. ABSOLUTE BIOCTAL (Compiler Output)

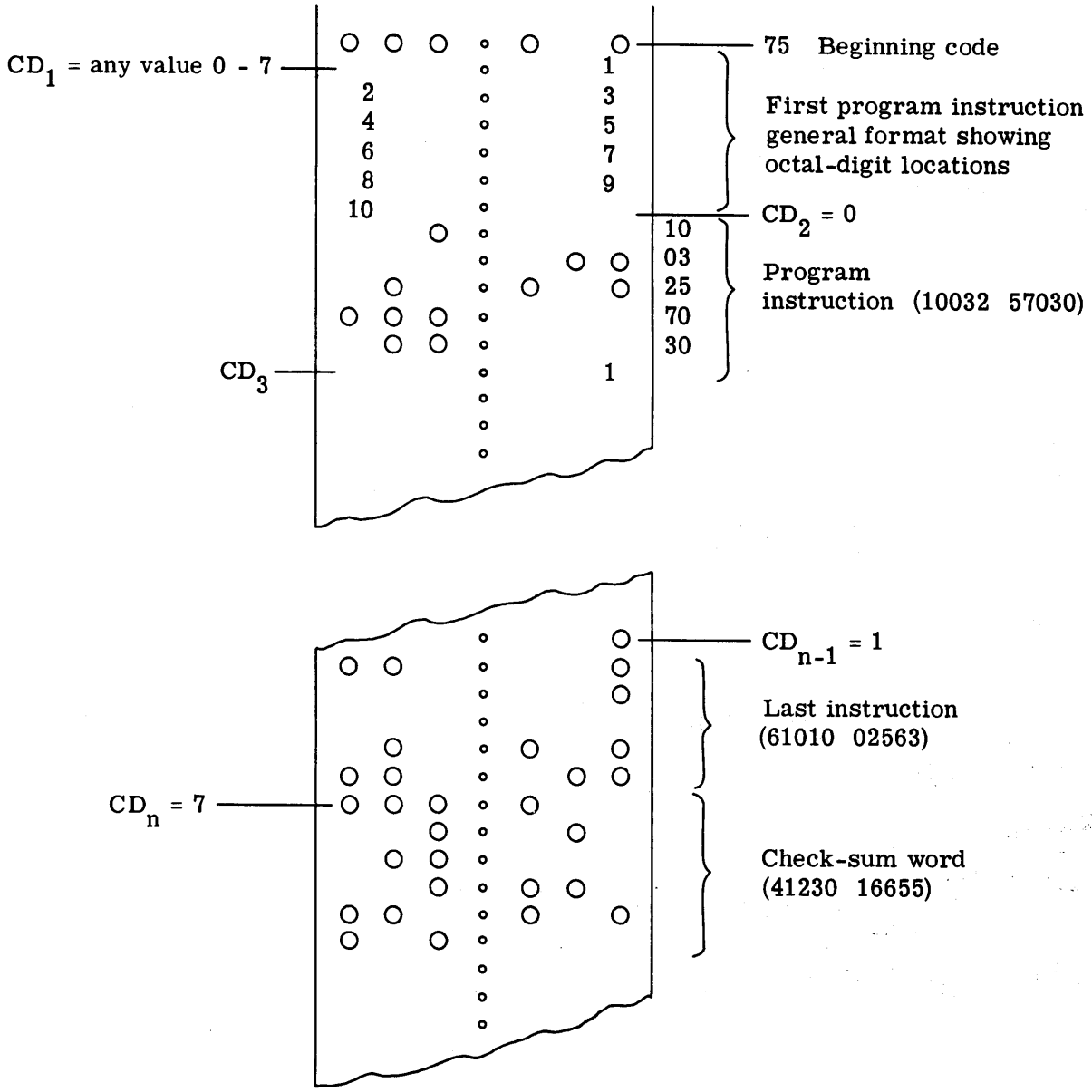
The format is completely bioctal (i.e., each frame of paper tape carries two octal digits). The first frame contains a 76, the next five frames contain the initial and final address, and the succeeding frames contain the absolute instructions (five frames per instruction). The last 10 frames contain check sums for the upper half-words and lower half-words, as computed from memory. Following is an example of a bioctal tape that loads from 23574 to 31677; check sums are 7412301765 for upper half-words and 1416500321 for lower half-words.



2. RELATIVE BIOCTAL (Compiler Output)

The first frame contains a 75; the next frames contain a control digit (0 through 7) followed by 10 octal digits of the instruction words as shown in the example below. A check sum appears at the end. Control Digit values are interpreted as described in Figure E1-3.

Example:



PROGRAMMERS' GUIDE

Addendum

The enclosed material entitled *NTDS Utility System* is to be inserted into RRU document PX 1493, *Programmers' Guide*, as Section E1.

SECTION E2

CS-1 UTILITY PACKAGE

(PAPER TAPE)

CONTENTS

CS-1 UTILITY PACKAGE
(PAPER TAPE)

1. BASIC INFORMATION

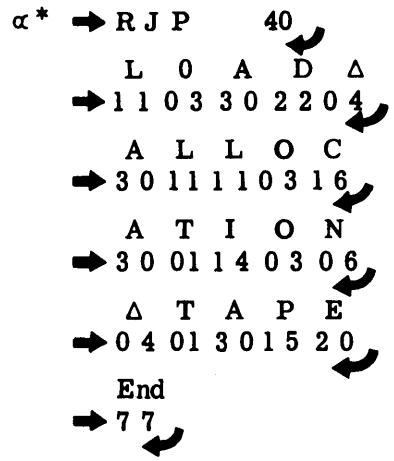
The Utility Package is a group of eight routines conveniently assembled and available on punched paper tape. These eight routines when loaded in the computer afford the programmer and/or operator a stored program which will load and dump subsequent programs or data. In addition, provisions for checking and changing address contents are included. The eight routines are 1) Type-Punch Text, 2) Relative Load, 3) Flex Load, 4) Bioctal Load, 5) Bioctal Dump, 6) Flex Dump, 7) Jump to Switcher, and 8) Inspect and Change. The programmer selects the routine applicable for loading the type of input tape he has, or chooses the type of output he desires by following the operation procedures (Paragraph 3., OPERATION INSTRUCTIONS) outlined in this section.

2. OPERATION DEFINITIONS AND PROGRAM FORMATS

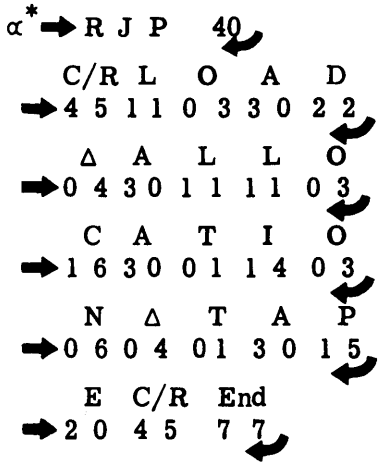
a. LOADING PROCEDURE FOR THE UTILITY PACKAGE. - The Utility Package is available on prepared paper tape in bioctal code. A Bootstrap Routine (see Paragraph 3) is entered manually into the computer to initiate the loading of the Utility Package. The loading process enters the Utility Package into the computer with each routine assigned to specific allocated areas in memory. A Check Sum Routine is included to ensure that the package has been entered properly.

b. TYPE-PUNCH TEXT ROUTINE. - The programmer uses the Type Text or the Punch Text Routines when he desires or needs a message he can read during a program run. He initiates the routines by means of a return jump to the applicable address - 00040 for Type Text or 00046 for Punch Text. The message to be typed or punched follows immediately the return jump operation. The programmer prepares the text of the message in legitimate 6-bit Flexowriter codes at the rate of five characters per word. All codes, including space codes and carriage return codes wherever desired, *must* be entered. A 77 code terminates the

message; it follows the last coded symbol of the message. The code 00 is ignored. Following is an example of a Type Text Routine coded in mnemonics:



To ensure that the typed text begins on a new line, the programmer places a carriage return code (Flexcode 45) at the beginning or at the beginning and end of his coded message. The above would then be:



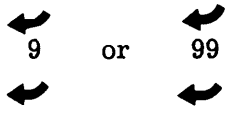
The programmer writes only the numeric Flexowriter codes. These can be found in Table 3-7, Page 3-154, Volume I of the NTDS UC Manual, OA( ) /AN/SSQ-25. The alphabetic characters show only the arrangement of the numeric codes.

c. RELATIVE LOAD ROUTINE. - This routine enables the computer to load a program into any programmer designated area of core storage. A flex-coded 9 or 99, preceded and followed by a flex-coded carriage return, precedes the program to be loaded and identifies it as relative input. Symbolically, either of the following appears before the first program

---

* α = current program address

instruction:



The entire program must be written relative to zero. A numeric code preceding each programmed instruction tells the computer how to modify the instruction for storage at a modified address. The modification code precedes the period in the line of an instruction. The code values have the following meanings relative to the *instruction:*

0) Make no modification
1) Modify the lower half with the initial address
2) Modify the upper half with the initial address
3) Modify the upper and lower halves with the initial address
4) Disable the reader — perform the instruction — if control is not lost, enable the reader and continue loading
5) Modify the lower half with B^5
6) Modify the lower half with B^6
7) Stop, or check sums following (depending on the initial code of 9 or 99):

    9. Relative format, no check sum required.
    99. Relative format, check sums required following a code 7 (end code)

Before loading a relative program, the operator enters in B^7 manually at the console, the address at which the storage is to begin. The content of B^7 enters automatically into B^1 to effect the program address modification, and into B^4 to establish the address counter. The value for B^5 or B^6, if code 5 or 6 is used in coding the tape, must be set manually for the desired modification.

The Relative Load Routine is assigned to memory locations 175-347. If B^7 is less than 350 or not set at all, the Flexowriter will type "set B^7". Although this routine accepts a subsequent program at address 350 and higher, a setting from 350-777 inclusive destroys the other Utility Package Routines stored in those memory locations. Since the Relative Load Routine uses part of these locations, the operator should enter a value of 1000 or over. Other limitations of Relative Load include:

1) B^7 may not be set to 77777.
2) Y, the lower half of an instruction, may not be 77777 if it is to be modified.
3) An address may not be formed in Y which equals 77777.

If the 9 or 99 identifier code is incorrect or missing, the Flexowriter types "format error." Incorrect reading and/or interpretation of the program by the computer causes the Flexowriter to type "check sum error". Following all error printouts, the routine sets the computer in a "run" condition. Table E2-1 illustrates the format of programs using the Relative Load Routine.

TABLE E2-1. FORMAT OF PROGRAMS USING RELATIVE LOAD ROUTINE

CHARACTERS	*EXPLANATION*
1. Without Check Sum.	
↵	carriage return
9	start code, no check sum required
↵	carriage return
*C. $X_1, X_2, - - - - - X_{10}$	first data word
C. $nX_1, nX_2, - - - - nX_{10}$	nth data word
7.	stop code
↵	carriage return
2. With check sum.	
↵	carriage return
99	start code, check sums required
↵	carriage return
C. $X_1, X_2, - - - - - X_{10}$	first data word
C. $nX_1, nX_2, - - - - - nX_{10}$	nth data word
7.	end of data, start check sum printout
↵	carriage return
- - - - - - - - - -	upper check sum
- - - - - - - - - -	lower check sum
7.	stop code
↵	carriage return

* "C" denotes the code for instruction modification (values 0-7).

d. FLEX LOAD ROUTINE. - This routine prepares the computer to accept instructions or programs from flex-coded punched paper tape. The program must begin in flex format with the code for a carriage return (45) followed by the code for 8 (60), followed by the code for carriage return (45). If check sums are required, 88 or 89, coded in flex, must follow the first carriage return. The 88 indicates the appearance of the computed check sums at the end of the flex-load tape. The code for 89 initiates provisions for storage of the computed sums before the actual load takes place; in this case an automatic subtraction from the stored sums occurs as the instructions are loaded. In either case a loading error causes the Flexowriter to type out "check sum error". A program tape which does not begin with a carriage return (45) followed by an 8 (60) will cause the Flexowriter to type "format error".

The computer recognizes 15 characters per instruction. The first five are the address, and the next ten are the instruction word. A carriage return identifies the end of an instruction. If an address or instruction does not contain the maximum number of digits, an erroneous load will occur. The partially assembled word will appear as shifted to the right of an address (the first five digits). This condition *will not* be indicated by a Flexowriter printout.

Only 15 characters are recognized between carriage returns. The computer ignores any digits in excess of 15 (e.g. notes). It also ignores codes other than carriage return, numbers 0 through 7, and double periods; therefore, it is important that the programmer follow each instruction with a carriage return. Failure to do so will result in the subsequent instruction(s) being ignored by the computer. Until another carriage return is encountered, the information is lost. Table E2-2 illustrates the format of programs using the Flex-Load Routine.

e. BIOCTAL LOAD ROUTINE. - A Bioctal program must begin with a 76 (code for Bioctal tape). Immediately after the 76 the initial and final addresses of the program are loaded and the rest of the tape is loaded, *without* additional addresses in the program. Because of the delimiting addresses, the load will stop immediately after an automatic check sum is completed without an end code.

If the beginning code is missing or in error, the Flexowriter types out "format error". A check sum deviation is typed out on the Flexowriter as in other routines. Following all error print-outs, the routine sets the computer in the run condition.

f. BIOCTAL DUMP ROUTINE. - The Bioctal Dump Routine is initiated manually by the operator at the console. The operation instructions appear in paragraph 3,f, of this section. The output, on punched paper tape, is the 76 code followed by 1) the initial and final addresses

TABLE E2-2. FORMAT OF PROGRAMS USING FLEX-LOAD ROUTINE

CHARACTERS	*EXPLANATION*
1. With no check sum.	
↵	carriage return
8	start code, no check sum required
↵	carriage return
Y_1 - - - - Y_5 X_1 - - - - X_{10}	first data word
nY_1 - - - - nY_5 nX_1 - - - - nX_{10}	last data word
. .	end code.
2. With check sum at end.	
↵	carriage return
88	start code, check sum required at end of load
↵	carriage return
Y_1 - - - - Y_5 X_1 - - - - X_{10}	first data word
nY_1 - - - - nY_5 nX_1 - - - - nX_{10}	last data word
. .	end code
_____	check sum, upper
_____	check sum, lower
3. With check sum at beginning.	
↵	carriage return
89	Start code, check sum required at beginning of load
↵	carriage return
_____	check sum, upper
_____	check sum, lower
Y_1 - - - - Y_5 X_1 - - - - X_{10}	first data word
nY_1 - - - - nY_5 nX_1 - - - - nX_{10}	last data word
. .	end code

of the program being dumped, 2) the contents of the included memory addresses, and 3) the check sums. Output does not indicate each individual address in memory from which the content is dumped, other than the initial and final addresses.

g. FLEX DUMP ROUTINE. - The computer operator initiates manually the Flex Dump Routine at the console. Operating instructions appear in paragraph 3 of this section. The "88" format (with the check sum at the end) is the only one dumped. The output on punched paper tape includes both the addresses and the contents of the memory locations being dumped.

h. JUMP TO SWITCHER ROUTINE. - This routine analyzes subsequent inputs and determines which of the three loads or two dumps is to be used, and jumps to the valid routine address. This jump is inserted in address 00000. This allows a subsequent program tape to be loaded or punched from a "Master Clear" without setting the P register.

i. INSPECT AND CHANGE ROUTINE. - This routine enables a display in the Q register of the contents of any address in memory. At that time, Q may be changed manually. The content of Q is returned to the address from which it was taken. Before Q is returned, the programmer may set B^4 to the next address he wishes to inspect. If no alteration of B^4 contents is made, the contents of the next chronological memory location will be automatically placed in Q as the first inspected location is returned. Paragraph 3 following, contains console operating instructions for this procedure.

3. OPERATION INSTRUCTIONS

a. TO USE UTILITY PACKAGE

1) Manually enter the following bootstrap program:

Address			
00000	12	200	00004
00001	63	100	00001
00002	17	140	00000
00003	03	000	00006
00004	72	200	00001
00005	14	031	00032
00006	72	100	00000
00007	61	000	00745

2) Master Clear

3) Start reader

a. Turn switch to "on".

b. Press "Enable" button.

4) Position Utility package tape on first information frame

5) Set B^1 = 00754

6) Set C^0 = 00001

7) Start

The above steps will load the Utility Package in the lower part of memory (00032-00776). A jump to the switcher (61000 00724) is placed at address 00000.

b. TYPE TEXT

1) Return jump to 00040

c. PUNCH TEXT

1) Return jump to 00046

d. FLEX OR BIOCTAL LOAD

1) Master Clear
2) Place tape in reader
3) Start

e. RELATIVE LOAD

1) Master Clear
2) Set beginning address in B^7
3) Place tape in reader
4) Start

f. FLEX DUMP

1) Master Clear
2) Set Key 3
3) Set beginning address in B^7
4) Set last address in B^6
5) If zero suppression is desired, set Key 1
6) Start

g. BIOCTAL DUMP

1) Master Clear
2) Set Key 2
3) Set beginning address in B^7
4) Set last address in B^6
5) Start

h. INSPECT AND CHANGE

1) Set p to 00020
2) Place address to be inspected in B^4
3) Start

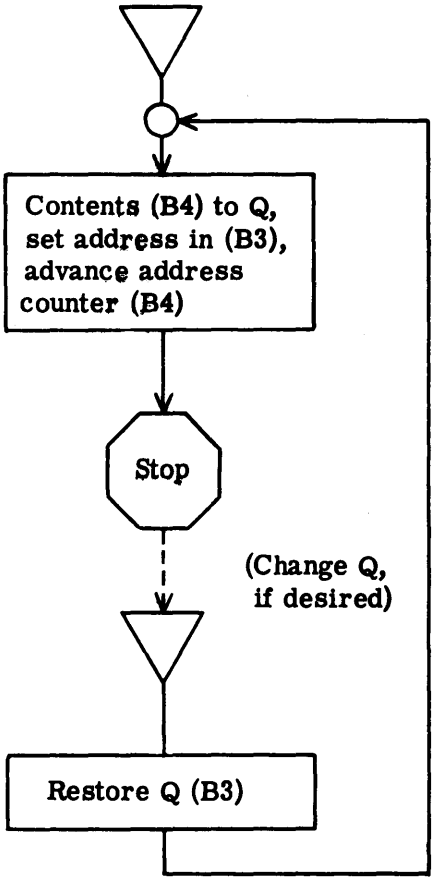
   Contents of the address will appear in Q with the address in B^3

4) Make desired changes in Q
5) Start

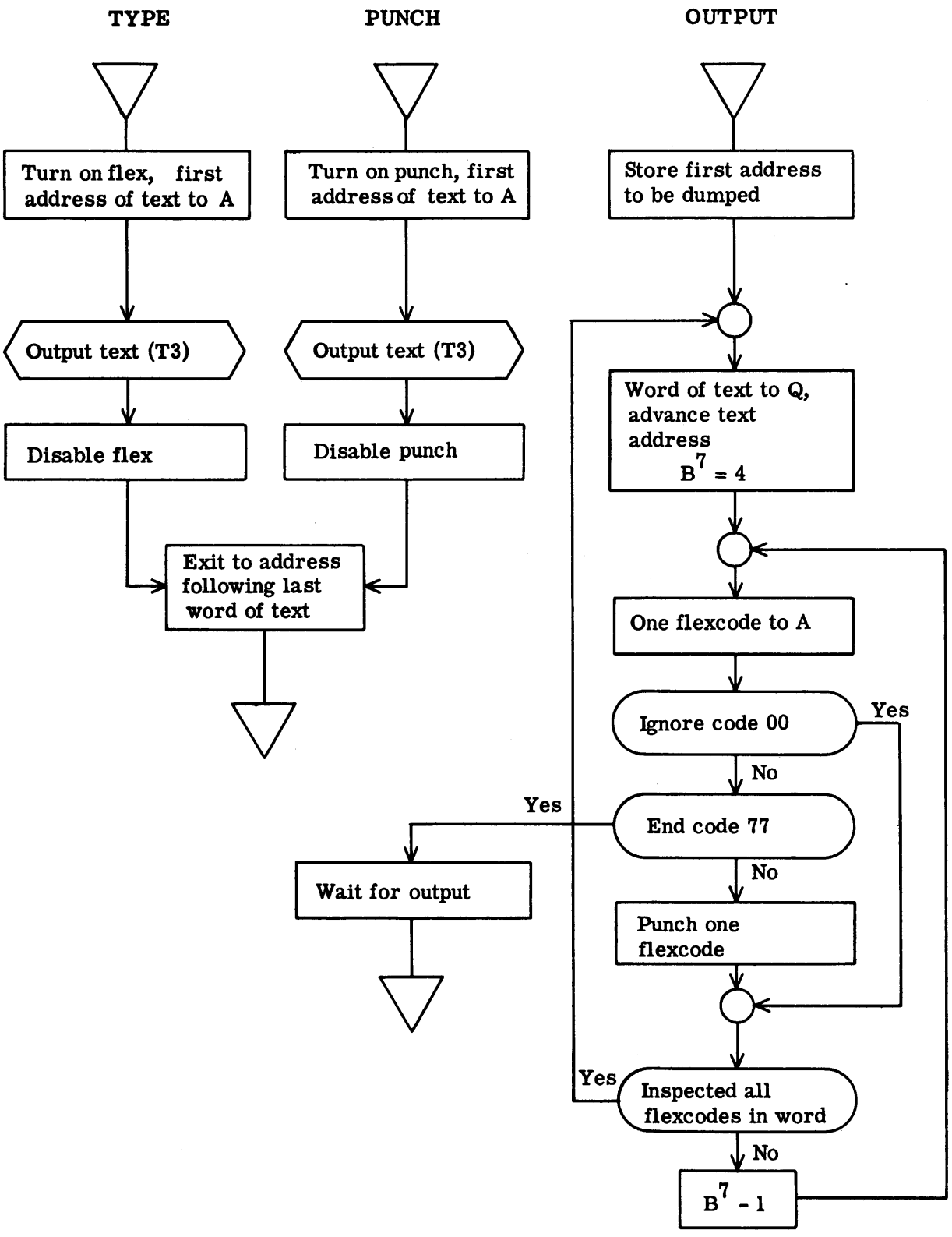
   Q will be restored and the contents of the next address will appear in Q unless a new address has been specified in B^4

4. FLOW CHARTS

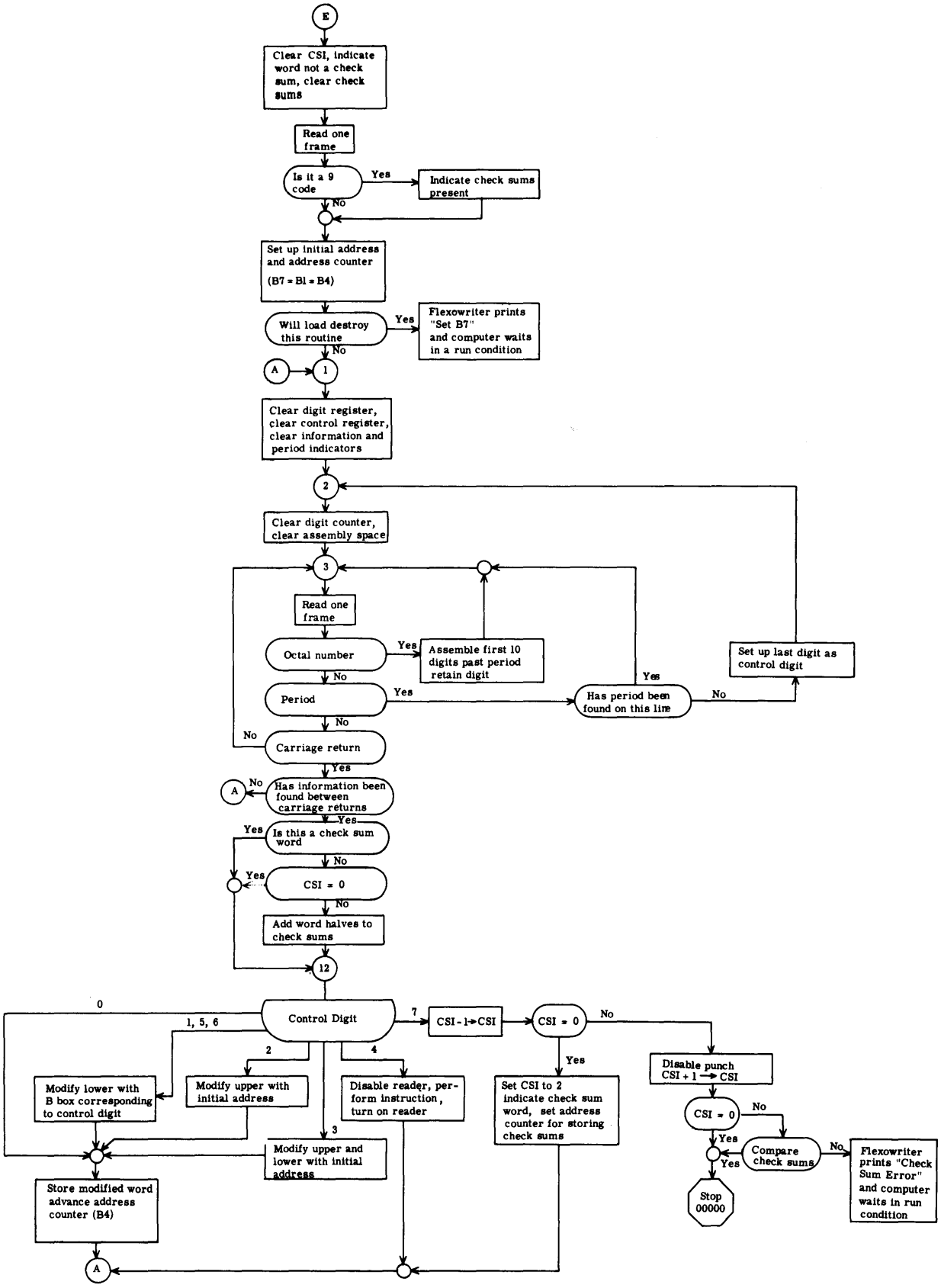
INSPT0:	Inspect and Change
TYPETX:	Type Text
PCHTX:	Punch Text
TXOUT0:	Output Text
RTCL0:	Relative Load
LIB0:	Flex Load
LIB25:	Bioctal Load
CSUP0:	Check Sum Utility Package
BIOCTALD0:	Bioctal Dump
FLEXD0:	Flex Dump
SWITCH0:	Switcher



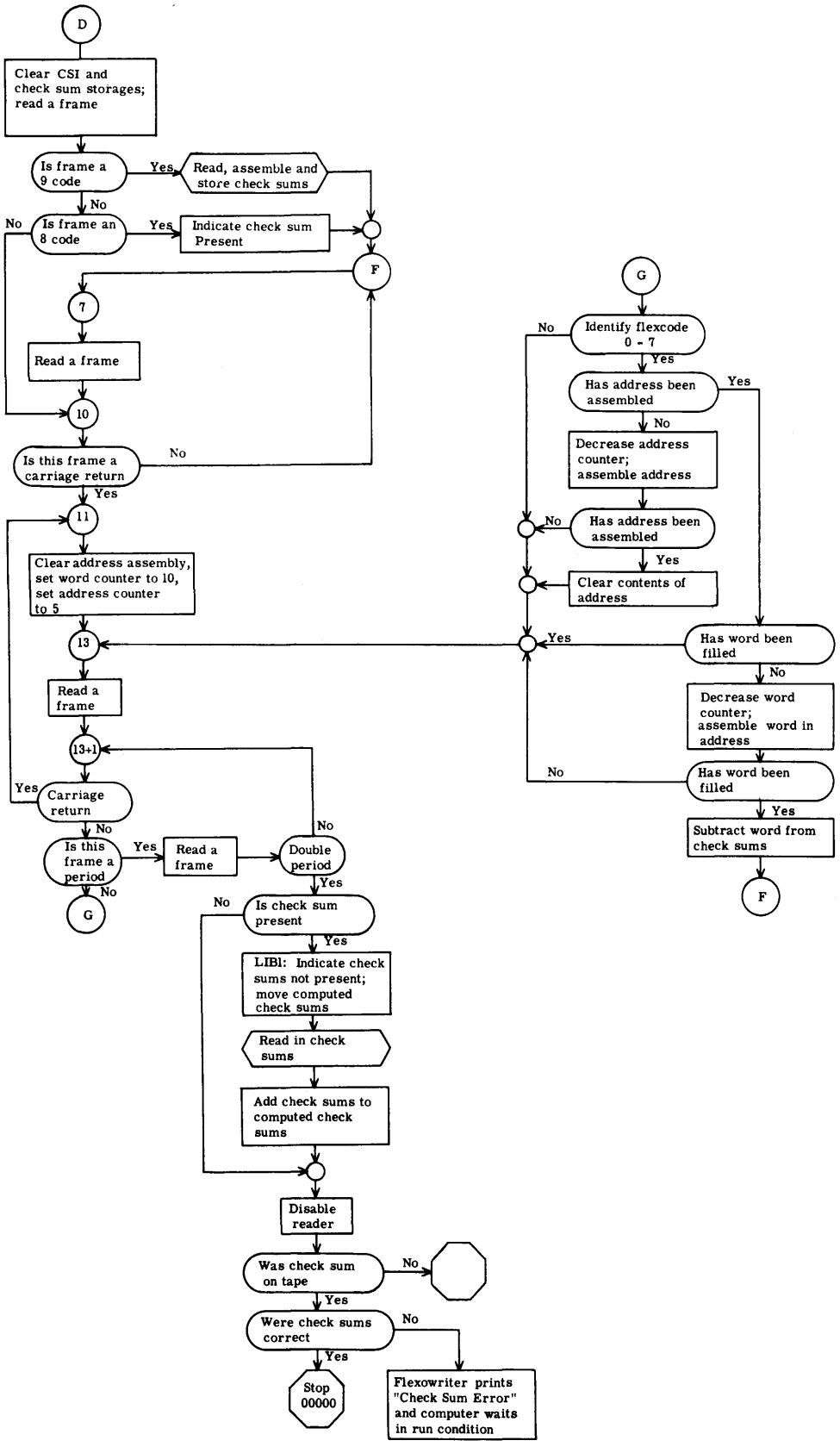
INSPT0: Inspect and Change



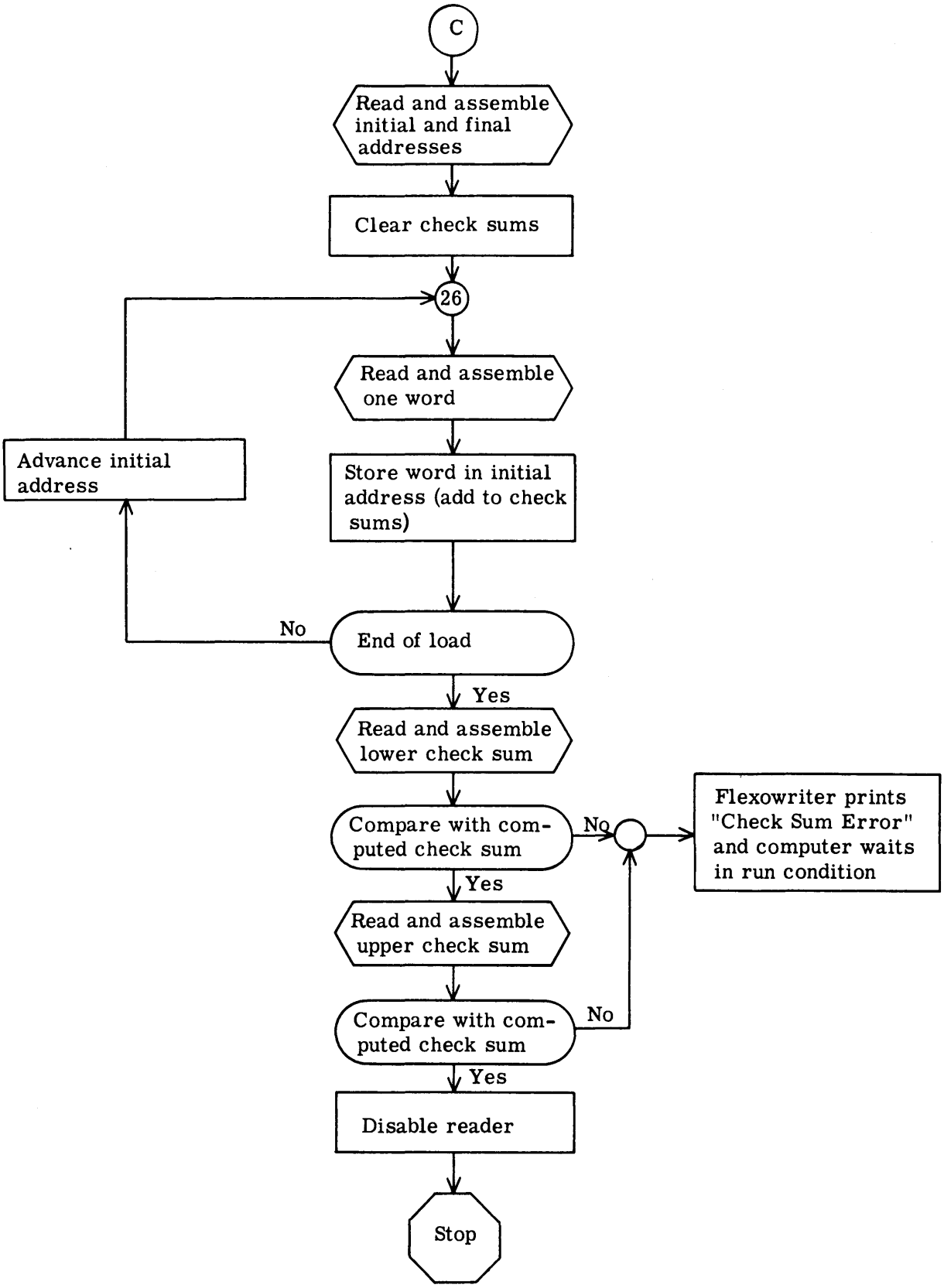
TYPETX: Type Text
PCHTX: Punch Text
TXOUT0: Output Text



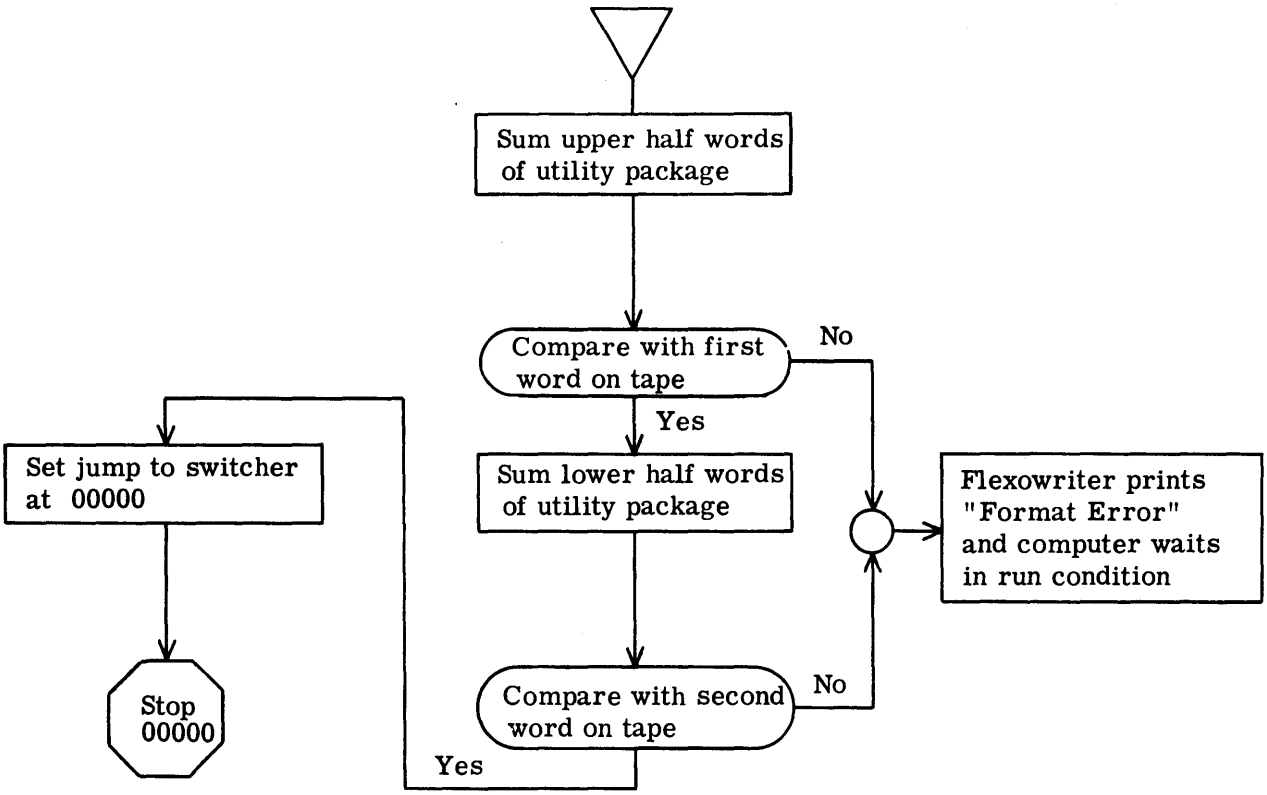
RTCL0: Relative Load



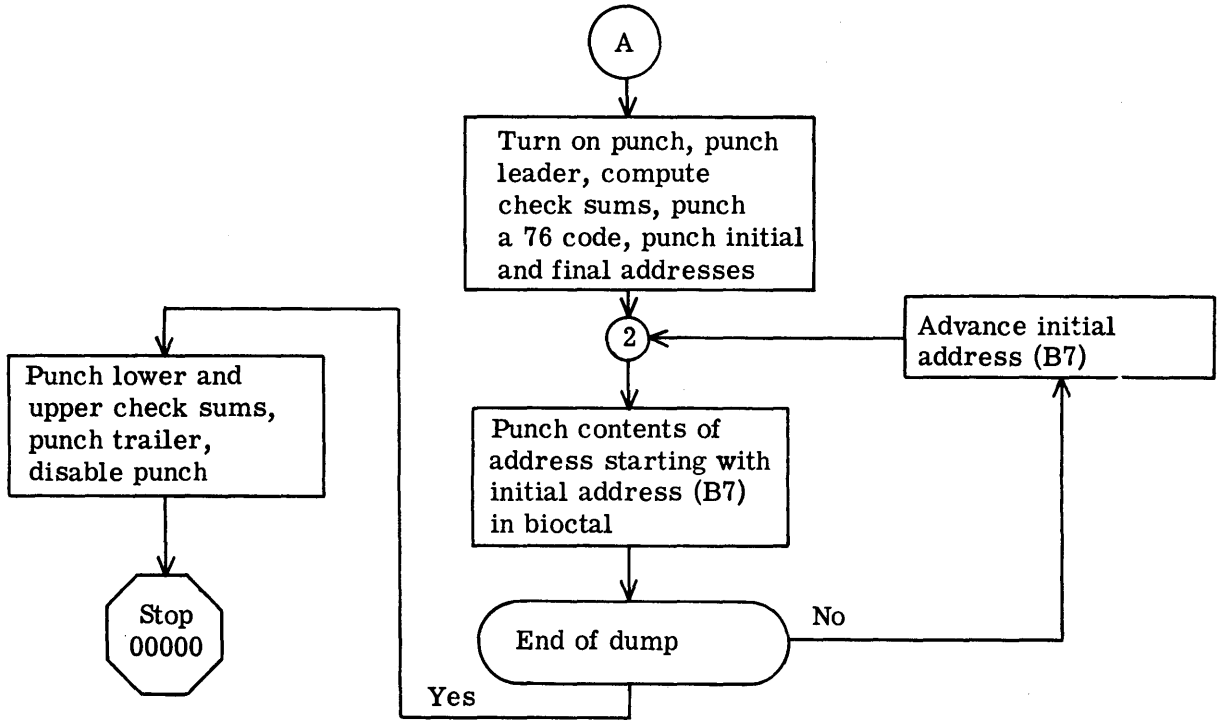
LIB0: Flex Load



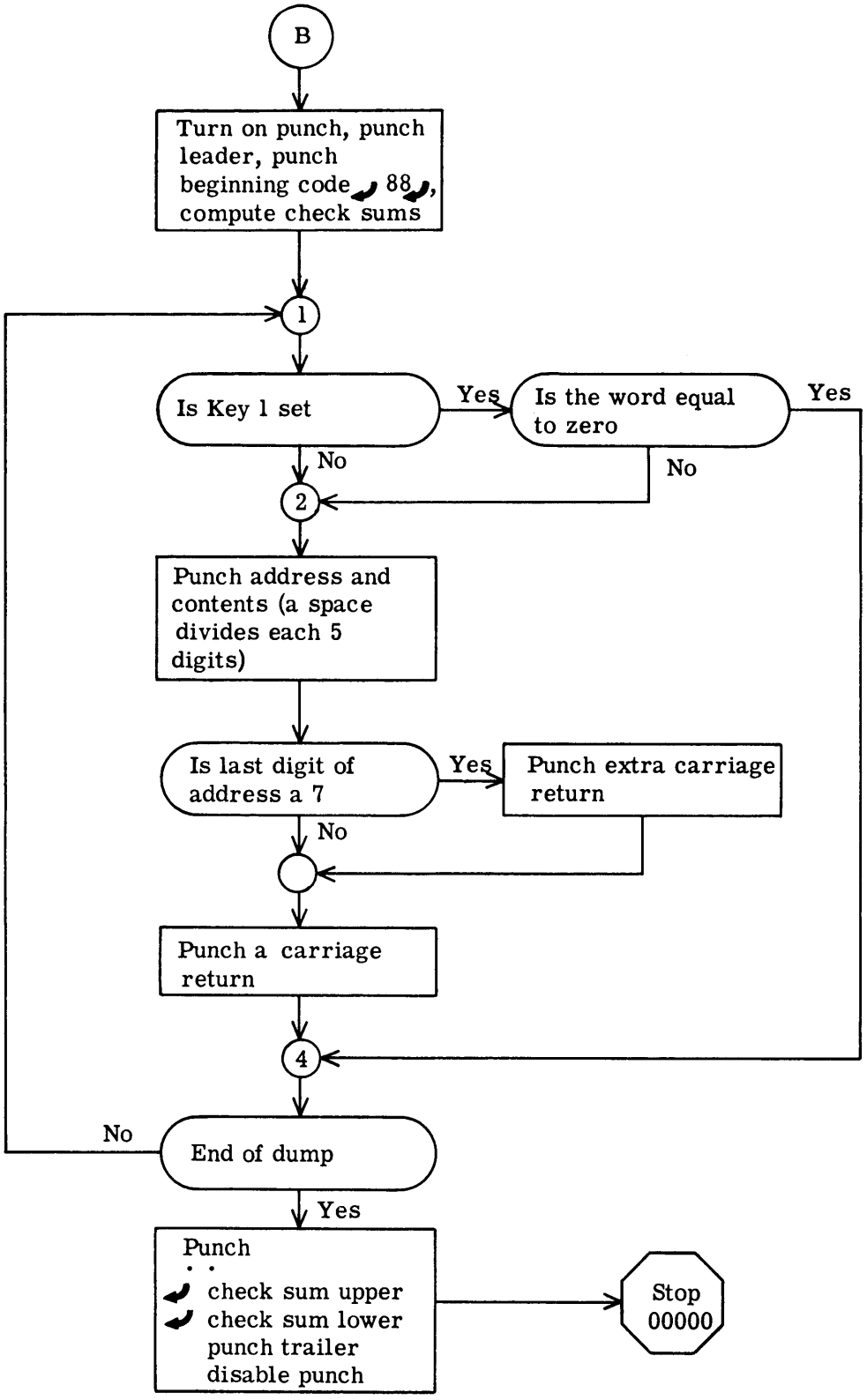
LIB 25: Bioctal Load



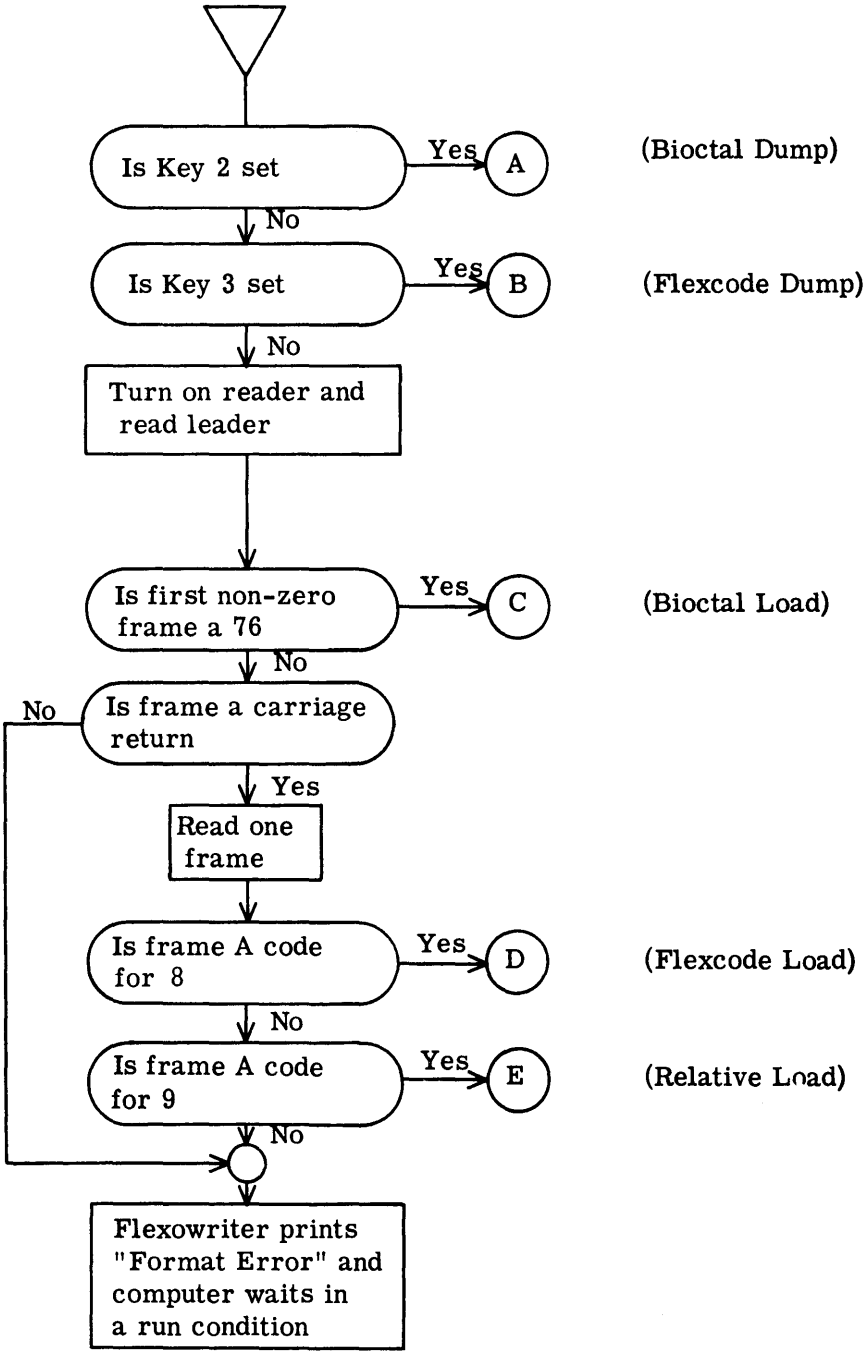
CSUP0: Check Sum Utility Package



BIOCTALD0: Bioctal Dump



FLEXD0: Flex Dump



SWITCH0: Switcher

```
        UTILITY PACKAGE -

        INSPECT AND CHANGE
00032   INSPTO          JP•INSPT1
        INSPT1          STR•Q•W(B3)             STORE WORD
0034                    ENT•Q•W(B4)             INSPECT AND CHANGE - START
                        ENT•B3•B4               ADDRESS TO B3
                        ENT•B4•B4+1             ADVANCE ADDRESS OF WORD TO INSPECT
                        JP•INSPTO•STOP          STOP

00040   TYPE TEXT
        TYPETX          JP•0                    EXIT - ADDRESS OF CODES
                        RJP•SFLEXO

                        ENT•A•L(TYPETX)         TURN ON FLEX-OUTPUT
                        RJP•TXOUTO              TYPE TEXT TO 77
                        ENT•C3•10016            DISABLE FLEX
                        JP•L(TXOUT1)STOP5       TO INSTRUCTION FOLLOWING CODES

        PUNCH TEXT
00046   PCHTX           JP•0                    EXIT - ADDRESS OF CODES
                        RJP•STPCH
                        ENT•A•L(PCHTX)          TURN ON PUNCH
                        RJP•TXOUTO              PUNCH TEXT TO 77

                        ENT•C3•10026            DISABLE PUNCH
                        JP•L(TXOUT1)STOP5       TO INSTRUCTION FOLLOWING CODES

        OUTPUT TEXT
        TXOUTO          JP•0                    EXIT
                        STR•A•L(TXOUT1)
        TXOUT1          ENT•Q•W(0)              FIRST WORD OF CODES TO Q
                        RPL•Y+1•W(TXOUT1)       ADVANCE ADDRESS OF CODES
                        ENT•B7•4                SET COUNTER
        TXOUT2          CL•A

                        LSH•AQ•6•ANOT           FIRST CODE TO A
                        JP•TXOUT4               IGNORE 00
                        COM•A•77•YMORE
                        JP•TXOUT5               77 FOUND
        TXOUT3          JP•TXOUT3•C1FULL        OUTPUT CODE
                        ENT•C1•A
        TXOUT4          BJP•B7•TXOUT2           5 CODES   NO
                        JP•TXOUT1               YES - CONTINUE

        TXOUT5          JP•TXOUT5•C1FULL        DELAY
                        JP•TXOUTO

        TURN ON(OFF) EQUIPMENT
00074   TORO            JP• 0                   TURN ON READER
                        SEL-CHAN•C1IN1•C3OUT
                        EX-FCT•1•33
                        EX-FCT•1•31
                        JP•TORO
```

```
                TURN ON (OFF) EQUIPMENT (cont.)
00112 STPCH     JP•0                        START PUNCH
                SEL-CHAN•C1OUT•C3OUT
                EX-FCT•1•23
                EX-FCT•1•11
                EX-FCT•1•22
                JP•STPCH
      SFLEX0    JP•0                        ENABLE FLEX OUTPUT
                SEL-CHAN•C1OUT•C3OUT
                ENT•C3•10013
                RPT•200
      SFLEX1    ENT•B0•0
                STR•C3•L(SFLEX1)
                EX-FCT•1•26
                EX-FCT•1•12
                JP•SFLEX0

                ERROR PRINT OUT
00156 TFE0      RJP•TYPETX                  FORMAT ERROR
                45472•60312
                07300•10420
                12120•31277
      TFE1      JP•TFE1

      TCSE0     RJP•TYPETX                  CHECK SUM ERROR
                45471•60520
                16360•42434
                07042•01212
                03125•74577

      TCSE1     JP•TCSE1

      SETB7     RJP•TYPETX                  SET B7
                45472•42001
                04237•25777
      SETB70    JP•SETB70

                RELATIVE LOAD
00175 RTCL0     ENT•Q•77                    MASK TO Q
                CL•L(CSI0)                  CLEAR CHECK SUM INDICATOR
                CL•W(STOR1)                 CLEAR CHECK SUM STORAGE

                CL•W(STOR2)
                ENT•A•RTCL11                SWITCH TO TEST FOR CHECK SUM
                STR•A•L(RTCL10)
                RJP•READ0                   FRAME TO A
                COM•MASK•33•ANOT
                RPL•Y+1•L(CSI0)             SECOND 9 FOUND
      DR0       STR•B7•A
      CR0       ENT•B1•A

      CSI0      ENT•B4•A
                SUB•A•RTCL21+2•APOS
```

```
00212                   JP•SETB7                SET B7
        RTCL1           CL•L(DRO)               CLEAR DIGIT STORAGE
                        CL•L(CRO)               CLEAR CONTROL DIGIT STORAGE
                        ENT•A•RTCL1             SWITCH TO RESTART FOLLOWING A CR
                        STR•A•L(VO)
                        ENT•B2•RTCL7            SWITCH TO USE A PERIOD TO CONTROL

        RTCL2           ENT•B3•0
                        CL•W(STORO)             CLEAR ASSEMBLY SPACE
        RTCL3           RJP•READO
                        ENT•B7•0                SET DIGIT TO ZERO
                        ENT•Q•77
        RTCL4           COM•MASK•U(RTCL13+B7)ANOT       CHECK FOR NUMBER CODE 07
                        JP•RTCL5                FOUND A NUMBER
                        BSK•B7•7

                        JP•RTCL4
                        COM•MASK•42•ANOT
        UO              JP•B2
                        COM•MASK•45•ANOT
        VO              JP•0
                        JP•RTCL3
        RTCL5           STR•B7•L(DRO)           STORE NUMBER
                        JP•RTCL6•KEY1           JUMP IF LAST TEN DIGITS TO BE USED

                        BSK•B3•12               ASSEMBLE FIRST 10 DIGITS
                        JP•RTCL6
                        ENT•B3•12               WORD ASSEMBLED - RESET B3
                        JP•RTCL3
        RTCL6           ENT•A•W(STORO)          ASSEMBLE DIGITS
                        LSH•A•3
                        ENT•Q•7
                        SEL•SU•L(DRO)

                        STR•A•W(STORO)
                        A-SET•L(VO)RTCL10       SWITCH TO STORE WORD AFTER CR FOUND
                        JP•RTCL3
        RTCL7           A-MOVE•1•L(DRO)L(CRO)   USE NUMBER AS CONTROL DIGIT
                        ENT•B2•RTCL3            SWITCH TO IGNORE SUCCEEDING PERIODS
                        JP•RTCL2                TO CLEAR ASSEMBLY AND COUNTER
        RTCL10          JP•RTCL1                SWITCH
        RTCL11          ENT•A•L(CSIO)           TEST CHECK SUM INDICATOR

                        JP•RTCL12•AZERO         NO CHECK SUM
                        ENT•Q•U(STORO)
                        RPL•Y+Q•W(STOR1)        COMPUTE CHECK SUM
                        ENT•Q•L(STORO)
                        RPL•Y+Q•W(STOR2)
        RTCL12          ENT•B3•L(CRO)
                        JP•W(RTCL13+B3)         TO JUMP TABLE
        RTCL13          00037•RTCL15            NO MODIFICATION
```

```
00272                00052•J0                MODIFY LOWER WITH B1
                     00074•J2                MODIFY UPPER WITH B1
                     00070•J3                MODIFY BOTH WITH B1
                     00064•J4                PERFORM INSTRUCTION
                     00062•J0                MODIFY LOWER WITH B5
                     00066•J0                MODIFY LOWER WITH B6
                     00072•J7                STOP
          J0         ENT•A•11040

                     ADD•A•L(CR0)            CONTROL DIGIT DEFINES B BOX
                     STR•A•U(RTCL14)
          RTCL14     0•0
                     RPL•A+Y•LX(STOR0)       MODIFY LOWER
          RTCL15     ENT•Q•W(STOR0)
                     STR•Q•W(B4)             STORE IN LOCATION DEFINED BY B4
                     ENT•B4•1+B4
                     JP•RTCL1

          J2         ENT•A•XB1
                     RPL•A+Y•UX(STOR0)       ADD B1 TO UPPER
                     JP•RTCL15
          J3         ENT•A•XB1
                     RPL•A+Y•LX(STOR0)       ADD B1 TO LOWER
                     JP•J2
          J4         A-MOVE•1•W(STOR0)W(RTCL17)      SET UP INSTRUCTION
                     ENT•C3•10036            DISABLE READER

          RTCL17     0•0                     PERFORM INSTRUCTION
                     RJP•TOR0
                     JP•RTCL1
          J7         RPL•Y-1•L(CSI0)         DECREASE CSI BY 1
                     JP•RTCL20•ANOT
                     A-SET•L(CSI0)•2         CHECK SUM-SET INDICATOR TO 2
                     A-SET•L(RTCL10)•RTCL12  SWITCH TO IGNORE TEST FOR CK SUM


                     JP•RTCL1
          RTCL20     ENT•C3•10036            DISABLE READER
                     RPL•Y+1•LX(CSI0)ANOT    ADVANCE CSI BY1
                     JP•0•STOP               STOP IF ZERO
                     ENT•A•W(STOR1)          COMPARE UPPER CHECK SUM
                     SEL•CP•W(STOR3)AZERO
                     JP•RTCL21
                     ENT•A•W(STOR2)          COMPARE LOWER CHECK SUM

                     SEL•CP•W(STOR4)AZERO
          RTCL21     JP•TCSE0                CHECK SUM ERROR
                     JP•0•STOP
```

```
00350 LIB0    CL•U(STOR2)              CLEAR CHECK SUM INDICATOR
              CL•W(STOR0)              CLEAR CHECK SUM STORAGE
              CL•W(STOR1)              CLEAR CHECK SUM STORAGE
              RJP•READ0                READ
              ENT•Q•77                 MASK TO Q

              COM•MASK•33•ANOT         SKIP IF NOT NINE CODE
              JP•LIB6                  NINE CODE
              COM•MASK•60•AZERO        SKIP IF EIGHT CODE
              JP•LIB10
              A-SET•L(LIB20)LIB1       SET SWITCH
              JP•LIB10
       LIB1   A-SET•L(LIB20)LIB21
              A-MOVE•1•W(STOR0)W(STOR3)        CHECK SUM, LOWER

              A-MOVE•1•W(STOR1)W(STOR4)        CHECK SUM, UPPER
              CL•W(STOR0)              CLEAR CHECK SUM STORAGE
              CL•W(STOR1)              CLEAR CHECK SUM STORAGE
              RJP•LIB2                 READ CHECK SUMS
              ENT•A•W(STOR3)
              RPL•A+Y•W(STOR0)         ADD LOWER CHECK SUM
              ENT•A•W(STOR4)
              RPL•A+Y•W(STOR1)         ADD UPPER CHECK SUM

              JP•LIB21                 JUMP TO TURN OFF READER
       LIB2   JP•0                     EXIT
              RPL•Y+1•U(STOR2)         SET CHECK SUM INDICATOR
              ENT•B2•0                 CLEAR B2
       LIB3   ENT•B1•11                SET INDEX
              ENT•B3•STOR0+B2          STORAGE ADDRESS TO B3
       LIB4   ENT•Q•77                 MASK TO Q
       LIB5   RJP•READ0                READ

              RPT•10•ADV
              COM•MASK•L(LIB23)AZERO   IDENTIFY FLEX CODE
              JP•LIB5                  READ NEXT FRAME, CODE NOT IDENTIFIED
              RJP•LIB24                ASSEMBLE
              BJP•B1•LIB4              READ REMAINDER OF CHECK SUM
              BSK•B2•1
              JP•LIB3                  READ LOWER CHECK SUM
              JP•LIB2                  TO EXIT

       LIB6   RJP•LIB2                 READ CHECK SUM
       LIB7   RJP•READ0                READ
       LIB10  SUB•A•45•AZERO           SKIP IF C•R•
              JP•LIB7
       LIB11  CL•L(STOR2)              CLEAR ADDRESS ASSEMBLY
              ENT•B1•5                 SET INDEX
              ENT•B2•12                SET INDEX
       LIB12  ENT•Q•77                 MASK TO Q

       LIB13  RJP•READ0                READ
              COM•MASK•45•ANOT         SKIP IF NOT CR
              JP•LIB11
              COM•MASK•42•ANOT         SKIP IF NOT PERIOD CODE
```

```
00435
                JP•LIB17                JUMP
LIB14           RPT•10•ADV
                COM•MASK•L(LIB23)AZERO  IDENTIFY FLEXCODE
                JP•LIB13                NOT IDENTIFIED, READ NEXT FRAME

                BJP•B1•LIB16            ASSEMBLE ADDRESS
                BJP•B2•LIB15            ASSEMBLE CONTENTS
                JP•LIB7                 READ TO NEXT CR
LIB15           RJP•LIB24               ASSEMBLE
                ENT•A•B2•AZERO          SKIP IF B2=0
                JP•LIB12                READ NEXT FRAME
                ENT•Q•U(B3)             UPPER HALF CONTENTS TO Q
                RPL•Y-Q•W(STOR0)        SUBTRACT FROM CHECK SUM

                ENT•Q•L(B3)             LOWER HALF CONTENTS TO Q
                RPL•Y-Q•W(STOR1)        SUBTRACT FROM CHECK SUM
                JP•LIB7                 READ
LIB16           ENT•A•L(STOR2)          ADDRESS ASSEMBLY TO A
                LSH•A•3
                ADD•A•B7                ADD ONE DIGIT
                STR•A•L(STOR2)
                ENT•A•B1•AZERO          SKIP IF B1=0

                JP•LIB13                READ NEXT FRAME
                ENT•B3•L(STOR2)         ADDRESS TO B3
                CL•W(B3)                CLEAR ADDRESS
                JP•LIB13•STOP6          READ CONTENTS
LIB17           RJP•READ0               READ
                COM•MASK•42•AZERO       SKIP IF PERIOD CODE
                JP•LIB13+1              JUMP
LIB20           JP•LIB21                SWITCH

LIB21           ENT•C3•10036            DISABLE READER
                ENT•A•U(STOR2)ANOT      CHECK SUM INDICATOR TO A, SKIP A=0
                JP•0•STOP
                ENT•A•W(STOR0)AZERO     UPPER CHECK SUM TO A, SKIP A=0
                JP•LIB22                CHECK SUM ERROR
                ENT•A•W(STOR1)AZERO     LOWER CHECK SUM TO A, SKIP A= 0
LIB22           JP•TCSE0                JUMP TO TYPE
                JP•0•STOP

LIB23           00037•72                FLEXCODE FOR OCTAL DIGITS
                00052•66
                00074•62
                00070•64
                00064•70
                00062•74
                00066•52
                00072•37

LIB24           JP•0                    EXIT ASSEMBLY
                ENT•Q•7                 MASK TO Q
                ENT•A•W(B3)             ASSEMBLY TO A
                LSH•A•3
                SEL•SU•B7               ADD ONE DIGIT
                STR•A•W(B3)             STORE
                JP•LIB24                EXIT
```

```
                BIOCTAL LOAD
00520 LIB25     RJP•LIB31           READ AND ASSEMBLE ADDRESSES

                STR•Q•L(LIB27)      STORE LAST ADDRESS
                LSH•AQ•17           FIRST ADDRESS TO A
                ENT•B3•A            STORE IN B3
                CL•W(STOR0)         CLEAR CHECK SUM STORAGES
                CL•W(STOR1)
      LIB26     RJP•LIB31           READ AND ASSEMBLE ONE WORD
                STR•Q•W(B3)         STORE
                ENT•Q•U(B3)         UPPER HALF WORD TO Q

                RPL•Y+Q•W(STOR0)    ADD TO CHECK SUM
                ENT•Q•L(B3)         LOWER HALF WORD TO Q
                RPL•Y+Q•W(STOR1)    ADD TO CHECK SUM
      LIB27     BSK•B3•0            TEST FOR END OF LOAD
                JP•LIB26            CONTINUE
                RJP•LIB31           READ AND ASSEMBLE LOWER CHECK SUM
                SUB•Q•W(STOR1)QZERO SKIP IF CHECK SUM CORRECT
                JP•TCSE0            CHECK SUM ERROR

                RJP•LIB31           READ AND ASSEMBLE UPPER CHECK SUM
                SUB•Q•W(STOR0)QZERO SKIP IF CHECK SUM CORRECT
      LIB30     JP•TCSE0            CHECK SUM ERROR
                ENT•C3•10036        DISABLE READER
                JP•0•STOP
      LIB31     JP•0                EXIT, READ AND ASSEMBLE FIVE FRAMES
                ENT•B1•4            SET INDEX
      LIB32     RJP•READ0           READ

                LSH•A•30            FRAME TO A UPPER
                LSH•AQ•6            ASSEMBLE IN Q
                BJP•B1•LIB32        READ NEXT FRAME
                JP•LIB31            TO EXIT
      LIB33     JP•0                EXIT, COMPUTE CHECK SUMS
                STR•B6•L(STOR2)     STORE LAST ADDRESS
                STR•B7•U(STOR2)     STORE FIRST ADDRESS
                ENT•B5•B7           FIRST ADDRESS TO B5

                CL•A
      LIB34     ADD•A•L(B5)         ADD LOWER CHECK SUMS
                BSK•B5•B6
                JP•LIB34
                STR•A•W(STOR1)      STORE LOWER CHECK SUM
                ENT•B5•B7           RESET B5 TO FIRST ADDRESS
                CL•A
      LIB35     ADD•A•U(B5)         ADD UPPER CHECK SUM

                BSK•B5•B6
                JP•LIB35
                STR•A•W(STOR0)      STORE UPPER CHECK SUM
                JP•LIB33            TO EXIT
```

```
        BIOCTAL DUMP
00575   BIOCTLD0        RJP•STPCH               TURN ON PUNCH
                        RJP•PCHLD0              PUNCH LEADER
                        RJP•LIB33               COMPUTE CHECK SUMS
        BIOCTLD1        JP•BIOCTLD1•C1FULL      PUNCH BEGINNING CODE

                        ENT•C1•76
                        ENT•Q•W(STOR2)          ADDRESSES TO Q
                        RJP•BIOCTLD3            PUNCH
        BIOCTLD2        ENT•Q•W(B7)             WORD TO Q
                        RJP•BIOCTLD3            PUNCH
                        BSK•B7•B6               TEST FOR END OF DUMP
                        JP•BIOCTLD2             CONTINUE
                        ENT•Q•W(STOR1)          LOWER CHECK SUM TO Q

                        RJP•BIOCTLD3            PUNCH
                        ENT•Q•W(STOR0)          UPPER CHECK SUM TO Q
                        RJP•BIOCTLD3            PUNCH
                        RJP•PCHLD0              PUNCH LEADER
                        RJP•PCHLD0              PUNCH LEADER
                        ENT•C3•10026            DISABLE PUNCH
                        JP•0•STOP
        BIOCTLD3        JP•0                    EXIT, PUNCH FIVE FRAMES

                        ENT•B1•4                SET INDEX
        BIOCTLD4        LSH•AQ•6
        BIOCTLD5        JP•BIOCTLD5•C1FULL
                        ENT•C1•A                PUNCH ONE FRAME
                        BJP•B1•BIOCTLD4         PUNCH FOUR MORE FRAMES
                        JP•BIOCTLD3             TO EXIT
        PCHLD0          JP•0                    EXIT PUNCH LEADER
                        ENT•B1•140              140 FRAMES LEADER

        PCHLD1          JP•PCHLD1•C1FULL
                        ENT•C1•0
                        BJP•B1•PCHLD1
                        JP•PCHLD0               TO EXIT

        FLEXCODE DUMP
00635   FLEXD0          RJP•STPCH               TURN ON PUNCH
                        RJP•PCHLD0              PUNCH LEADER
                        ENT•Q•W(FLEXD14)        BEGINNING CODE TO Q
                        RJP•BIOCTLD3            PUNCH

                        RJP•LIB33               COMPUTE CHECK SUMS
                        RJP•FLEXD12             PUNCH C.R.
        FLEXD1          ENT•B1•0                CLEAR B1
                        JP•FLEXD6•KEY1
        FLEXD2          ENT•Q•B7                ADDRESS TO Q
                        LSH•Q•17                ADDRESS TO Q UPPER
                        RJP•FLEXD7              PUNCH
                        ENT•B1•B3               LAST DIGIT TO B1
```

FLEXCODE DUMP (Cont.)

```
00651                ENT•B5•1                  SET INDEX
                     ENT•Q•W(B7)               WORD TO Q
FLEXD3               JP•FLEXD3•C1FULL
                     ENT•C1•4                  PUNCH SPACE
                     RJP•FLEXD7                PUNCH HALF WORD
                     BJP•B5•FLEXD3             PUNCH SECOND HALF WORD
                     ENT•A•X77770+B1•ANOT      SKIP IF ADDRESS DOES NOT END IN 7
                     RJP•FLEXD12               PUNCH C.R.

                     RJP•FLEXD12               PUNCH C.R.
FLEXD4               BSK•B7•B6                 TEST FOR END OF DUMP
                     JP•FLEXD1                 CONTINUE
                     ENT•Q•W(FLEXD15)          END CODE TO Q
                     RJP•BIOCTLD3              PUNCH
                     ENT•Q•W(STOR0)            UPPER CHECK SUM TO Q
                     RJP•FLEXD7                PUNCH
                     RJP•FLEXD7                PUNCH

                     RJP•FLEXD12               PUNCH C.R.
                     ENT•Q•W(STOR1)            LOWER CHECK SUM TO Q
                     RJP•FLEXD7                PUNCH
                     RJP•FLEXD7                PUNCH
                     RJP•PCHLD0                PUNCH LEADER
                     RJP•PCHLD0                PUNCH LEADER
                     ENT•C3•10026              DISABLE PUNCH
                     JP•0•STOP

FLEXD6               ENT•Q•W(B7)               WORD TO Q
                     LSH•AQ•36•AZERO           WORD TO A, SKIP IF A=0
                     JP•FLEXD2                 JUMP TO PUNCH
                     JP•FLEXD4                 BY PASS PUNCH
FLEXD7               JP•0                      EXIT, PUNCH FIVE FRAMES
                     ENT•B2•4                  SET INDEX
FLEXD10              LSH•Q•3
                     ENT•LP•7                  ONE DIGIT TO A

                     ENT•B3•A                  ONE DIGIT TO B3
FLEXD11              JP•FLEXD11•C1FULL
                     ENT•C1•U(LIB23+B3)        PUNCH ONE DIGIT
                     BJP•B2•FLEXD10            NEXT DIGIT
                     JP•FLEXD7                 TO EXIT
FLEXD12              JP•0                      EXIT, PUNCH C.R.
FLEXD13              JP•FLEXD13•C1FULL
                     ENT•C1•45

                     JP•FLEXD12                TO EXIT
FLEXD14              00456•06045               BEGINNING CODE FOR FLEX DUMP
FLEXD15              42424•50000               END CODE FOR FLEX DUMP
```

```
          SWITCHER
00724     SWITCH0     JP•BIOCTLD0•KEY2        BIOCTAL DUMP IF KEY2
                      JP•FLEXD0•KEY3          FLEX DUMP IF KEY3
          SWITCH2     RJP•TOR0
          SWITCH1     RJP•READ0
                      JP•SWITCH1•AZER0

                      ENT•Q•77
                      COM•MASK•76•ANOT
                      JP•LIB25                BIOCTAL LOAD
                      COM•MASK•45•AZER0
                      JP•TFE0
                      RJP•READ0
                      COM•MASK•60•ANOT
                      JP•LIB0                 FLEX LOAD

                      COM•MASK•33•AZER0
                      JP•TFE0
                      JP•RTCL0                RELATIVE LOAD
          STOR10      0•0

          COMPUTE CHECK SUMS ON UTILITY PACKAGE
00745     CSUP0       CL•A
                      RPT•SWITCH1+6•ADV
                      ADD•A•U(INSPT0)
                      SUB•A•W(CSUP2)AZER0      TEST UPPER CHECK SUM

                      JP•TCSE0
                      CL•A
                      RPT•SWITCH1+6•ADV
                      ADD•A•L(INSPT0)
                      SUB•A•W(CSUP3)AZER0      TEST LOWER CHECK SUM
                      JP•TCSE0
                      ENT•A•W(CSUP1)SKIP
          CSUP1       JP•SWITCH0

                      STR•A•W(0)
                      JP•0•STOP
          READ0       JP•0                     READ A FRAME TO A
          READ1       JP•READ1•C1EMPTY
                      STR•C1•A
                      JP•READ0

          ••
```

SECTION E3

AN/USQ-20

WIRED BOOTSTRAP PROGRAM

AN/USQ-20 WIRED BOOTSTRAP PROGRAM

The AN/USQ-20 Unit Computer contains a 16-word auxiliary memory in addition to the large main memory. This auxiliary memory is semipermanently wired and operates in a nondestructive read-out mode. It is used to contain a bootstrap (program-load) routine to facilitate rapid read-in of all-purpose utility routines.

Instructions listed in Table E3-1 are wired into the semipermanent wired-core memory of the AN/USQ-20 Unit Computers.

This wired bootstrap reads in a paper-tape bootstrap of bioctal format (beginning code of 76) containing 136 octal instructions; the instructions are stored in main memory Addresses 00140 through 00275. Load information, beginning and last addresses, and the two words of check-sum found at the beginning and end of the tape are stored at Addresses 00136, 00137, 00276, and 00277. On completion of the load, program control is given to Address 00140.

The bootstrap tape requires no positioning in the reader and is read into the computer in the forward direction; only 6-level tape is acceptable. To initiate the bootstrap routine at the console, the operator *master clears* the computer and sets the AUTOMATIC RECOVERY switch to the DOWN position.

If the AUTOMATIC RECOVERY switch is in the UP position when a computer fault occurs, the wired-memory mode is set and the instruction at Address 00000 of wired memory is executed. This automatically causes read-in of the bioctal bootstrap tape which has previously been placed in the reader.

The tape that is read into the AN/USQ-20 Unit Computer by the wired-memory bootstrap contains instructions to perform the following functions:

1) Bioctal load with automatic check-sum computation.

2) Incorrect format detection.

3) Check sum the load of the bootstrap tape.

TABLE E3-1. WIRED BOOTSTRAP INSTRUCTIONS

00000	12100	00000	CL•B^1	Set counter, fault address
00001	12200	40040	ENT•B^2•40040	Initialize, reader code
00002	13j30	00001	Ex Fct•C^j•W(01)	Enable reader
00003	10000	00000	CL•Q	Clear assembly word
00004	73j30	00017	IN•C^j•W(17)	Buffer 1 frame
00005	62j00	00005	JP•05•C^j ACTIVEIN	Wait on frame
00006	05000	00006	LSH•Q•6	Normalize
00007	26430	00136	ADD•Q•W(136)QZero	Form word, test 76
00010	11401	00000	ENT•A•B^1•AZero	76 code found
00011	72200	00004	BJP•B^2•04	Test frame index
00012	14031	00136	STR•Q•W(136+B^1)	Store word
00013	12200	00004	ENT•B^2•4	Set frame index
00014	71100	00141	BSK•B^1•141	Test word index
00015	61000	00003	Jp•03	Repeat
00016	61000	00140	Jp•140	To check cksum
00017	00136	00136	00136•00136	Buffer address

4) Store Switcher Jump and Fault Jump.

The wired-memory bootstrap loads the bootstrap tape and jumps to Address 00140. Address 00140 gives program control to the internal check-sum routine which compares the computed check-sum words with the two words loaded from the tape. This routine disables the reader, clears the wired-memory mode, and drops the interrupt lockout. It places a *Jump-to-Address* 00000 at the fault entrance address, Address 00000, and places jump to the switcher at Address 00001. Address 00001 retains program control with a key-stop option (Key 7). If the check-sum computation is incorrect, the FAULT light on the punch tape unit is lighted and registers A and Q are set to all *ones* and/or the computer will "hang-up" (see Figure E3-1).

The bioctal load routine entered on the bootstrap tape reads any bioctal tape and stores it in a specified memory area. If desired, any other program in bioctal format can be physically contained on this same tape, separated by leader; by not exercising the key-stop option, this program is loaded in memory with the one operation.

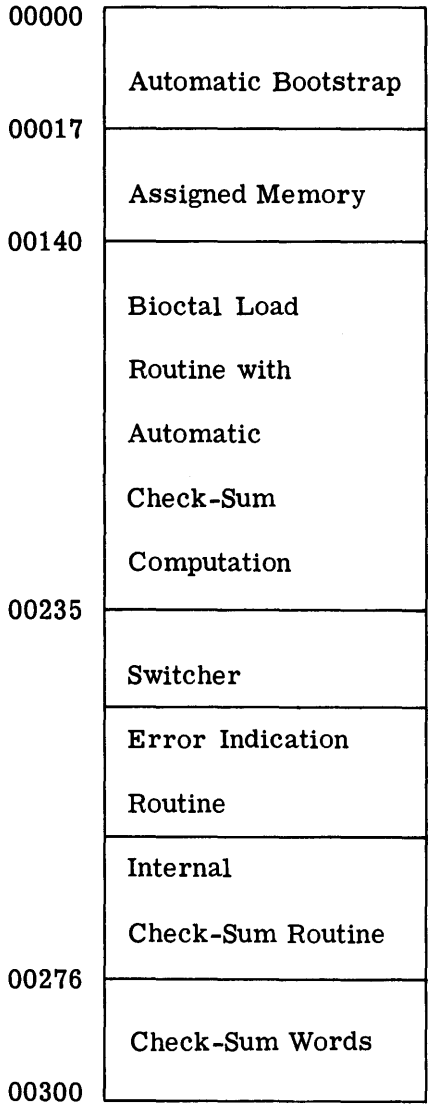


Figure E3-1. Memory Locations Used by Automatic Bootstrap and Bootstrap Tape Programs

SECTION E4

PAPER-TAPE CONTROL PROGRAM

FOR AN/USQ-20 COMPUTER

PAPER-TAPE CONTROL PROGRAM
FOR AN/USQ-20 COMPUTER

1. BASIC INFORMATION

The Paper-Tape Control Program (PTCP) described herein becomes an integral part of the NTDS Utility System for AN/USQ-20 Computers; it may, however, be used independently as a paper-tape utility routine*.

As defined in Section E3, "AN/USQ-20 Wired Bootstrap Program", activation of the Wired Bootstrap Program first causes read-in of a bootstrap tape. The tape bootstrap routine then causes a subsequent read-in of the PTCP tape which is in bioctal format.

Including the Bioctal Load routine of the bootstrap tape, the PTCP performs the following operations:

1) Bioctal Load
2) Bioctal Dump
3) Bioctal Check-Read
4) Relative-Bioctal Load
5) Flex Load
6) Flex Dump
7) Store Q
8) Inspect And Change

The CS-1 Compiler produces output formats consistent with those required for handling by the PTCP. For a detailed description of these formats see NTDS Technical Note No. 220, *CS-1 Utility Package,* or NTDS Technical Note No. 243, *NTDS Utility System.*

The switcher of the PTCP supersedes the switcher of the bootstrap program; therefore, the switcher of the PTCP routes a request to the appropriate routine for any of the operations listed. The *jump-to-the-switcher* at Address 00001 remains valid; hence, the program is initiated

*The PTCP performs the same paper-tape control functions for the AN/USQ-20 as the CS-1 Utility Package performs for the AN/USQ-17.

by setting the P register to either 00001 or 00235.

2. OPERATIONS DEFINITIONS AND PROGRAM FORMATS

A. *BIOCTAL LOAD*

A Bioctal program must begin with a 76 (code for Bioctal tape). Immediately after the 76, initial and final addresses of the program are loaded, and the rest of the tape is loaded without additional addresses in the program. Because of the delimiting addresses, the load will stop immediately after an automatic check sum is completed without an end code.

B. *BIOCTAL DUMP*

The Bioctal Dump routine is initiated manually by the operator at the console. The output, on perforated paper tape, is the 76 code followed by 1) initial and final addresses of the program being dumped, 2) contents of the included memory addresses, and 3) check sums. Output does not indicate each individual memory address from which the content is dumped (other than the initial and final addresses).

C. *BIOCTAL CHECK-READ*

The Bioctal Check-Read routine enables the computer to check an absolute bioctal perforated tape against its associated core-memory cells. Objectively, the task assures good bioctal dumps. If the Check-Read detects a discrepancy, the A register holds the tape word, the Q register holds the memory word, and the computer stops. The address of the discrepancy appears in the B^4 register. The operator may then check to determine the cause of the discrepancy; he is able to continue the Check-Read from that point.

D. *RELATIVE-BIOCTAL LOAD*

A Relative-Bioctal program must begin with a 75 (code for relative-bioctal tape). The entire program must be written relative to zero. A numeric code preceding each programmed instruction tells the computer how to modify the instruction for storage at a modified address. A control digit of 7, followed by a check sum, terminates the load and initiates an automatic check-sum computation.

E. *FLEX LOAD*

A Flex program must begin with a 45 code (carriage return), followed by a 60 code (8), followed by a 45 code. If check sums are required, two 60 codes appear between the 45 codes. Each instruction of a flex tape is preceded by a 5-digit address, the addresses need not be

sequential. The next 10 octal characters comprise the instruction word. All other character codes including the notes are ignored. A return indicates the end of the instruction. An end code of double period (..), followed by check sum, terminates the load and initiates an automatic check-sum computation.

F. *FLEX DUMP*

The computer operator initiates manually the Flex Dump routine at the console. The "88" format (with the check sum at the end) is the only one dumped. The output on perforated paper tape includes both the addresses and the contents of the memory locations being dumped.

G. *STORE Q*

The Store-Q function permits the computer operator to load an area of memory with a value which he manually enters into the Q register. The first and last addresses of the area which is to be filled are specified in the B registers. If Q = 0, the area is cleared.

H. *INSPECT AND CHANGE*

The Inspect and Change routine causes the contents of a specified address to be entered into registers A and Q. Content of Q may then be changed manually; content of A serves as a visual aid. Content of Q is then returned to the address from which it was taken. The inspection address need be entered only the first time; thereafter, contents of sequential addresses will be brought into registers A and Q with each successive performance of the Inspect and Change function. Should the operator wish to inspect the contents of some address other than the next sequential address, he may do so by setting the new address before returning contents of Q.

3. OPERATING INSTRUCTIONS

1) To load Paper-Tape Control Program

   a) Place tape in reader (need not be positioned)

   b) *Master clear* the computer

   c) Depress AUTOMATIC RECOVERY switch

2) Error Detection

   a) Format error - Q and A are set to all *ones*
   FAULT light on punch-tape unit is lighted
   Computer hangs up

b) Check-sum error - same as for format error

3) Flex, Bioctal, or Relative-Bioctal Load

a) Place tape in reader

b) *Master clear* the computer

c) Set B^4 to base address for Relative-Bioctal Load

d) Set P to 1 and start

4) Bioctal Dump

a) *Master clear* the computer

b) Set Key 2

c) Set beginning address in B^6

d) Set last address in B^5

e) Set P to 1 and start

5) Bioctal Check-Read

a) *Master clear* the computer

b) Set Key 1

c) Set P to 1 and start

6) Flex Dump

a) *Master clear* the computer

b) Set Key 3

c) Set beginning address in B^6

d) Set last address in B^5

e) Set P to 1 and start

7) Store Q

a) *Master clear* the computer

b) Set data to be stored in Q

c) Set B^6 to base address of storage area

d) Set B^5 to last address of storage area

e) Set P to 750 and start

8) Inspect and Change

a) *Master clear* the computer

b) Set inspection address in B^4

c) Set P to 760 and Start

Contents of the address will appear in registers A and Q; the address will appear in B^3.

d) Make desired change in Q.

e) Start

SECTION E5

TELETYPE PAPER-TAPE CONTROL PROGRAM

FOR AN/USQ-20 COMPUTER

CONTENTS

TELETYPE PAPER-TAPE CONTROL PROGRAM FOR AN/USQ-20 COMPUTER

1. BASIC INFORMATION

The Teletype Paper-Tape Control Program (PTCP-TWX) described herein becomes an integral part of the NTDS Utility System for the AN/USQ-20 computers. It may, however, be used independently as a paper-tape utility routine.

As defined in Section E3, "AN/USQ-20 Wired Bootstrap Program", activation of the *wired bootstrap* first causes read-in of a *bootstrap tape*. The tape bootstrap routine then causes a subsequent read-in of the PTCP tape which is in bioctal format.

Including the bioctal load routine of the bootstrap tape, the PTCP-TWX performs the following operations:

1) Bioctal Load
2) Bioctal Dump
3) Bioctal Check-read
4) Relative-bioctal Load
5) Teletype Load
6) Teletype Dump
7) Store Q
8) Inspect and Change

With the exception of Teletype format on paper tape, the CS-1 Compiler produces output formats which are consistent with those necessary for handling by the PTCP. For a detailed description of these formats see Section E2, "CS-1 Utility Package," or Section E1, "NTDS Utility System". The Teletype format on paper tape is anologous to the Flex format which is described in the above sections. Teletype code, of course, is used in place of the Flexowriter code. In addition, the following requirements must be adhered to when producing a paper tape on the Teletype:

1) A carriage return must be followed by a "line feed"

2) Leader must be a "letters" code (37).

The switcher of the PTCP supersedes the switcher of the bootstrap program; therefore, the switcher of the PTCP routes a request for any of the functions listed above to the appropriate routine. The *jump-to-the-switcher* at address 00001 remains valid; hence, the program is initiated by setting the P register to either 00001 or 00235.

2. OPERATION DEFINITIONS AND PROGRAM FORMATS

A. *BIOCTAL LOAD*

A Bioctal program must begin with a 76 (code for Bioctal tape). Immediately after the 76, the initial and final addresses of the program are loaded and the rest of the tape is loaded without additional addresses in the program. Because of the delimiting addresses, the load will stop immediately after an automatic check sum is completed without an end code.

B. *BIOCTAL DUMP*

The Bioctal dump routine is initiated manually by the operator at the console. The output, on punched paper tape, is the 76 code followed by: 1) the initial and final addresses of the program being dumped, 2) the contents of the included memory addresses, and 3) the check sums. Output codes do not indicate each individual address in memory from which the content is dumped, other than the initial and final addresses.

C. *BIOCTAL CHECK-READ*

The Bioctal Check-read routine enables the computer to check an absolute bioctal punched tape against its associated core memory cells. Objectively, the task assures good bioctal dumps. If the check-read detects a discrepancy, the A register holds the tape word, the Q register holds the memory word, and the computer stops. The address of the discrepancy appears in the B^4 register. At this time, the operator may check to determine the cause of the discrepancy; he is able to continue the check-read from this point by depressing the HIGH SPEED switch.

D. *RELATIVE-BIOCTAL LOAD*

A Relative-bioctal program must begin with a 75 (code for relative-bioctal tape). The entire program must be written relative to zero. A numeric code preceding each programmed instruction tells the computer how to modify the instruction for storage at a modified address.

A control digit of 7, followed by check sum, terminates the load and initiates an automatic check-sum computation.

E. *TELETYPE LOAD*

A Teletype program must begin with a 02, a 10, and a 33 code (carriage return, line feed, and figures) followed by a 14 code (8) followed by a 02 and a 10 code. If check sums are required, two 14 codes (88) appear. Each instruction of a Teletype tape is preceded by a 5-digit address, the addresses need not be sequential. The next 10 octal characters comprise the instruction word. All other character codes including the notes are ignored. The codes 02 and 10 (carriage return and line feed) indicate the end of the instruction. An end code of double period (..) terminates the load. If there is a check sum, it follows the double periods and is computed automatically by the load routine. The TWX load may be loaded via the high-speed reader or by the Teletype reader. Adjustments for tape width are made at the respective readers. When producing a Teletype tape on the Teletype, the following considerations should be adhered to:

1) A carriage return must be followed by a *line feed*

2) Leader must be a *letters* code (37)

3) To stop Teletype reader, two blank frames must follow the double periods.

See Figure 1 for example of Teletype format on paper tape.

F. *TELETYPE DUMP*

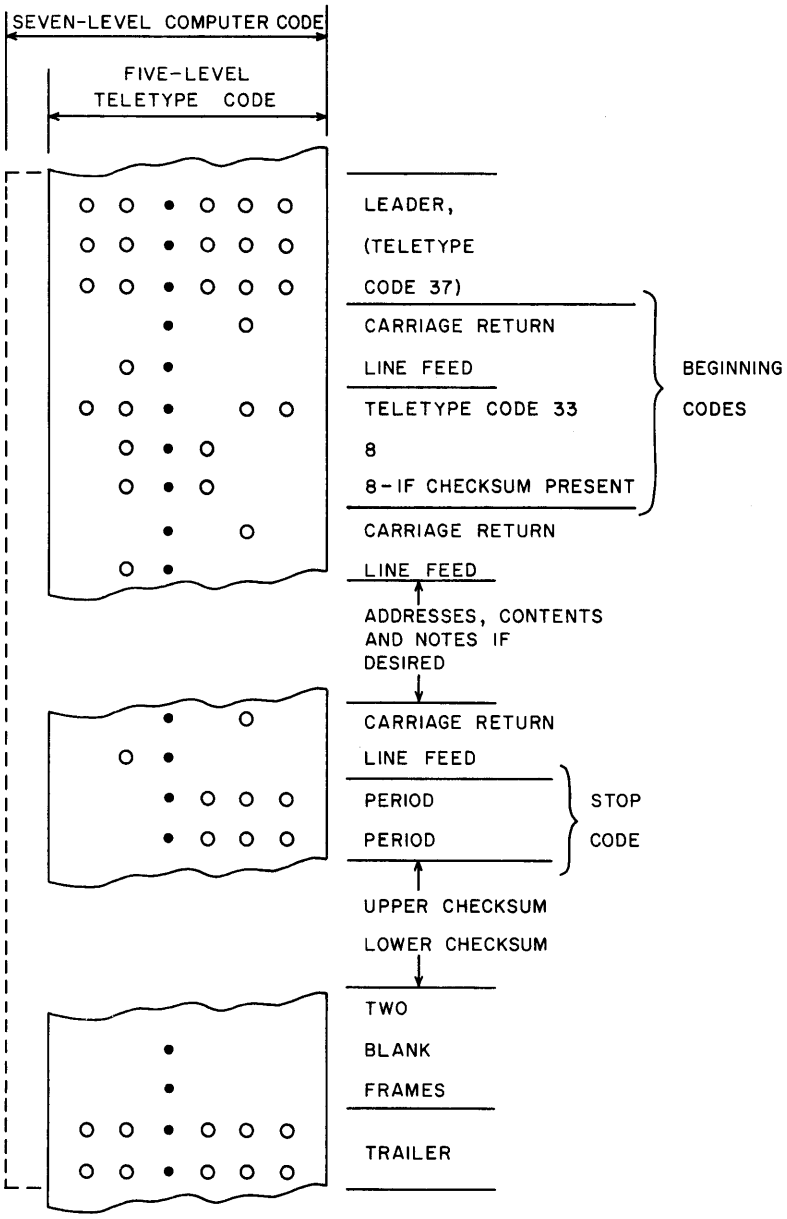
The computer operator manually initiates the Teletype Dump routine at the console. The "88" format (with the check sum at the end) is the only one dumped. The output on punched paper tape includes both the addresses and the contents of the memory locations being dumped. The TWX dump may be via the high-speed punch or via the Teletype.

G. *STORE Q*

The Store Q function permits the computer operator to load an area of memory with a value which he manually enters into the Q register. The first and last addresses of the area which is to be filled are specified in the B registers. If Q = O, the area is cleared.

H. *INSPECT AND CHANGE*

The Inspect and Change routine causes the contents of a specified address to be entered into registers A and Q. The contents of Q may then be changed manually, with A serving as a



NOTE:

A TELETYPE CARRIAGE RETURN MUST BE FOLLOWED BY A LINE FEED, A SINGLE BEGINNING CODE OF 8 INDICATES NO CHECKSUM WHILE AN 88 CODE REQUIRES CHECKSUM WORDS TO BE PRESENT. SEE TECHNICAL NOTE NO. 238, FUNCTIONAL SPECIFICATIONS FOR THE SERVICE TEST TELETYPE ADAPTER.

Figure E5-1. Teletype Format on Paper Tape

visual aid, Q is then returned to the address from which it was taken. The inspection addresses will be brought into A and Q with each successive performance of the Inspect and Change function. Should the operator wish to inspect the contents of some address other than the next sequential address, he may do so be setting the new address before returning Q.

3. OPERATING INSTRUCTIONS

1) To load Paper Tape Control - TWX Program

   a) Place tape in reader (need not be positioned)

   b) Master clear the computer

   c) Depress AUTOMATIC RECOVERY switch

2) Error Detection

   a) Format error - Q and A are set to all *ones*
      FAULT light on punch-tape unit is lighted
      Computer hangs up

   b) Check-sum error - same as for format error.

3) TWX, Bioctal or Relative-bioctal Load

   a) Via the high-speed reader

      1 Place tape in reader

      2 Master clear the computer

      3 Set B^4 for base address of relative-bioctal load

      4 Set P to 1 and start

   b) TWX via the Teletype reader (for tapes of Teletype format only)

      1 Place tape in reader on special leader (37 codes)

      2 Position switch on reader to RUN

      3 Master clear the computer

      4 Set Key 1 and 3

      5 Set P to 1 and start
      (Computer hangs up to permit execution of step 6

      6 Depress the SEND button on the Teletype

4) Bioctal Dump

   a) Master clear the computer

   b) Set Key 2

   c) Set beginning address in B^6

   d) Set last address in B^5

   e) Set P to 1 and start

5) Bioctal Check-read

   a) Master clear the computer

   b) Set Key 1

   c) Set P to 1 and start

6) TWX Dump

   a) Master clear the computer

   b) Set Key 3 to dump on high-speed punch or set Key 2 and Key 3 to dump on Teletype

   c) Set beginning address in B^6

   d) Set last address in B^5

   e) Set P to 1 and start

7) Store Q

   a) Master clear the computer

   b) Set data to be stored in Q

   c) Set B^6 to base address of storage area

   d) Set B^5 to last address of storage area

   e) Set P to 750 and start

8) Inspect and Change

   a) Master clear the computer

   b) Set inspection address in B^4

c) Set P to 760 and start

Contents of the address will appear in A and Q registers; the address will appear in B^3.

d) Make desired change in Q

e) Start

Ser. 106

PROGRAMMERS' GUIDE

Addendum

The enclosed material entitled *Teletype Paper-Tape Control Program for AN/USQ-20 Computer* is to be inserted into RRU document PX 1493, *Programmers' Guide*, as Section E5.

SECTION F

CHARTS

SECTION F1

MACHINE CODE CHART

(for AN/USQ-17 COMPUTERS)

MACHINE CODE CHART

(Applicable to the AN/USQ-17 Computers Only)

01	SHIFT Q RIGHT	SHIFT (Q) RIGHT BY (Y)
02	SHIFT A RIGHT	SHIFT (A) RIGHT BY (Y)
03	SHIFT AQ RIGHT	SHIFT (AQ) RIGHT BY (Y)
04	COMPARE	SENSE (j) $(A)_i = (A)_f$ *
05	SHIFT Q LEFT	SHIFT (Q) LEFT BY (Y)
06	SHIFT A LEFT	SHIFT (A) LEFT BY (Y)
07	SHIFT AQ LEFT	SHIFT (AQ) LEFT BY (Y)
10	ENTER Q	(Y) → Q
11	ENTER ACCUMULATOR	(Y) → A
12	ENTER B REGISTER	$(Y) \rightarrow (B)_j$
13	ENTER C REGISTER	$(Y) \rightarrow (C)_j$
14	STORE Q	(Q) → Y
15	STORE ACCUMULATOR	(A) → Y
16	STORE B REGISTER	$(B)_j \rightarrow Y$
17	STORE C REGISTER	$(C)_j \rightarrow Y$
20	ADD	(A) + (Y) → A
21	SUBTRACT	(A) − (Y) → A
22	MULTIPLY	(Q)(Y) → AQ
23	DIVIDE	(AQ) / (Y) → Q, R → A_f
24	ADD REPLACE	(A) + (Y) → Y & A
25	SUBTRACT REPLACE	(A) − (Y) → Y & A
26	Q ADD	(Q) + (Y) → Q. $(A)_i = (A)_f$
27	Q SUBTRACT	(Q) − (Y) → Q. $(A)_i = (A)_f$
30	LOAD A ADD Q	(Y) + (Q) → A
31	LOAD A SUBTRACT Q	(Y) − (Q) → A
32	ADD Q AND STORE	(A) + (Q) → Y & A
33	SUBTRACT Q AND STORE	(A) − (Q) → Y & A
34	REPLACE ADD Q	(Y) + (Q) → Y & A
35	REPLACE SUBTRACT Q	(Y) − (Q) → Y & A
36	REPLACE ADD ONE	(Y) + 1 → Y & A
37	REPLACE SUBTRACT ONE	(Y) − 1 → Y & A
40	ENTER LOGICAL PRODUCT	L(Y)(Q) → A
41	ADD LOGICAL PRODUCT	L(Y)(Q) + (A) → A
42	SUBTRACT LOGICAL PRODUCT	(A) − L(Y)(Q) → A
43	MASKED COMPARISON	(A) − L(Y)(Q), SENSE (j), (A) + L(Y)(Q). $(A)_i = (A)_f$
44	REPLACE LOGICAL PRODUCT	L(Y)(Q) → Y & A
45	REPLACE ADD LOGICAL PRODUCT	L(Y)(Q) + (A) → Y & A
46	REPLACE SUBTRACT LOGICAL PRODUCT	(A) − L(Y)(Q) → Y & A
47	STORE LOGICAL PRODUCT	L(A)(Q) → Y. $(A)_i = (A)_f$
50	SELECTIVE SET	SET $(A)_n$ FOR $(Y)_n = 1$
51	SELECTIVE COMPLEMENT	COMPLEMENT $(A)_n$ FOR $(Y)_n = 1$
52	SELECTIVE CLEAR	CLEAR $(A)_n$ FOR $(Y)_n = 1$
53	SUBSTITUTE	$(Y)_n \rightarrow (A)_n$ FOR $(Q)_n = 1$
54	REPLACE SELECTIVE SET	SET $(A)_n$ FOR $(Y)_n = 1$, → Y & A
55	REPLACE SELECTIVE COMPLEMENT	COMPLEMENT $(A)_n$ FOR $(Y)_n = 1$, → Y & A
56	REPLACE SELECTIVE CLEAR	CLEAR $(A)_n$ FOR $(Y)_n = 1$, → Y & A

57 REPLACE SUBSTITUTE ..... $(Y)_n \rightarrow (A)_n$ FOR $(Q)_n = 1, \rightarrow Y$
60 ARITHMETIC JUMP ..... j = 0, NO JUMP; J ≠ 0, JUMP *
61 MANUAL JUMP ..... JUMP *
62 INPUT JUMP ..... $(C)_j$ FULL JUMP TO ADDRESS (Y)
63 OUTPUT JUMP ..... $(C)_j$ EMPTY JUMP TO ADDRESS (Y)
64 ARITHMETIC RETURN JUMP ..... j=0 NO JUMP, j ≠ 0 JUMP TO Y + 1, $P \rightarrow Y$ *
65 MANUAL RETURN JUMP ..... RETURN JUMP, STOP IF KEY SET *
66 INPUT RETURN JUMP ..... $(C)_j$ FULL RETURN JUMP TO ADDRESS (Y)
67 OUTPUT RETURN JUMP ..... $(C)_j$ EMPTY RETURN JUMP TO ADDRESS (Y)
70 INITIATE REPEAT ..... EXECUTE NI (Y) TIMES *
71 INDEX SKIP ..... $(B)j = (Y)$ SKIP NI, CLEAR Bj ; $(B)j \neq (Y)$ ADV Bj, RNI
72 INDEX JUMP ..... $(B)j = 0$ READNI ; $(B)j \neq 0$ $(B)j-1$, JUMP TO ADDRESS (Y)
73 (TO BE REVISED)
74 (TO BE REVISED)
75 INITIATE INPUT BUFFER ..... $k = 3, (Y) \rightarrow 0000j; k \neq 3, (Y)_L \rightarrow (0000j)_L$, SET (d) TO 4
76 INITIATE OUTPUT BUFFER ..... $k = 3, (Y) \rightarrow 0000j; k \neq 3, (Y)_L \rightarrow (0000j)_L$, SET (d) TO 6.

J DESIGNATORS FOR * COMMANDS

j	04	60	61	64	65	70
0	NO SKIP	NO JUMP	JUMP	NO JUMP	RET JUMP	NOTE 1
1	SKIP	JUMP	JUMP KEY 1	RET JUMP	JUMP KEY 1	" 2
2	SKIP (Y) ≤ (Q)	JUMP Q POS	JUMP KEY 2	JUMP Q POS	JUMP KEY 2	" 3
3	SKIP (Y) > (Q)	JUMP Q NEG	JUMP KEY 3	JUMP Q NEG	JUMP KEY 3	" 4
4	SKIP (Q) ≥ (Y) AND (Y) > (A)	JUMP A = 0	JUMP, STOP	JUMP A = 0	JUMP, STOP	NOTE 1
5	SKIP (Q) < (Y) OR (Y) ≤ (A)	JUMP A ≠ 0	JUMP, STOP KEY 5	JUMP A ≠ 0	JUMP, STOP KEY 5	" 2
6	SKIP (Y) ≤ (A)	JUMP A POS	JUMP, STOP KEY 6	JUMP A POS	JUMP, STOP KEY 6	" 3
7	SKIP (Y) > (A)	JUMP A NEG	JUMP, STOP KEY 7	JUMP A NEG	JUMP, STOP KEY 7	" 4

NOTE 1: REPEATED INSTR UNMODIFIED
NOTE 2: ADV EXC ADDRESS EACH EXC
NOTE 3: BACK EXC ADDRESS EACH EXC
NOTE 4: ADD $(B)_b$ EACH EXC

DESIGNATORS

	j	k			
		READ	STORE	REPLACE READ	REPLACE STORE
0	NO SKIP	$(U)_L \rightarrow X_L, 0 \rightarrow X_U$	$(X) \rightarrow Q$	NOT USED	
1	SKIP	$(Z)_L \rightarrow X_L, 0 \rightarrow X_U$	$(X)_L \rightarrow Y_L$	$(Z)_L \rightarrow X_L$	$(X)_L \rightarrow Y_L$
2	SKIP Q POS	$(Z)_U \rightarrow X_L, 0 \rightarrow X_U$	$(X)_L \rightarrow Y_U$	$(Z)_U \rightarrow X_L$	$(X)_L \rightarrow Y_U$
3	SKIP Q NEG	$(Z) \rightarrow X$	$(X) \rightarrow Y$	$(Z) \rightarrow X$	$(X) \rightarrow Y$
4	SKIP A = 0	$(U)_L \rightarrow X_L, (U)_{14} \rightarrow X_U$	$(X) \rightarrow A$	NOT USED	
5	SKIP A ≠ 0	$(Z)_L \rightarrow X_L, (Z)_{14} \rightarrow X_U$	$(X)'_L \rightarrow Y_L$	$(Z)_L \rightarrow X_L, (Z)_{14} \rightarrow X_U$	$(X)_L \rightarrow Y_L$
6	SKIP A POS	$(Z)_U \rightarrow X_L, (Z)_{29} \rightarrow X_U$	$(X)'_L \rightarrow Y_U$	$(Z)_U \rightarrow X_L, (Z)_{29} \rightarrow X_U$	$(X)_L \rightarrow Y_U$
7	SKIP A NEG	$(A) \rightarrow X$	$(X)' \rightarrow Y$	NOT USED	

SECTION F2

MACHINE CODE CHART

(for AN/USQ-20 COMPUTERS)

NTDS UNIT COMPUTER

AN/USQ-20

Repertoire of Instructions

01	**Right SHift • Q**	Shift (Q) Right by **Y**
02	**Right SHift • A**	Shift (A) Right by **Y**
03	**Right SHift • AQ**	Shift (AQ) Right by **Y**
04*	**COMpare • A, • Q, • AQ**	Sense (j); $(A)_j = (A)_f$
05	**Left SHift • Q**	Shift (Q) Left by **Y**
06	**Left SHift • A**	Shift (A) Left by **Y**
07	**Left SHift • AQ**	Shift (AQ) Left by **Y**
10	**ENTer • Q**	Y → Q
11	**ENTer • A**	Y → A
12	**ENTer • B^n**	Y → B^j
13^	**EXternal - FunCTion • C^n**	$\hat{j} \neq 0$ or 1, (Y) → $C^{\hat{j}}$, $\hat{j}$=0 or 1, See Note.
14	**STоRe • Q**	(Q) → Y; k=0, Q' → Q
15	**STоRe • A**	(A) → Y; k=4, A' → A
16	**STоRe • B^n**	$(B)^j$ → Y
17^	**STоRe • C^n**	$(C)^{\hat{j}}$ → Y
20	**ADD • A**	(A) + Y → A
21	**SUBtract • A**	(A) − Y → A
22	**MULtiply**	(Q) Y → AQ
23*	**DIVide**	(AQ) / Y → Q. R → A_f
24	**RePLace • A+Y**	(A) + (Y) → Y & A
25	**RePLace • A−Y**	(A) − (Y) → Y & A
26*	**ADD • Q**	(Q) + Y → Q. $(A)_j = (A)_f$ } j interpretation
27*	**SUBtract • Q**	(Q) − Y → Q. $(A)_j = (A)_f$ } reversed for A&Q
30	**ENTer • Y+Q**	Y + (Q) → A
31	**ENTer • Y−Q**	Y − (Q) → A
32	**STоRe • A+Q**	(A) + (Q) → Y & A
33	**STоRe • A−Q**	(A) − (Q) → Y & A
34	**RePLace • Y+Q**	(Y) + (Q) → Y & A
35	**RePLace • Y−Q**	(Y) − (Q) → Y & A
36	**RePLace • Y+1**	(Y) + 1 → Y & A
37	**RePLace • Y−1**	(Y) − 1 → Y & A
40*	**ENTer • LP****	L[Y(Q)] → A; j=2, even parity; j=3, odd parity
41	**ADD • LP**	L[Y(Q)] + (A) → A
42	**SUBtract • LP**	(A) − L[Y(Q)] → A
43	**COMpare • MASK**	(A) − L[Y(Q)] SENSE (j), (A) + L[Y(Q)]; $(A)_j = (A)_f$
44*	**RePLace • LP**	L(Y)(Q) → Y & A; j=2, even parity; j=3, odd parity
45	**RePLace • A+LP**	L(Y)(Q) + (A) → Y & A
46	**RePLace • A−LP**	(A) − L(Y)(Q) → Y & A
47	**STоRe • LP**	L(A)(Q) → Y; $(A)_j = (A)_f$
50	**SELective • SET**	SET $(A)_n$ FOR Y_n = 1
51	**SELective • CP****	COMPLEMENT $(A)_n$ FOR Y_n = 1
52	**SELective • CL****	CLEAR $(A)_n$ FOR Y_n = 1
53	**SELective • SU****	Y_n → $(A)_n$ FOR $(Q)_n$ = 1
54	**Replace SElective • SET**	SET $(A)_n$ FOR $(Y)_n$ = 1, → Y & A
55	**Replace SElective • CP**	COMPLEMENT $(A)_n$ FOR $(Y)_n$ = 1, → Y & A
56	**Replace SElective • CL**	CLEAR $(A)_n$ FOR $(Y)_n$ = 1, → Y & A
57	**Replace SElective • SU**	$(Y)_n$ → $(A)_n$ FOR $(Q)_n$ = 1, → Y
60	**JumP** (arithmetic)	} Jump to Y if j-condition is satisfied.
61	**JumP** (manual)	} (see JP & RJP j - Designators)
62^	**JumP** (if • C^n has **ACTIVE INput buffer**)	Jump to Y if $C^{\hat{j}}$ input buffer active } (see JP & RJP j - Designators)
63^	**JumP** (if • C^n has **ACTIVE OUTput buffer**)	Jump to Y if $C^{\hat{j}}$ output buffer active } (see JP & RJP j - Designators)
64	**Return JumP** (arithmetic)	} Jump to Y+1 and P+1 → Y_L if j condition is satisfied (see JP & RJP j - Designators)
65	**Return JumP** (manual)	}
66^	**TERMinate • C^n • INPUT**	Terminate input buffer on channel $\hat{j}$
67^	**TERMinate • C^n • OUTPUT**	Terminate output buffer on channel $\hat{j}$
70*	**RePeaT**	Execute NI Y times
71	**BSKip • B^n**	$(B)^j$ = Y, skip NI and clear $(B)^j$; $(B)^j \neq$ Y, Advance B^j and read NI
72	**BJumP • B^n**	$(B)^j$ = 0, read NI; $(B)^j \neq 0$, $(B)^j$ − 1 and jump to address Y
73^	**INput • C^n** (without monitor mode)	Buffer IN on $C^{\hat{j}}$; $\hat{k}$ = 3, (Y) → $(00100+\hat{j})$; $\hat{k}$ = 1, $(Y)_L$ → $(00100+\hat{j})_L$; $\hat{k}$ = 0, Y → $(00100+\hat{j})_L$.
74^	**OUTput • C^n** (without monitor mode)	Buffer OUT on $C^{\hat{j}}$; $\hat{k}$ = 3, (Y) → $(00120+\hat{j})$; $\hat{k}$ = 1, $(Y)_L$ → $(00120+\hat{j})_L$; $\hat{k}$ = 0, Y → $(00120+\hat{j})_L$.
75^	**INput • C^n** (with • **MONITOR** mode)	Buffer IN on $C^{\hat{j}}$ with mon. $\hat{k}$ = 3, (Y) → $(00100+\hat{j})$; $\hat{k}$ = 1, $(Y)_L$ → $(00100+\hat{j})_L$; $\hat{k}$ = 0, Y → $(00100+\hat{j})_L$. mon. inter. at $00040+\hat{j}$
76^	**OUTput • C^n** (with • **MONITOR** mode)	Buffer OUT on $C^{\hat{j}}$ with mon. $\hat{k}$ = 3, (Y) → $(00120+\hat{j})$; $\hat{k}$ = 1, $(Y)_L$ → $(00120+\hat{j})_L$; $\hat{k}$ = 0, Y → $(00120+\hat{j})_L$. mon. inter. at $00060+\hat{j}$

— **NO - OPeration**
— **ComPlement • A** or **• Q**
— **CLear • A, • Q, • B^n**, or **Y**
— **Remove Interrupt Lockout**
— **Remove Interrupt Lockout and JumP • Y**
— **TEST • CO** or **• CI**

} CS−1 Mono - codes

****LP** - Logical Product **CP** - Complement **SU** - Substitute **CL** - Clear ***^** } Special j & k Designators (see opposite side of card) **Y** - The operand; Y or (Y)

NOTE: Skip NI if other Computer (on channel 0 or 1) has input buffer active. Execute twice.

NTDS UNIT COMPUTER

AN/USQ-20 *Repertoire of Instructions*

JP & RJP j-DESIGNATORS

j	JP 60	RJP 64	JP 61	RJP 65
0	(No Jump)*		(Uncond. Jump)	
1	(Uncond. Jump)*		KEY 1	
2	Q POS		KEY 2	
3	Q NEG		KEY 3	
4	A ZERO		STOP	
5	A NOT Zero		STOP 5	
6	A POS		STOP 6	
7	A NEG		STOP 7	
$\hat{j}$	62 $\hat{j}$		63 $\hat{j}$	
$0\text{-}15_8$	C^n ACTIVE IN		C^n ACTIVE OUT	

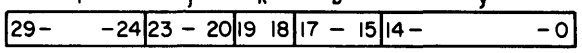
*60 Clears interrupt & bootstrap modes.

$\hat{j}$-DESIGNATORS

(4 bits)

$\hat{j}$ Occupies 4 bit positions and represents C^n where n may be $0—15_8$

The instruction word assumes the format:



$\hat{k}$-DESIGNATORS

(2 bits)

$\hat{k}$	EX–FCT 13	STR•C^n 17	JP 62 63	IN•C^n, OUT•C^n 73 75 74 76
0	'not used'	'not used'	'blank'	'blank'
1	'not used'	'not used'	L	L
2	'not used'	'not used'	U	'not used'
3	W	W	W	W

* j-DESIGNATORS

j	COM•A, •Q, •AQ 04	DIV 23	ADD•Q, SUB•Q 26 27	ENT•LP, RPL•LP 40 44	RPT 70	
0	(no skip)	(no skip)	(no skip)	(no skip)	(no mod.)	Y of NE = Y
1	(unconditional skip)	SKIP	SKIP	SKIP	ADV	Y of NE = Y+1
2	Y LESS ¦ Y ≤ (Q)	NO Over Flow	A POS	EVEN parity	BACK	Y of NE = Y−1
3	Y MORE ¦ Y > (Q)	Over Flow	A NEG	ODD parity	ADD B	Y of NE = Y+B^b
4	Y IN ¦ (Q) ≥ Y and Y > (A)	A ZERO	Q ZERO	A ZERO	Rpl. Inc.	Y of NE = Y $[+B^6]$ ✓
5	Y OUT ¦ (Q) < Y or Y ≤ (A)	A NOT Zero	Q NOT Zero	A NOT Zero	ADV R	Y of NE = Y+1 $[+B^6]$ ✓
6	Y LESS ¦ Y ≤ (A)	A POS	Q POS	A POS	BACK R	Y of NE = Y−1 $[+B^6]$ ✓
7	Y MORE ¦ Y > (A)	A NEG	Q NEG	A NEG	ADD B R	Y of NE = Y+B^b $[+B^6]$ ✓

✓ B^6 Increment if NI is RPL class; increments Y address for the store portion of the replace.

NE — Next execution

NORMAL j-DESIG.

j	(Not applicable on * or ^) Skip Code
0	(no skip)
1	SKIP
2	Q POS
3	Q NEG
4	A ZERO
5	A NOT Zero
6	A POS
7	A NEG

NORMAL k-DESIGNATORS

k	READ Code	READ Origin	STORE Code	STORE Dest.	REPLACE Code	REPLACE Origin	REPLACE Dest.
0	'blank'	U_L	Q	Q	'not used'	—	—
1	L	M_L	L	M_L	L	M_L	M_L
2	U	M_U	U	M_U	U	M_U	M_U
3	W	M	W	M	W	M	M
4	X	XU_L	A	A	'not used'	—	—
5	LX	XM_L	CPL	Cpl M_L	LX	XM_L	M_L
6	UX	XM_U	CPU	Cpl M_U	UX	XM_U	M_U
7	A	A	CPW	Cpl M	'not used'	—	—

LEGEND

M — Memory word (30 bits)
M_L — Lower half memory word
M_U — Upper half memory word
X — Sign bit extended
Cpl — Complement
A — A-register
Q — Q-register
U — U-register

SECTION F3

EXCESS-THREE CODE

FOR THE HIGH-SPEED PRINTER

EXCESS-THREE CODE FOR THE HIGH-SPEED PRINTER

CHARACTER	PARITY BIT	CODE ZONE	CODE BODY	COMP DIGIT	CHARACTER	PARITY BIT	CODE ZONE	CODE BODY	COMP DIGIT
(IGNORE)	1	00	0000	5	(IGNORE)	0	10	0000	N
(IGNORE)	0	00	0001	6	(IGNORE)	1	10	0001	O
-	0	00	0010		(FF II)	1	10	0010	P
0	1	00	0011		)	0	10	0011	
1	0	00	0100		J	1	10	0100	
2	1	00	0101		K	0	10	0101	
3	1	00	0110		L	0	10	0110	
4	0	00	0111		M	1	10	0111	
5	0	00	1000		N	1	10	1000	
6	1	00	1001		O	0	10	1001	
7	1	00	1010		P	0	10	1010	
8	0	00	1011		Q	1	10	1011	
9	1	00	1100		R	0	10	1100	
'	0	00	1101		$	1	10	1101	
&	0	00	1110		*	1	10	1110	
(	1	00	1111		(FF III)	0	10	1111	M
(ML)	0	01	0000	E	(STOP)	1	11	0000	V
,	1	01	0001		(BP)	0	11	0001	W
.	1	01	0010		:	0	11	0010	
;	0	01	0011		+	1	11	0011	
A	1	01	0100		/	0	11	0100	
B	0	01	0101		S	1	11	0101	
C	0	01	0110		T	1	11	0110	
D	1	01	0111		U	0	11	0111	
E	1	01	1000		V	0	11	1000	
F	0	01	1001		W	1	11	1001	
G	0	01	1010		X	1	11	1010	
H	1	01	1011		Y	0	11	1011	
I	0	01	1100		Z	1	11	1100	
#	1	01	1101		%	0	11	1101	
(IGNORE)	1	01	1110	C	(FF IV)	0	11	1110	T
(FF I)	0	01	1111	D	UNUSED	1	11	1111	

NOTE: Non-printable symbols are indicated in the CHARACTER column by parentheses. The alternate symbols which will be printed in the Computer Digit mode of operation are listed in the COMP DIGIT column.

SECTION F4

FLEXOWRITER CODE CHART

FLEXOWRITER CODE CHART

The upper case, *UC*, or lower case, *LC*, character is typed according to the position of the type bars.

Type Letter *UC*	Type Letter *LC*	Octal
A	a	30
B	b	23
C	c	16
D	d	22
E	e	20
F	f	26
G	g	13
H	h	05
I	i	14
J	j	32
K	k	36
L	l	11
M	m	07
N	n	06
O	o	03
P	p	15
Q	q	35
R	r	12
S	s	24
T	t	01
U	u	34
V	v	17
W	w	31
X	x	27
Y	y	25
Z	z	21

Type Letter *UC*	Type Letter *LC*	Octal
!	1	52
@	2	74
#	3	70
$	4	64
%	5	62
¢	6	66
&	7	72
*	8	60
(	9	33
)	0	37

Perform Typewriter Operation	Octal
Space	04
Shift up	47
Shift down	57
Back space	61
Car. return	45
Tabulator	51
Color shift	02
Code delete	77*
Stop	43*

Type Symbol *UC*		Type Symbol *LC*		Octal
-	(Superscript Minus)	-	(Hyphen or Minus)	56
.	(Multiply)	=	(Equals)	44
/	(Virgule)	+	(Plus)	54
(	(Open Parens)	,	(Comma)	46
)	(Close Parens)	.	(Period)	42
_	(Underline)	|	(Absolute)	50

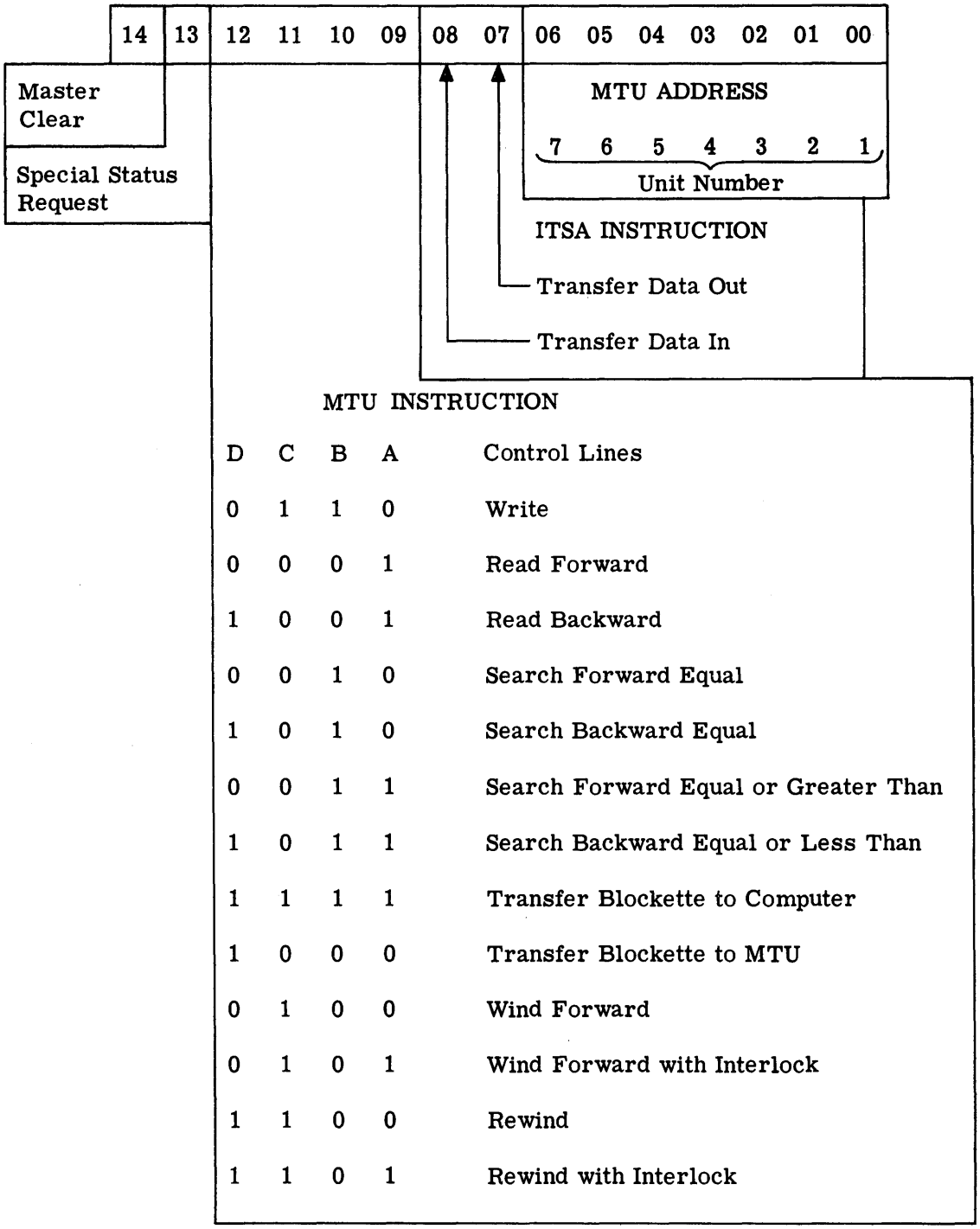
NOTE: Codes not used are: 00, 10, 40, 41, 53, 55, 63, 65, 67, 71, 73, 75, 76.
*Codes 43 and 77 are considered illegal codes in operations with the computer.

SECTION F5

EQUIPMENT FUNCTION WORD CHARTS

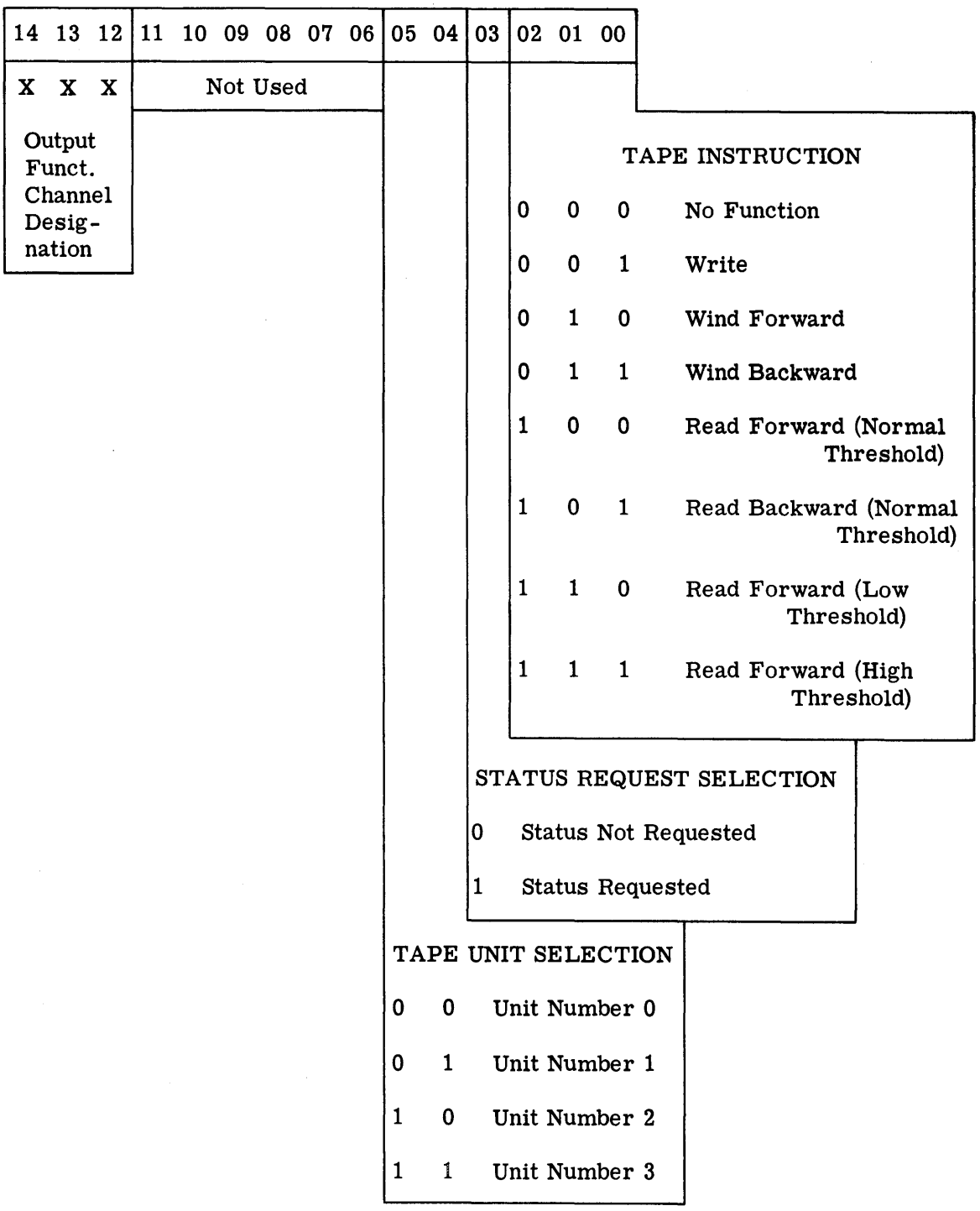
FOR AN/USQ-17 COMPUTERS

EQUIPMENT FUNCTION WORD - AN/USQ-17 COMPUTER TO ITSA

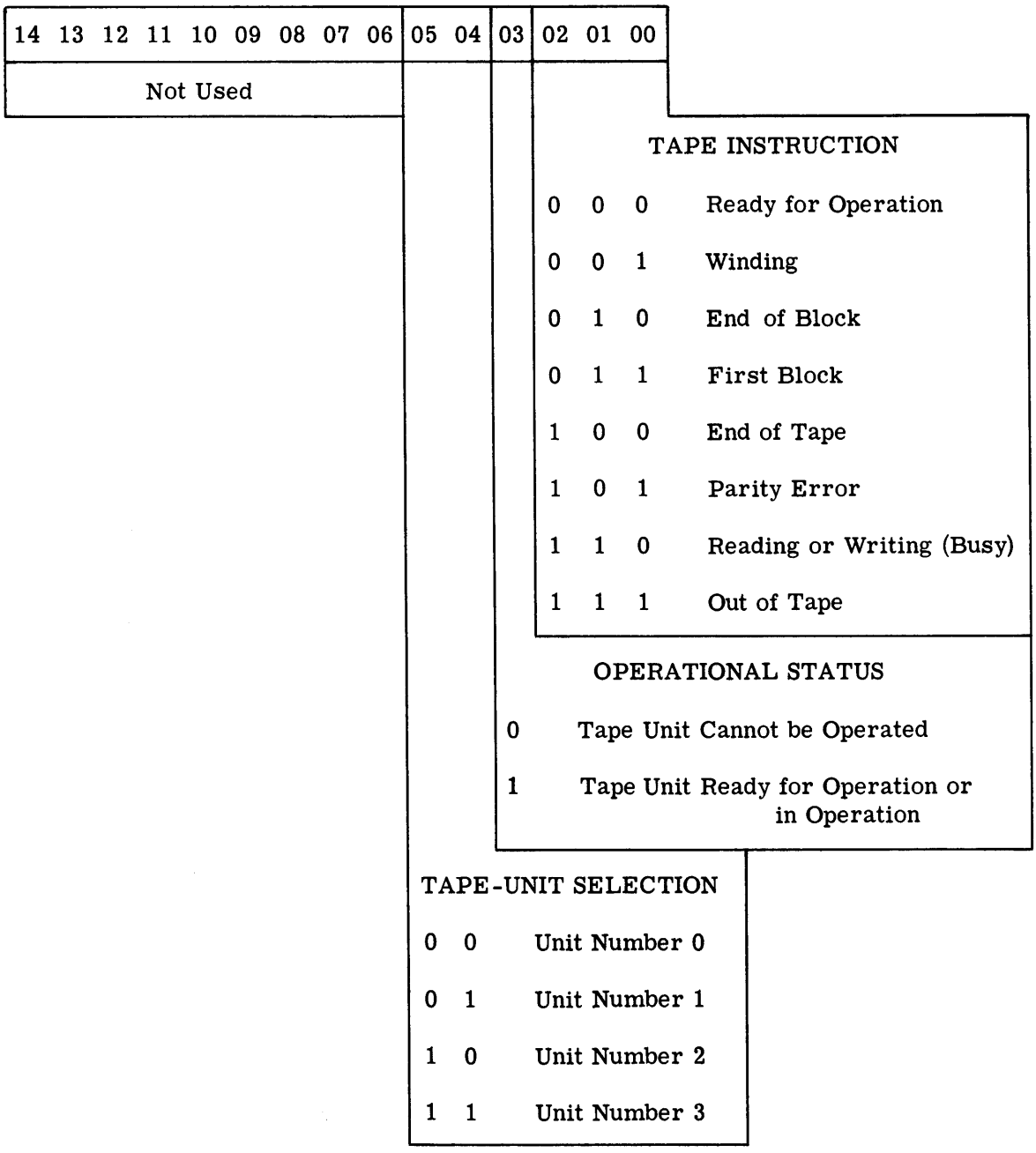


D	C	B	A	Control Lines
0	1	1	0	Write
0	0	0	1	Read Forward
1	0	0	1	Read Backward
0	0	1	0	Search Forward Equal
1	0	1	0	Search Backward Equal
0	0	1	1	Search Forward Equal or Greater Than
1	0	1	1	Search Backward Equal or Less Than
1	1	1	1	Transfer Blockette to Computer
1	0	0	0	Transfer Blockette to MTU
0	1	0	0	Wind Forward
0	1	0	1	Wind Forward with Interlock
1	1	0	0	Rewind
1	1	0	1	Rewind with Interlock

EQUIPMENT FUNCTION WORD - AN/USQ-17 COMPUTER TO POTTER TAPE UNIT



EQUIPMENT FUNCTION WORD (Status Reply) - POTTER TAPE UNIT
TO AN/USQ-17 COMPUTER



EQUIPMENT FUNCTION WORD - AN/USQ-17 COMPUTER TO HIGH-SPEED PRINTER VIA HIGH-SPEED PRINTER ADAPTER

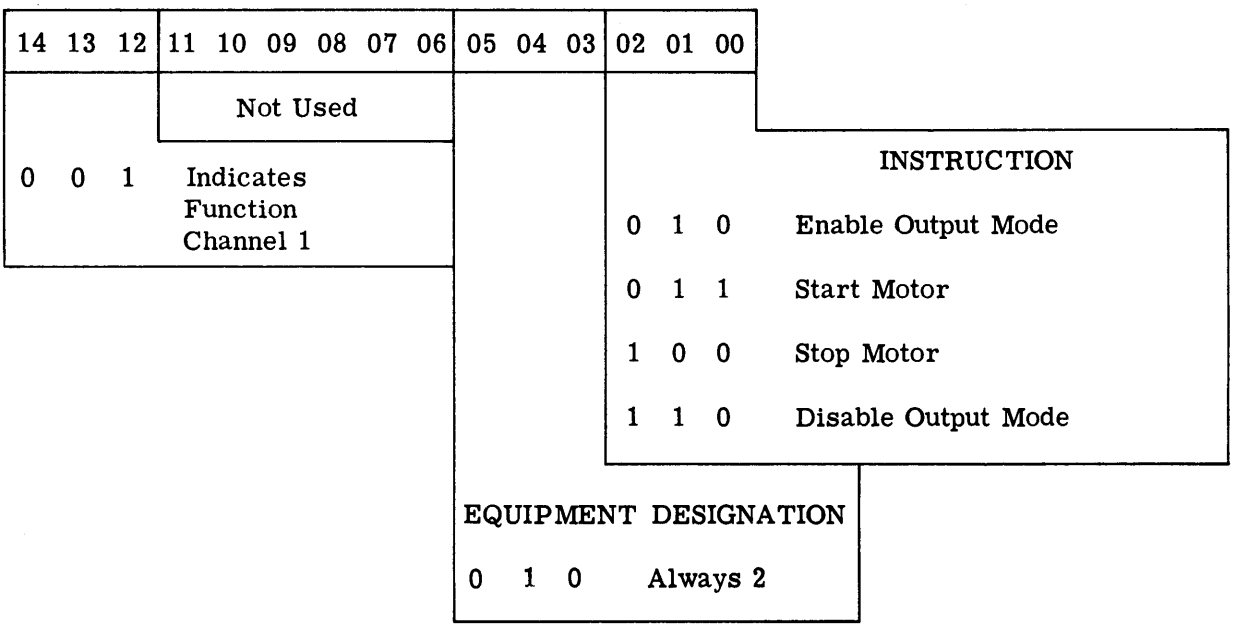
14 13 12	11 10 09 08 07 06	05 04 03 02 01 00
Function Channel	Not Used	FUNCTION CODE

Clear Adapter and High-Speed Printer Circuits	0	1	1	0	0	1
Clear Printer Memory and Enable Data Transfer Circuits	0	1	1	1	0	0

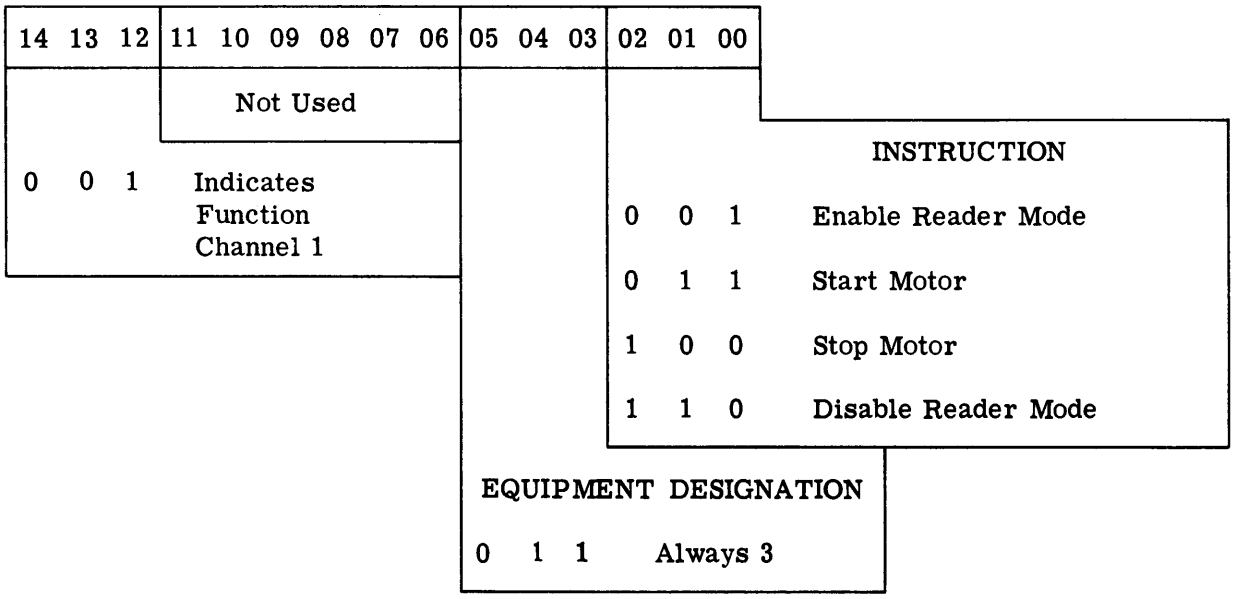
EQUIPMENT FUNCTION WORD - AN/USQ-17 COMPUTER TO MONITORING TYPEWRITER

14 13 12	11 10 09 08 07 06	05 04 03	02 01 00
	Not Used		INSTRUCTION
0 0 1 Indicates Function Channel 1			0 0 1 Enable Input Mode 0 1 0 Enable Output Mode 0 1 1 Start Motor 1 0 0 Stop Motor 1 1 0 Disable Input or Output Mode
		EQUIPMENT DESIGNATION 0 0 1 Always 1	

EQUIPMENT FUNCTION WORD - AN/USQ-17 COMPUTER TO HIGH-SPEED PUNCH



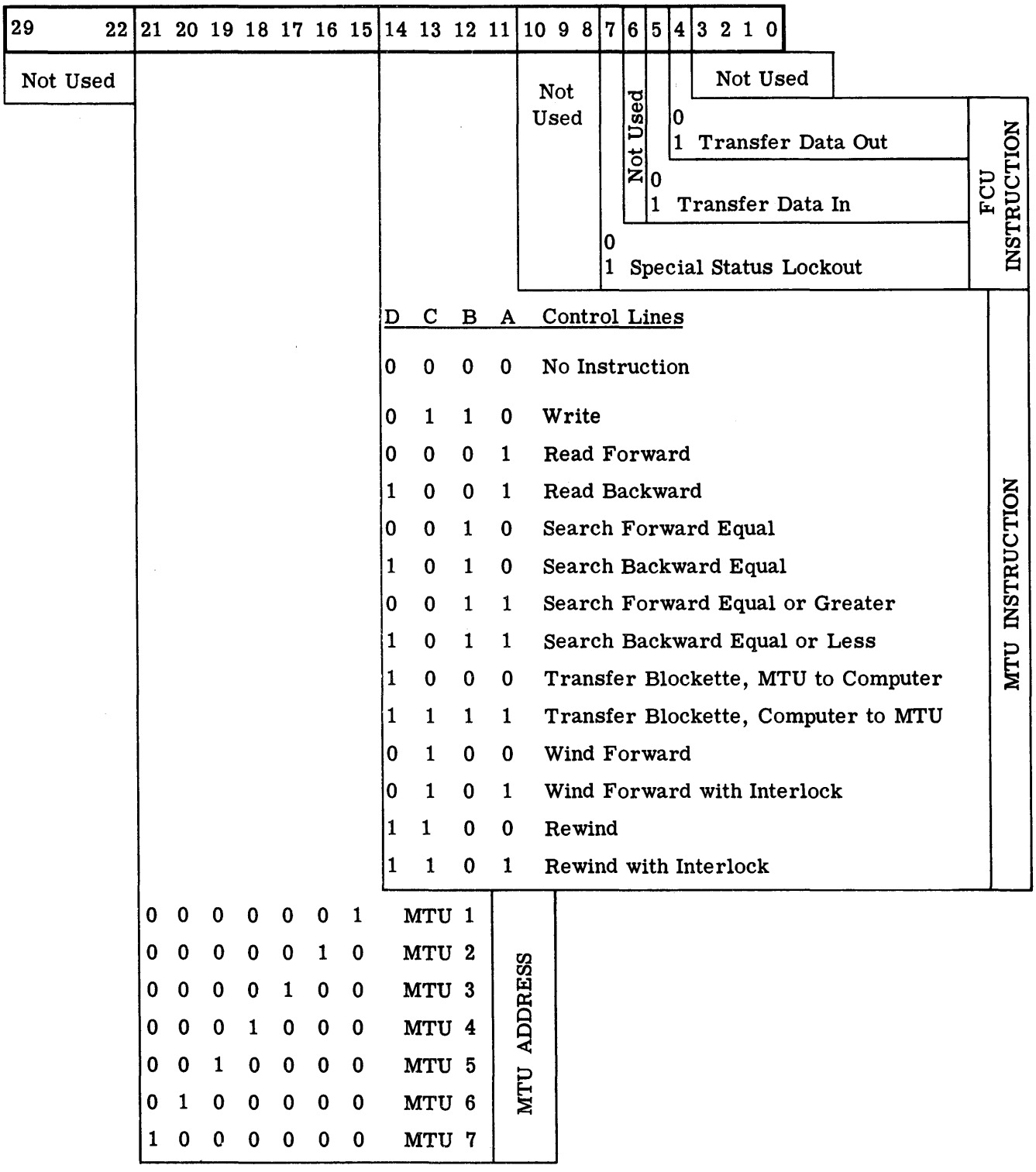
EQUIPMENT FUNCTION WORD - AN/USQ-17 COMPUTER TO PHOTOELECTRIC TAPE READER



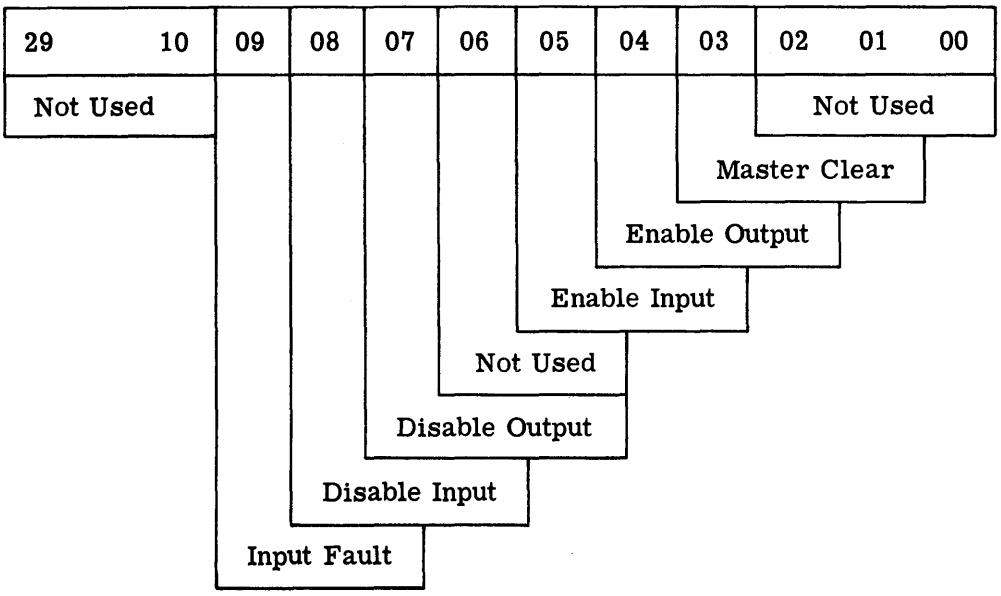
SECTION F6

EQUIPMENT FUNCTION WORD CHARTS FOR AN/USQ-20 COMPUTERS

EQUIPMENT FUNCTION WORD - AN/USQ-20 COMPUTER TO FORMAT CONTROL UNIT

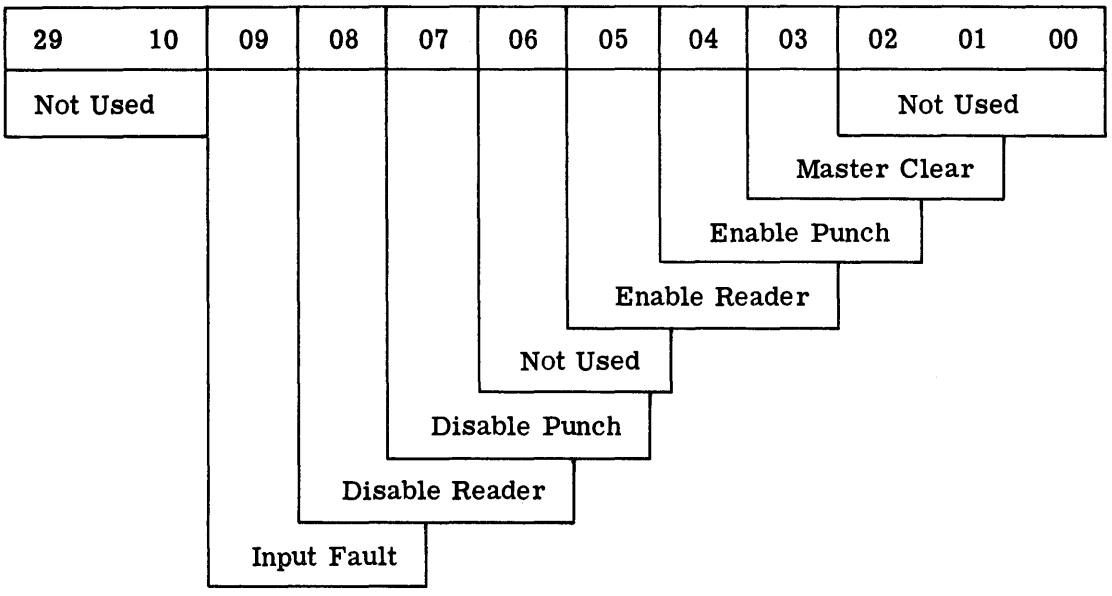


EQUIPMENT FUNCTION WORD - AN/USQ-20 COMPUTER TO MONITORING TYPEWRITER



A one in the bit position indicates desired function

EQUIPMENT FUNCTION WORD - AN/USQ-20 COMPUTER TO HIGH-SPEED PUNCH



SECTION F7

CS-1 COMPILER CODE CARD

PROBLEM-ORIENTED PROGRAMMING OPERATIONS

LABEL	OPERATOR	OPERANDS
[label]	SYSTEM	• [programmer's name] • [date]
	COMMON-B	• [No. of B-register] • [data name]
	SEL-DD	• [label of SYS-DD] } See CS-1 Librarian
	SEL-PROC	• [label of SYS-PROC, key] • [label, key - of each non-unique label] } See CS-1 Librarian
	SEL-SYS	• [key] • [label, key - of each non-unique label] } See CS-1 Librarian
[label]	SYS-DD	• [programmer's name] • [date]
	END-SYS-DD	
[label]	SYS-PROC	• [programmer's name] • [date]
	LOC-DD	
	TABLE	• [name] (2) • H (or V) • [words/item] • [max. items] • [name maj. index] (1)
	SUB-TABLE	• [name] • [initial item no.] • [max. items] • [name maj. index] (1)
	FIELD	• [name] • FXPOS (or FXWS,MW) • [word loc.] • [no. words or bit pos] • [binary point] (1)
	ITEM-AREA	• [name] • • • •
	END-TABLE	• [name] (2)
	VRBL	• [name] • FXW (or FXH) • [binary point] (1)
	SWITCH	• [name] • [statement label] (3) • • • •
	S	• [statement label] (4)
	END-SWITCH (4)	• [name]
	END-LOC-DD	
	PROCEDURE	• [name] • INPUT (1) • [formal name(s)] • OUTPUT (1) • [formal name(s)] • EXIT (1) • [formal name(s)]
	B-REG-P	• [B-reg no. (1-6)] • • • •
	B-REG-T	• [B-reg no. (1-6)] • • • •
	INDEX	• [name] • • • •
	SET	• [data name] • TO (or EQ) • [data name, const., alg. exp.]
	GOTO	• [statement label]
	GOTO	• [switch name] • [switch setting]
	IF	• [data name] • [decider] (5) • [data name, const., alg. exp.] • THEN • • • •
	IF	• DATA • FOUND (or NOT FOUND) • THEN • • • •
	IF	• DATA • VALID (or INVALID) • THEN • • • •
[label]	VARY	• [data name] • [prepositional operand] (6)
	RESUME	• [FIND or VARY label]
	END	• [VARY label]
[label] (7)	FIND	• [data name] • [decider] (5) • [data name, const, alg. exp.] • VARYING (1)* • [prepositional operand] (1)*,(6)
	RETURN	• [formal statement label] (1) • STOP (or STOP 5, 6, 7) (1)
	[procedure name]	• INPUT (1) • [data name, const., alg. exp.] • OUTPUT (1) • [data name(s)] • EXIT (1) • [statement label(s)]
	TYPE	• [data name] • • • [data name] • • • [CR] • • • [TAB] • • • THEN (1) • • • •
	TYPE-TEXT	• [text and flex commands]
	PUNCH	• [data name] • • • [data name] • • • [CR] • • • [TAB] • • • THEN (1) • • • •
	PUNCH-TEXT	• [text and flex commands]
	FORM	• [buffer name] • [initial char. position] • [data name] • THEN (1) • • • •
	FORM-TEXT	• [buffer name] • [initial char. position] • [text • • •] • THEN (1) • • • •
	PRINT-BUF	• [base addr.] • [jump cond]
	PRINT-TBL	• [data name] • • • [data name] • THEN (1) • • • •
	PUT-ADR	• [data name] • IN • [data or reg.name] • THEN (1) • • • •
	COMMENT	• [message]
	STOP	
	END-PROC	• [name]

LIST OF DECIDERS

CODE	MEANING
• EQ •	= Equal
• NOT •	≠ Not Equal
• LTEQ •	≤ Less Than or Equal To
• LT •	< Less Than
• GT •	> Greater Than
• GTEQ •	≥ Greater Than or Equal To

LIST OF PREPOSITIONAL OPERANDS

CODE	OBJECT PRESCRIBES	OBJECT FORMS PERMITTED
• FROM •	Starting Point	Data Name, Alg. Exp., Constant, Index
• THRU •	Terminal Point	Data Name, Alg. Exp., Constant, Index
• BY •	Index Increment	Data Name, Alg. Exp., Constant, Index
• WITHIN •	Table or Sub-Table Parameters	Table or Sub-Table Name

LEGEND

(1) Use is optional
(1)* Optional if varied throughout table
(2) Limited to 5 alphanumeric characters
(3) Not used in switch table design
(4) Used in switch table design
(5) See list of deciders
(6) See list of prepositional operands
(7) Required if return made by RESUME or GOTO
(8) Reserves B-register for data unit throughout program

COMPILER-CONTROL OPERATIONS

LABEL	OPERATOR	OPERANDS	COMMENTS
[label]	➡ C-CONTROL	• [programmer's name] • [date]	General Header for Compiler Control Operators
	➡ AN/USQ-20 (or AN/USQ-17)		Specifies for which computer object program is to be compiled
	➡ P-TRACE	• [procedure name] • [procedure name] • • • •	Specifies procedures whose linking operations are to be traced
	➡ P-IGNORE	• [procedure name] • [procedure name] • • • •	Specifies procedures (of a procedure chain) not to be compiled
	➡ DEBUG-AIDS		Informs compiler that Debugging Aids are desired
	➡ OUTPUTS	• [output no.] • [output no.] • • • •	Informs compiler which outputs are desired in object program
	➡ CHAN-SET		Operator heading communications channel assignments
[label]	➡ [input/output assignment]		Specifies desired communication channel assignments
	➡ ALLOCATION		Operator heading normal allocation instructions
BASE	➡ [absolute address]		Specifies Initial Address for compiler allocation of L_4 object program
ENTRANCE	➡ [primary procedure name or S/R label]		Generates manual entrance at object program's base address
DEBUG	➡ [absolute address]		Specifies initial address for Debugging Package when not placed at 76000
TABLE POOL	➡ [absolute address]		Specifies initial POOL address for table allocation
[label]	➡ [numeric allocation value, label, tag]		Direct allocation instruction format
[label]	➡ DELETE		Deletes indicated label from compliers allocation tables
	➡ INDR-ALLOC		Operator heading indirect allocation instructions
[S/R label]	➡ [6 digit number, k and y]		Specifies memory cell (k-designator and address) containing initial address of subroutine
	➡ REL-ALLOC		Operator heading relative allocation instructions
[label]	➡ [increment alloc. value]		Specifies increment to a given base address (place new base in B^7 while loading)

DEBUGGING OPERATIONS

➡ DEF-AREA • [area name] • [initial area tag, label, address] • [no. of words]
➡ CORE-IMAGE • [area name] • [initial image tag, label, address] • KEY 1 (or 2, 3)(1)
➡ DRUM-IMAGE • [area name] • [initial image tag, label, address] • KEY 1 (or 2, 3)(1)
➡ TEST-IMAGE • [area name] • • • • KEY 1 (or 2, 3)(1)
➡ DUMP-REG • KEY 1 (or 2, 3)(1)
➡ DUMP-AREA • [area name] • • • • KEY 1 (or 2, 3)(1)

(1) Use is optional

PROGRAM CORRECTION OPERATIONS

[label]	➡ CORRECT - L_1 • [programmer's name] • [date]	Operator heading list of program corrections
[L_1 ID • Ins No]	↙	Specifies L_1-ID with insert number for insert position
	➡ [insert operation]	Indicates insert to be made
[L_1 ID]	↙	Specifies L_1-ID for correction position
	➡ DELETE (or Replacement Operation)	Indicates correction to be made

LIBRARY UPDATING OPERATIONS

[label] ➡ LIBRARY
➡ INS-DD • [label] • [programmer's name] • [date]
➡ RPL-DD • [label] • [programmer's name] • [date]
➡ DEL-DD • [label] • [programmer's name] • [date]
➡ INS-PROC • [label], [key] • [programmer's name] • [date]
➡ RPL-PROC • [label], [key] • [programmer's name] • [date]
➡ DEL-PROC • [label], [key] • [porgrammer's name] • [date]

LIBRARY LISTING OPERATIONS

[label] ➡ LIBRARY
➡ LIST-PROC • [label], [key]
➡ LIST-DIR
➡ LIST-DIR • [directory number] • • • •
➡ LIST-DD • LIBRARY
➡ LIST-PROC • LIBRARY
➡ LIST-DD • [label]
➡ LIST-DD • HISTORY