Digital Equipment Corporation
Maynard, Massachusetts

digital

PDP-15 Systems

User's Handbook
Vol. 2 Peripherals

DEC-15-H2DB-D

PDP-15 SYSTEMS
USER'S HANDBOOK
VOLUME 2 PERIPHERALS

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

1st Edition September 1970
2nd Edition (Rev) November 1970

CONTENTS

CONTENTS (Cont)

CONTENTS (Cont)

CONTENTS (Cont)

CONTENTS (Cont)

ILLUSTRATIONS

ILLUSTRATIONS (Cont)

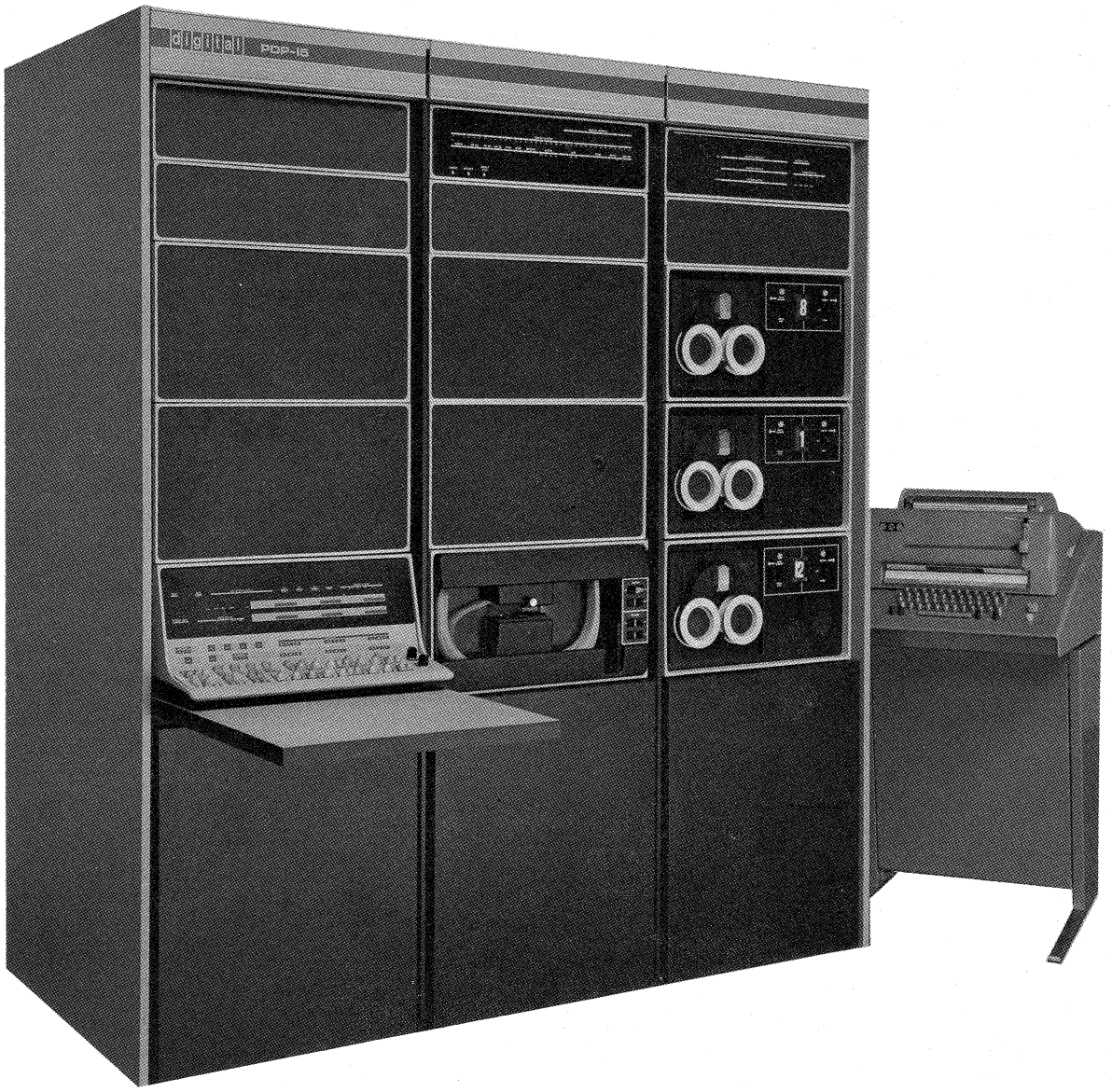
TABLES

FOREWORD

This volume covers standard PDP-15 system peripherals.

User's information on other peripherals used with the PDP-15 system is included in the appropriate peripheral maintenance manuals.

PDP-15 USER'S HANDBOOK
VOLUME II, PERIPHERALS


DIGITAL
PDP-15

Chapter 1
PC15 High-Speed Paper-Tape Reader Punch

1.1 INTRODUCTION

The PC15 High-Speed Paper Tape Reader/Punch is used to input perforated paper-tape programs into core memory, or to punch core memory programs or data on paper tape. Information is punched on 8-channel fanfolded paper tape in the form of 6- or 8-bit characters at a maximum rate of 50 characters/second. Information is read at a maximum rate of 300 characters/second. The PC15 consists of a PC05 Paper Tape Reader/Punch with interface and control logic for using the reader/punch with a PDP-15.

1.2 PAPER-TAPE READER

1.2.1 Characteristics and Capabilities

Data can be read from tape and transferred to the PDP-15, using the computer hardware readin logic or using program-controlled transfers. For hardware readin operation, the hardware readin logic supplies inputs for selecting the operating mode, starting tape motion, and implementing transfers. For program-controlled transfers, the computer issues input/output transfer (IOT) instructions that select the operating mode, advance the tape, and implement the transfer. To maintain a maximum rate of 300 characters/second, a new select IOT must be issued within 1.67 ms of the last reader flag. If not, the reader operates start-stop and reads characters at a 25 character/second rate. The requirements for maximum character rate are described in detail in Programming Considerations, Paragraph 1.3.7.

The reader interfaces with the automatic priority interrupt (API) facility, the program interrupt facility, and the input/output skip chain. For API operation, the reader is assigned API level 2; a unique entry address of 50_8 is assigned to its service routine.

The reader contains a no-tape sensor and flag (character ready for transfer) circuits. If a no-tape condition is detected, the reader flag is set, and a program interrupt is initiated whenever a reader select IOT is given. The states of the reader flag, the reader API 2 level, PI request and skip request

devices are displayed on an indicator panel at the top of Cabinet H963E (Bay 1R). In addition, this panel displays the reader buffer contents and the I/O address (API unique entry address). These items and the reader controls are described in Controls and Indicators, Paragraph 1.2.3.

Reader mechanical facilities include a right-hand bin for supply for tape being read, a left-hand bin for receiving the tape, and a feed-through mechanism to control passage of the tape into the receiving bin. A snap-action retainer on the feed-through mechanism facilitates simple loading of the tape.

1.2.2 Operating Modes

The PC15 reader operates in either an alphanumeric or binary mode. For program-controlled transfers, the operating mode is selected by IOT instructions. For hardware readin operation, control logic in the reader automatically selects the binary mode.

When alphanumeric mode is selected, one 8-bit character (in ASCII code) is read and transferred to the PDP-15 accumulator. In the binary mode, the reader reads three 6-bit characters (three frames with channels 7 and 8 ignored) from tape and assembles them into an 18-bit word for transfer to the accumulator.

1.2.3 Controls and Indicators

Two front panel controls are provided for the PC15 Paper-Tape Reader: ON LINE/OFF LINE and FEED. The ON LINE position places the reader under computer control. The OFF LINE position, which is used for loading paper tape, raises an out-of-tape flag and places the reader under local control. The indicators associated with reader operation are located on an indicator panel at the top of cabinet H963E (Bay 1R). Table 1-1 lists the indicators and their functions.

Table 1-1
Indicators Associated with Paper-Tape Reader

Indicator	Function
READER BUFFER 00-17	Indicates the contents of the paper-tape reader buffer.
API 2 RDR	Denotes API level 2 is active as the result of a reader interrupt.
I/O ADDRESS	Indicates the unique trap address associated with I/O devices; address 50$_8$ for paper-tape reader.
RDR FLG	Denotes information has been read from tape and is available for transfer from reader buffer.

Table 1-1 (Cont)
Indicators Associated with Paper-Tape Reader

Indicator	Function
PI RQ	Denotes one of the I/O devices (including paper-tape reader) handled by the BA15 Peripheral Expander has generated an interrupt request.
SKIP RQ	Denotes one of the I/O devices (including paper-tape reader) handled by BA15 has responded to a skip IOT instruction.

1.2.4 Tape Formats

The format of the perforated paper tapes for the alphanumeric (ASCII usage) mode is shown in Figure 1-1. In addition, tape channels are related to the PDP-15 accumulator stages. The leader and trailer portions of the tape are used to introduce or conclude a paper-tape program. Only the feed hole is punched for the leader/trailer portions. Note that each character is read by one IOT instruction.

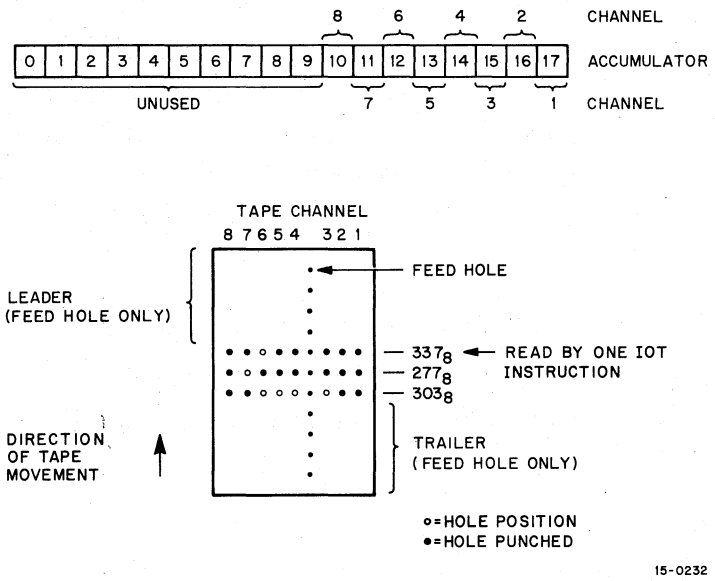


Figure 1-1 Tape Format and Accumulator Bits (Alphanumeric Mode)

The paper-tape format for binary mode using hardware readin (HRI) is shown in Figure 1-2 as well as the relationship of accumulator stages for the 18-bit word. Note that only the feed hole is perforated for the leader/trailer portion and that channel 8 is always punched in the program portion of the tape. Any character without hole 8 punched will be ignored. Channel 7 punched in the last character indicates the last 18-bit instruction is to be executed by the computer. This instruction can halt machine operation or can transfer machine control to another part of the program. When using this format, channel 7 must be punched using the alphanumeric mode.

1.2.5 Instructions

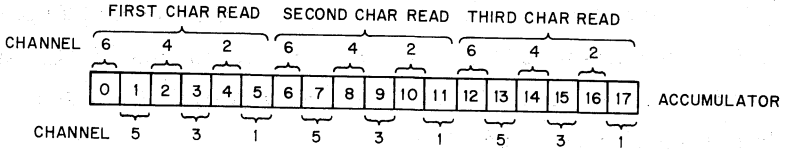
The PDP-15 IOT instructions used for program-controlled loading of paper-tape data are listed below. Refer to Volume 1 of this handbook for IOT instruction format.

Mnemonic	Octal Code	Operation Performed
RSF	700101	Skip next instruction if reader flag is a 1.
RCF	700102	Clear reader flag. Read reader buffer, inclusively OR contents of reader buffer with AC, and deposit result in AC.
RRB	700112	Read reader buffer and clear reader flag. Clear AC and transfer contents of reader buffer to AC.
RSA	700104	Select alphanumeric mode and place one 8-bit character in reader buffer. Clear flag before character is read from tape. Set reader flag to 1 when transfer to reader buffer is complete.
RSB	700144	Select binary modes. Assemble three 6-bit characters in reader buffer. Clear reader flag during assembly and set flag when assembly is complete.

The paper-tape reader responds to an input/output read status (IORS) instruction by supplying the status of its device flags and no-tape flags to the accumulator. The reader device flag (reader interrupt) interfaces with bit 01 of the accumulator. The reader no-tape flag interfaces with bits 08 of the accumulator.

1.2.6 Functional Description

The PC15 reader consists of an electromechanical tape feed system, a light source and photo cells for sensing tape perforations, a buffer register for storing and assembling data, and control logic for computer interface, tape advance, and transfer operations. These circuits can be used with the PDP-15 hardware readin logic, or can be used for program-controlled transfers, as described in the following paragraphs.



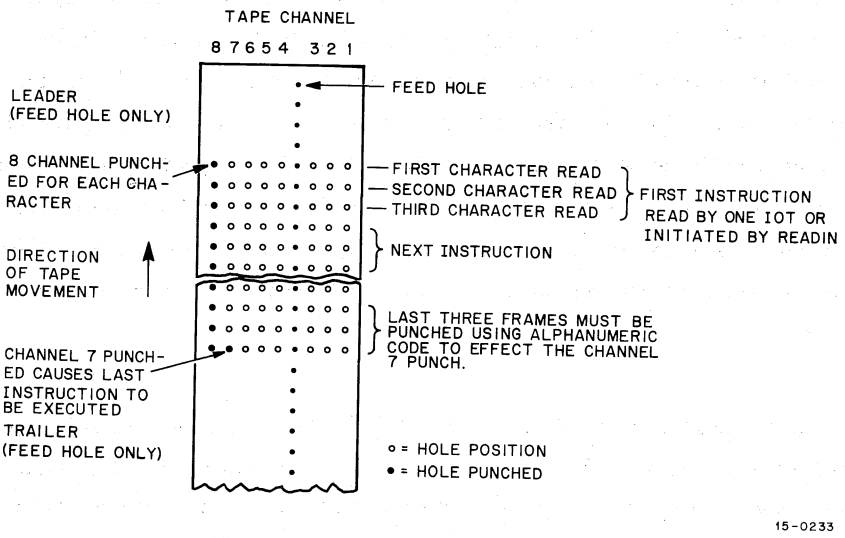


Figure 1-2 HRI Tape Format and Accumulator Bits (Binary Mode)

1.2.6.1 Hardware Readin Operation - The PC15 reader can be used with PDP-15 hardware readin logic to load programs from paper tape at a rate of 300 characters/second. For this operation, the desired tape is installed in the high-speed reader, and the program loading address is selected, using the console ADDRESS switches. The console RESET key is then pressed to initialize the computer and paper-tape reader. A readin operation is started by pressing the READIN key on the console.

With this key action, a Readin (RI) condition is stored in the reader, and the binary mode is selected. The reader then advances the tape, reads three characters from tape, assembles them into an 18-bit word in the reader buffer, and signals the hardware readin logic with a program interrupt. The hardware readin logic, in turn, transfers the 18-bit word to the accumulator under I/O processor and computer timing. The word is subsequently loaded into core memory by forcing a DAC instruction. The first 18-bit word is stored at the address specified by the console ADDRESS switches. Subsequent 18-bit words are stored in sequential memory locations.

The readin operation continues until a perforated hole 7 is detected. This condition is inserted in the last character of the last 18-bit instruction. When this condition is detected, the reader supplies the hardware readin logic with a skip request. As a result, the hardware readin logic causes the last instruction to be loaded into the Memory Input register for execution. This instruction can halt machine operation (HALT) or can transfer program control to another part of the program (JMP). When using the readin feature with the MP15 Memory Parity option, the last instruction on the paper tape (which will be executed by the processor) will not be written into the next sequential memory location. That location, however, will be loaded with data that may contain wrong parity. Therefore, that location should be re-stored by the program before an attempt is made to read from it. Otherwise, a parity error will occur.

1.2.6.2 Program-Controlled Operation - The PC15 reader operates in the binary or alphanumeric mode depending on the select IOT instructions issued by the computer. On decoding a reader select alphanumeric (RSA) mode IOT (700104_8), the reader advances the tape one character, loads this character into the reader buffer, and sets the reader device flag. The reader then signals the computer that data are available by providing a reader interrupt to the API or PI, or by responding to an RSF IOT instruction. If the API facility is being used, program control is transferred to the reader service routine where the computer services the request, and an RCF (700102_8) or RRB (700112_8) instruction is issued. If the API facility is not being used, the computer issues an RSF instruction, and the reader returns a skip request whenever its flag is set. The skip request causes the next instruction (normally a JMP .-1 in wait loops) to be skipped so that the character can be transferred to the accumulator by issuing an RCF or RRB instruction. The RCF or RRB instruction transfers the reader buffer character to the I/O bus and loads it into the least significant bits (10 through 17 for 8-bit alphanumeric character) of the accumulator. The character is subsequently stored in a core memory location designated by the program. The read reader buffer (RRB) instruction also clears the reader flag for the next read operation.

For binary mode operation, the computer issues a reader select binary (RSB) mode instruction (octal 700144). On decoding this instruction, the reader clears its device flag, advances the tape three

characters, reads these characters from tape, and assembles them into an 18-bit word in the reader buffer. The reader also counts the number of characters with hole 8 punched read from tape and, when a count of three is reached, generates an interrupt request. The control functions for transfer of the 18-bit word to the accumulator is the same as that described for the alphanumeric mode.

1.3 PAPER-TAPE PUNCH

1.3.1 Characteristics and Capabilities

The PC15 paper-tape punch consists of a tape feed system, a mechanical punch assembly, a buffer register, and control logic for mode selection and activation of the tape feed and punch mechanism. Tape advance, mode selection, and transfer of information to the punch are controlled by IOT instructions. Tape is perforated at a rate of 50 characters/second. When the punch is selected by an IOT instruction, data from the PDP-15 accumulator (AC10-AC17) are transferred to the punch buffer. Then, without further inputs, a character is perforated on tape.

The punch contains a device flag that denotes punch status for transfers. This device flag interfaces with the PI facility and I/O skip chain. The status of the punch flag is displayed on an indicator panel at the top of Cabinet H963E (Bay 1R). An out-of-tape switch is located on the punch mechanism. This switch initiates action that stops punch operations when approximately one inch of unpunched tape remains.

Power for the punch operation is available whenever the PDP-15 power is on. The punch runs when selected by an IOT instruction or when the FEED switch is pressed.

Punch mechanical features include a magazine for unpunched tape and a container for tape chad. Both are accessible when the reader-punch drawer is extended from the cabinet.

1.3.2 Operating Modes

The PC15 Punch operates in the alphanumeric or binary mode as designated by IOT select instructions. One of these instructions is required for each character punched for mode change. In the alphanumeric mode, an 8-bit character (in ASCII or modified ASCII code) is punched for each accumulator transfer to the punch. For the binary mode, one 6-bit data character is perforated for each accumulator transfer. Hole 8 is always punched, and hole 7 is never punched. Three of these characters, however, form one computer word for readin operations.

1.3.3 Controls and Indicators

The PC15 Punch has a front panel FEED control. This control is used to advance the tape from the punch as required for leader or trailer. The punch also has one indicator (PUN FLG) directly associated with its operation. This indicator, located on an indicator panel at the top of Cabinet H963E (Bay 1R), indicates the status of the device flag and, shows that the punch is available for a punch operation when lit. The punch also shares the PI RQ and SKIP RQ indicators on this panel with other I/O devices.

1.3.4 Tape Formats

Tape formats are shown in Figures 1-1 and 1-2.

1.3.5 Instructions

The PDP-15 IOT instructions used for punching of paper tape under program control are listed below. Refer to Volume 1 of this handbook for IOT instruction format.

Mnemonic	Octal Code	Operation Performed
PSF	700201	Skip next instruction if punch flag is a 1.
PCF	700202	Clear punch flag and punch buffer.
PSA	700204	Select alphanumeric mode and punch one character. Set punch flag when punch is complete.
PSB	700244	Select binary mode and punch one 6-bit character. Set punch flag when punch is complete.

The punch responds to the IORS instruction (Volume 1, Paragraph 3.7.1) by supplying the status of its device flag and no-tape flag to the accumulator. The device flag interfaces with bit 02 of the accumulator, and the no-tape flag interfaces with bit 09.

1.3.6 Functional Description

The PC15 Punch operates in the alphanumeric or binary mode, depending on whether a PSA or PSB instruction is issued. When one of these instructions is decoded, information is loaded into the punch buffer from bits 10 through 17 of the accumulator and is punched onto tape. During the interval the punch operation is in progress, the punch flag is cleared to indicate the punch is busy. When the punch operation is complete, the punch flag is set to 1 to indicate it can accept another input character.

The operating sequence for punch operations normally begins with a PSF instruction to test the device flag. If the device flag is 1, a skip request is returned to the computer, and the computer issues a PCF instruction. This instruction clears the device flag and the punch buffer. The computer then issues a PSA or PSB instruction. On decoding a PSA instruction, the reader loads the accumulator input into its buffer, advances the tape, and punches one character. For the alphanumeric mode channel 8 is punched as a function of bit AC10. For the alphanumeric mode channel 7 is perforated as a function of bit AC11. After the character is punched, the reader sets its device flag, and the process is repeated. This operation, performed by the PCF and PSA instructions, can be combined by microprogramming the two instructions to form octal 700206.

The same principles are used for punching a binary character; however, a PSB instruction is used in place of the PSA instruction. On decoding a PSB, the punch perforates channel 8 and inhibits the punching of channel 7. The remaining six channels are punched as a function of AC12 through AC17, and represent one 6-bit character of a computer word.

1.4 PROGRAMMING CONSIDERATIONS

1.4.1 High-Speed Paper-Tape Reader

To use the reader at the transfer rate of 300 cps, a select IOT (RSA or RSB) must be issued within 1.67 ms after each flag. This action is required because a 40 ms reader stop delay is present. When this delay is activated, it overrides the select IOT input and subsequently stops the tape. Thus, if a new select IOT is not received within 1.67 ms of the setting of the flag, the reader operates start-stop and reads characters at 25 cps rate. No data is lost.

The RSA (octal 700104) and RCF (octal 700102) can be microprogrammed to form an octal 700106 instruction. This instruction reads the character, transfers the character to the accumulator, and advances the tape in one operation. An RSF (octal 700101) and RRB (700112) cannot be microprogrammed.

1.4.2 High-Speed Paper-Tape Punch

Channel 7 can be punched using only the alphanumeric mode. Therefore, when punching the last character of a tape for hardware readin operation, the last character must be punched in the alphanumeric mode.

The PCF instruction can be microprogrammed with a PSA or PSB instruction to form octal 700206 or 700246. This instruction clears the punch flag and buffer, selects the applicable mode, loads the

punch buffer, advances the tape, and perforates the character on tape. After completing the punching, the punch flag is set to denote the punch can accept another character. Microprogramming the PCF and PSF instructions is not allowed.

1.5 PROGRAMMING EXAMPLES

1.5.1 Paper-Tape Reader/Punch Handlers

All PDP-15 Systems are supplied with standard I/O device handler subroutines for the paper-tape reader/punch hardware. For PDP-15/10 Systems with 4K core, the COMPACT software includes paper-tape handler routines such as PTLIST and PTDUP. The Basic I/O Monitor, supplied with PDP-15/10E Systems with 8K core or greater, include standard I/O device handlers for the high-speed paper-tape reader and punch. These standard device handlers operate in systems with or without API and are upward compatible with all other monitors on the PDP-15/20 Software System. Complete instructions on use of standard paper-tape reader and punch handlers and their modification for special applications are provided in the PDP-15/10 Software System Manual, DEC-15-GR1A-D.

1.5.2 Paper-Tape Reader Programming Example

The following subroutine illustrates the use of programmed IOT instructions to read a group of binary words from paper tape. Twenty-five 18-bit words are read and stored in a table starting at ADDRESS.

NOTE

This example is for instructional purposes only and is not to be considered a complete, fully tested software system segment.

```
SUBRTE  0
        LAW   -31       /25 DECIMAL WORDS
        DAC   WDCNT
        LAC   (ADRESS
        PAX
READLP  IORS
        AND   (1000     /IS THE PAPER TAPE READER EMPTY?
        SZA             /YES IF NON-ZERO.
        JMP*  SUBRTE    /EXIT....ITS EMPTY
        RSB             /NO.  START READING A WORD.
        RSF
        JMP   .-1       /WAIT FOR IT.
        RRB             /GET IT FROM HARDWARE BUFFER.
        DAC   0,X
        AXR   1         /POINT TO NEXT LOC AT ADDR.
        ISZ   WDCNT     /HAVE 25 WORDS BEEN READ?
        JMP   READLP    /NO...CONTINUE LOOPING.
        JMP*  SUBRTE    /YES. EXIT.
```

1.5.3 Paper-Tape Punch Programming Example

The following subroutines illustrate some paper-tape punch programming considerations. Their purpose is to unpack successive 6-bit ASCII characters from a table, convert them to 7-bit ASCII, and punch them on paper tape. The starting address of the table is placed in a location named ADDRESS. The number of words in the table is placed in WORDCNT. After these parameters have been deposited, the subroutines are entered by a JMS to PNCHOUT.

NOTE

This example is for instructional purposes only and is not to be considered a complete, fully tested software system segment.

```
PNCHOUT   0
          LAC   WORDCNT        /THIS INITIALIZATION
          TCA                  /ROUTINE STORES 2'S
          DAC   WORDCNT        /COMPLEMENT WORDCNT
          CLX                  /AND CLEARS XR.
NXTWORD   LAW   -3             /SET UP A COUNTER FOR
          DAC   COUNT          /3 CHARACTERS.
          LAC   ADDRESS,X      /USE XR TO GET EACH WORD.
          RAL                  /AC HOLDS 3 6-BIT ASCII
                               /CHARS. ROTATE INTO LINK.
NXTCHAR   RTL                  /ROTATE WORD 6 PLACES
          RTL                  /THRU LINK. THE NEXT
          RTL                  /6-BIT CHAR. IS IN AC12-17.
          DAC   SAVEAC         /SAVE REMAINING CHARS.
          AND   (77)           /THIS ROUTINE CONVERTS
          TAD   (40)           /THE 6-BIT ASCII IN AC12-17
          AND   (77)           /TO 7-BIT ASCII IN
          TAD   (40)           /AC11-17.
          JMS   PPCHAR         /READY TO PUNCH CHAR.
          LAC   SAVEAC         /RESTORE SHIFTED AC.
          ISZ   COUNT          /LAST CHARACTER?
          JMP   NXTCHAR        /NO. DO NEXT CHARACTER.
          AXR   1              /POINT TO NEXT WORD.
          ISZ   WORDCNT        /LAST WORD?
          JMP*  PPASCII        /NO. DO NEXT WORD.
          JMP*  PNCHOUT        /YES. RETURN TO PROGRAM.
PPCHAR    0
          DAC   STORE          /SAVE CHAR. FOR 'NO TAPE'
                               /TEST.
          IORS                 /LOAD PUNCH STATUS INTO AC.
          AND   (400)          /TEST NO PUNCH TAPE BIT.
          SZA                  /SKIP IF TAPE OK.
          JMP   EOT            /GO TO END OF TAPE RTE.
```

```
        LAC     STORE           /LOAD AC WITH CHARACTERS
        PSA                     /SELECT ALPHA MODE & PUNCH
        PSF                     /WAIT FOR FLAG.
        JMP     .-1
        PCF
        JMP*    PPCHAR          /RETURN TO SUBPROGRAM.
```

1.5.4 Programming With API or PI

The standard device handlers for the high-speed paper-tape reader and punch include complete interrupt subroutines for both API and PI service. Details on how the Program Interrupt Control (PIC) skip chain and the Automatic Priority Interrupt (API) channels are set up and provided in Part III of the PDP-15/10 Software System Manual. The following example of a hypothetical interrupt service subroutine is provided for general understanding of interrupt servicing.

NOTE

This example is not a complete, fully-tested interrupt service handler.

1.5.4.1 Program Interrupt Example

```
                RSB                     /ISSUE READER SELECT
                 .                      /BINARY IOT WITH PI ENABLE.
                 .                      /REST OF USER PROGRAM.

                .LOC    0
PI              0                       /SAVE PC, LINK, EXTEND MODE
                                        /& MEM. PROT. BITS AT LOC0.
                JMP     SKPCHN          /GO TO SKIP CHAIN.
                 .
                 .
SKPCHN          SPFAL                   /POWER FAIL FLAG TEST.
                SKP                     /GO TO NEXT TEST.
                JMP*    INT6            /GO TO POWER FAIL SUBRTE.
                RSF                     /PAPER-TAPE READER DONE?
                SKP                     /GO TO NEXT TEST.
                JMP*    INT2            /GO TO PTR INTERRUPT.
                PSF                     /PAPER-TAPE PUNCH DONE?
                SKP                     /GO TO NEXT TEST.
                JMP*    INT3            /GO TO PTP INTERRUPT.
                 .                      /OTHER TESTS
                 .
                 .

/INT6, INT2, AND INT3 ARE PART OF A TABLE
/OF INTERRUPT SERVICE ROUTINE STARTING ADDRESSES.
/AN EXAMPLE OF INT2 FOLLOWS:
```

```
INT2        PTRPIC                  /15-BIT ADDRESS OF PAPER
              .                     /TAPE READER SERVICE
              .                     /ROUTINE.
              .
              .                     /OTHER I/O SERVICE ROUTINE
              .                     /POINTERS
              .
PTRPIC      DAC     PTRAC           /SAVE AC.
            LAC*    (0              /SAVE PC, LINK, BANK MODE
              .                     /AND USER MODE IN PTROUT.
              .
              .                     /REST OF INTERRUPT HANDLED.
              .
```

1.5.4.2 API Example

```
            RSB                     /SELECT READER IN
              .                     /BINARY MODE
              .
              .                     /REST OF INTERRUPT
                                    /HANDLED.
            .LOC    50
            JMS     PTRINT          /PAPER TAPE READER
              .                     /API ENTRY LOCATION
              .
              .
PTRINT      0
            DAC     PTRAC           /API ENTRY. SAVE AC.
            LAC     PTRINT          /SAVE PC, LINK, BANK
                                    /MODE & USER MODE BITS.
            DAC     PTROUT          /
```

Chapter 2
The DECdisk System

2.1 INTRODUCTION

The DECdisk system is a computer peripheral that stores digital data on fixed-head rotating disks in serial format. The data can be randomly accessed at selectable speeds and, when necessary, protected from overwriting.

2.1.1 System Description

The DECdisk system comprises an RF15 DECdisk Controller and from one to eight RS09 Disk Drives. The controller connects to the computer I/O Bus and communicates with the central processor for control and status information. For data information, the controller communicates with memory through the data channel. Each disk drive connects to the controller through a parallel disk bus. Both control and data information pass through the parallel disk bus. Figure 2-1 illustrates the DECdisk System.

2.1.2 Storage of Digital Data on Fixed-Head Rotating Disks

Each RS09 Disk Drive consists of a rotating disk, a hysteresis synchronous motor, a matrix of 128 fixed read/write heads, and the electronics required to drive the heads.

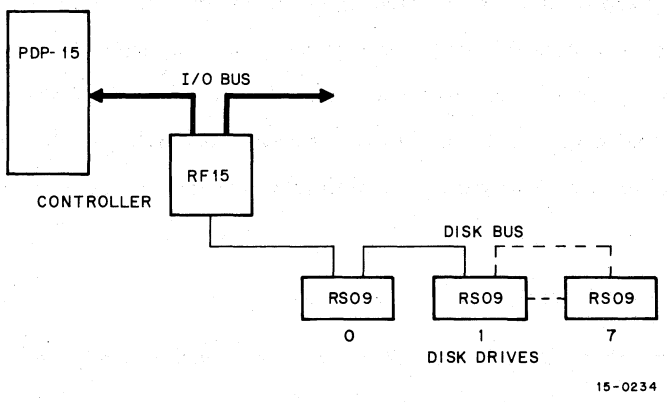


Figure 2-1 DECdisk System Configuration

The 128 magnetic read/write heads ride on the surface of the rotating disk, which is nickel-cobalt plated. Each read/write head covers a separate track on the nickel-cobalt surface; thus, disk action is similar to the operation of many circular tapes running simultaneously in continuous loops.

Each track on the disk can store 2048 eighteen-bit data words. As a track fills, the system automatically moves to the next track. The disk rotates at 1800 rpm (60 Hz power) and can, therefore, transfer a word every 16 μs. The storage capacity of each disk is 262,144 words (2048 words x 128 heads). Total system capacity is 2,097,152 words (8 disk drives x 262,144 words).

2.1.3 Storage of Data in a Serial Format

The DECdisk System stores the data on each disk in a serial format. The serial format causes the bits of each word to be recorded one at a time along a single track, rather than all at once across eighteen tracks. Therefore, only 1 of a possible 128 data heads is actively reading or writing data at a single time.

2.1.4 Random Accessing of Data

The DECdisk is a random-access storage system. Each disk is logically segmented into 2048 slices or words, and each slice is preassigned a number or address from 1 to 3777_8. The controller, in response to the computer, can select at random any track of a disk and any address along that track to read or write a word.

2.1.5 Data Accessing at Selectable Speeds

There are three speeds (switch-selected by the operator) at which data can be transferred between the disk surface and the computer. The highest speed transfers a data word with each successive address, covering a track in one revolution. The medium speed transfers every second word of a track in the first revolution, and then transfers the alternate words on the same track during the second revolution. The slowest speed takes four revolutions to cover a complete track. Once the operator has selected the desired speed, the controller hardware controls proper interleaving of the words. However, the data should be read back at the same speed at which it was written to avoid scrambling the data.

2.1.6 Data Protection from Over-Writing

Sixteen switches are available on each RS09 Drive to protect disk-stored data. Each switch can inhibit the computer from overwriting on eight separate tracks.

2.2 DECDISK OPERATION

Information flow within the DECdisk System is determined by the recording format on the disk surface and the internal architecture of the controller. The following paragraphs describe the operation of the disk recording format and the system architecture.

2.2.1 Disk Surface Recording Format

As previously described, 128 read/write heads covering 128 concentric tracks ride on each disk surface. The circumference of each disk is logically divided into 2048 data segments or addresses, and in each segment of any track a complete 18-bit computer word can be stored. A 2049th segment called a gap is provided to give the heads time to switch tracks. This segment has no address and stores no data or timing tracks. It is used as a marker to notify the controller each time a revolution has been completed.

Each data segment must store, in addition to its data word, six control bits; and each disk, in addition to its data tracks, must contain six control tracks. The control bits are recorded with the data bits; the control tracks are prerecorded on the disk surface at the factory. Figure 2-2 illustrates the location of these bits and control tracks.

Data are recorded serially on each track in 24 bit words; 18 data bits, one parity bit, four guard bits, and one data control bit. Each 24-bit word unit is identified by an address that is prerecorded on a special track before the disk is connected to the computer in the plant. This address is recorded serially on the B track (see Figure 2-2) exactly one word before the word with which it is associated. The controller can then assemble and identify the address before the heads reach the word itself. Each address is 13 bits long; 11 bits supply addressing data, 1 bit is a control bit, and 1 bit is a parity bit.

There are five additional prerecorded tracks on the disk surface. The A track is a prerecorded track with pulses 660 ns apart that are used to strobe data into or out of the data tracks. The C track is a track used to delimit each word unit. The controller relies on the C track to signal when a word has been assembled or written. The controller can then notify the computer to accept the word read or to supply another word to be written. Each of the three prerecorded tracks described - the A, B, and C tracks - are copied on three spare tracks that are used if one of the original tracks is accidentally erased in the field. If the spare tracks are damaged, all the timing tracks can be rewritten in the field with a special timing track writer.

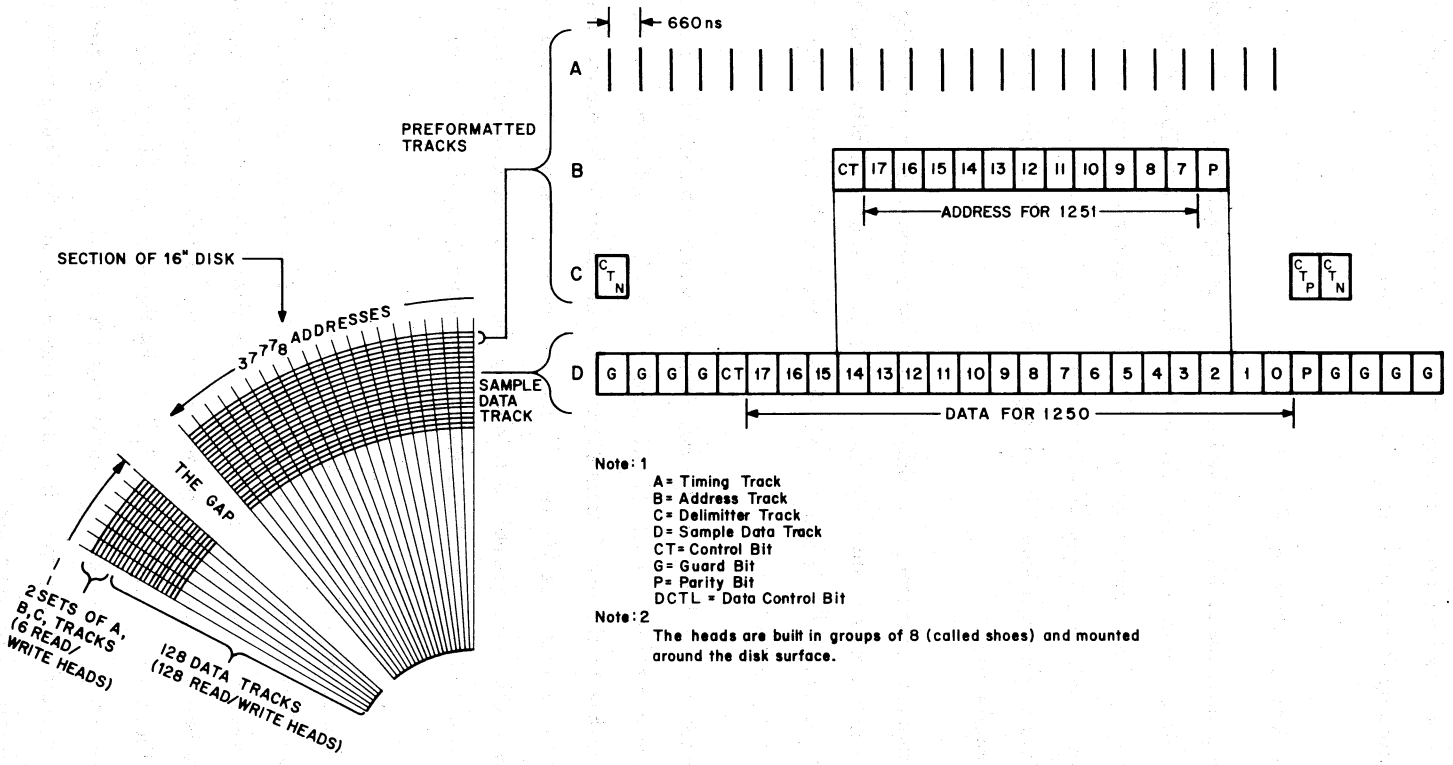


Figure 2-2 Disk Surface Recording Format

2.2.2 DECdisk Architecture

In this manual, the DECdisk System architecture is presented in three parts; the Control section, the Data Transfer section, and the Maintenance section (shown in Figures 2-3, 2-4, and 2-5 and 2-6, respectively). Through the Control section, the Software Operating System initializes the controller by selecting the disk drive (RS09) to be used, the track address within that drive (Data Track Matrix) to be used, and the first address within the track to be used. One of three functions is then selected: READ the disk; WRITE on the disk; or WRITE CHECK what has already been read or written. The Data Transfer section assembles the word off the selected track for a READ operation, or writes the word bit by bit onto the track during a WRITE operation. This section also notifies the computer when it has assembled a word or needs another word to write, and the data is transferred through the three-cycle data channel. When the last word has been transferred, the computer issues an overflow pulse to the controller. An interrupt then occurs, and transfers are stopped. The Maintenance section simulates either the disk surface head signals or RS09 output signals and is used exclusively for testing the DECdisk System.

2.2.2.1 The Control Section - The block diagram of the Control section in Figure 2-3 shows 11 relatively independent sections. Some of these sections contain registers, and the bits of these registers are numbered according to the position they occupy when they are read from or into the accumulator of the Central Processor.

Three of these registers - the Disk Number, the Track Address, and the Word Address - are set by the software system to select the disk (one of a possible eight), the track within that disk (the read/write head matrix), and the starting address within the track. Each time a word is transferred, the word address is automatically incremented by one to prepare for the next word. When the Word Address Register overflows, the track address is automatically incremented; and when all tracks have been exhausted, the Disk Number Register is incremented. These registers continually step from word to word, track to track, and disk to disk until the system has been covered.

NOTE

Incrementing occurs during a valid operation only.

After the system has been covered, the computer is notified that it has run out of disks. The dead space (gap) shown in Figure 2-2 is used to give the controller time to switch tracks when it needs to do so.

The Word Address Register is constantly being compared to the contents of the Segment Register, which in turn is sampling the "B" or address track. When the "C" or delimiter track indicates that a valid

address in the Segment Register, the word address is compared with the assembled address; and if the two match, an ADDRESS OK signal is passed to the data transfer logic. This signal informs the data transfer logic that the data it wants to read is presently passing over the read head of its selected channel, or that the space in which the data transfer logic wants to write is about to come under the read/write heads.

The interlace logic is used by the operator to reduce the transfer rate of the disk to either a medium or a low speed. The medium speed cuts the rate in half by adjusting the final address of the Disk Segment Register so that only every second address is used in the first revolution of the disk, and the alternate addresses are picked up from the same track on the subsequent revolution. The low speed cuts the transfer rate by four. Each address is then adjusted to require four revolutions of the disk before a complete track is filled. Bits X4 and X2 indicate low and medium speed, respectively, and are set if these speeds are selected by the operator. The flag BZ sets whenever a valid operation is under way, and WB sets when a data "1" is to be written. All of these bits can be read into the accumulator under program control.

The ADS Register receives each valid current segment address from the Segment Register. The current segment address is then available to the accumulator in the ADS Register under program control. Note that the ADS Register receives the current address, and not the adjusted address for low or medium speed transfers.

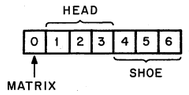
There are three bits in the Function Register, which is double buffered. Bits 15 and 16 specify the function that is to be performed by the controller. The function is loaded into the first buffer, and an execute IOT (DSCN) is issued to load it into the second buffer for execution. At the end of an operation, or if an error occurs, the second buffer is cleared and execution stops. The operation can then be continued by issuing a DSCN IOT execute. Table 2-1 shows the bit configuration needed to select each function. Bit 17, also contained in the Function Register, enables the program interrupt and API logic of the control.

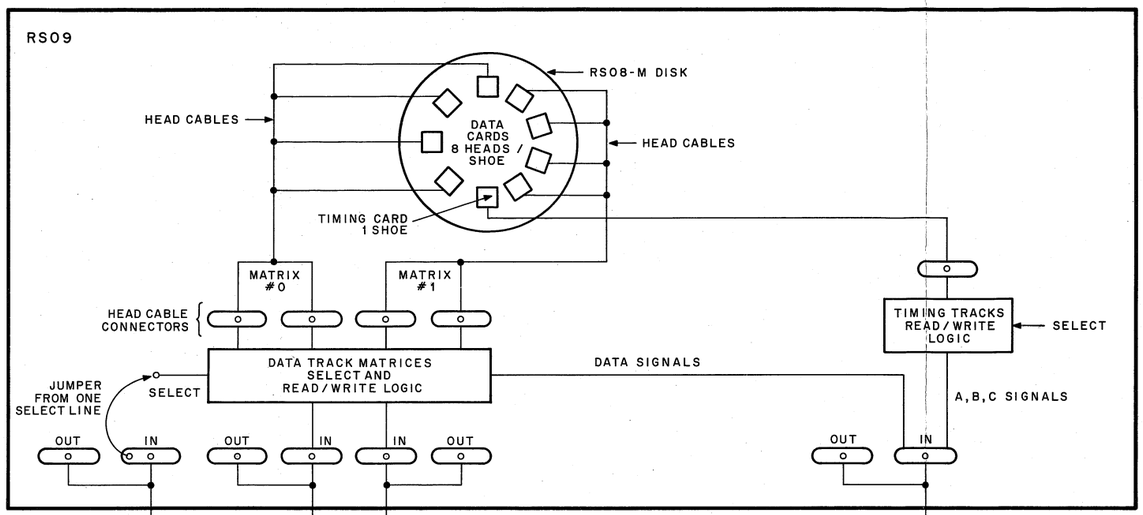
Table 2-1
The Function Register Bit Configuration

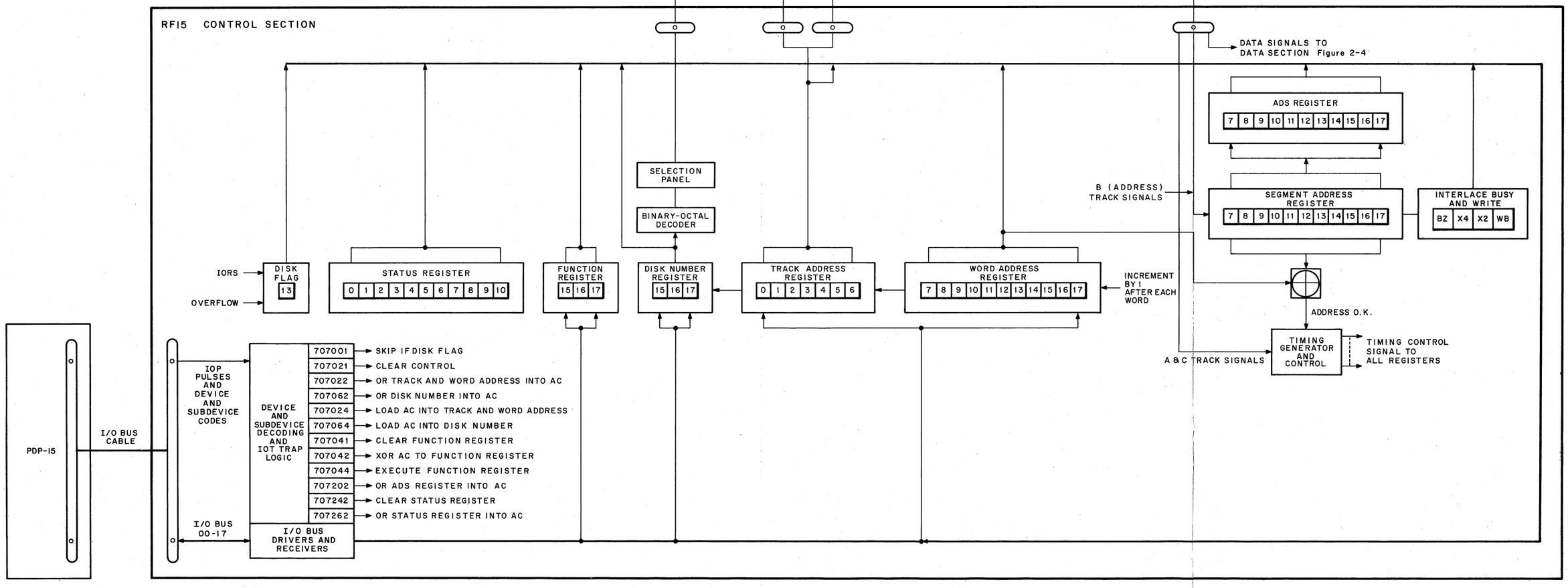
Function	Bit 15	Bit 16
No effect	0	0
Read	0	1
Write	1	0
Write Check	1	1

NOTES:

1. The READ/WRITE data heads are mounted on shoes in groups of 8 and each shoe is mounted on a card. The cards are mounted around the underside of the disk so that each head covers a different track. The timing card has 1 shoe.
2. The cards are cabled from the RS09 READ/WRITE logic and selection matrix, which in turn cable to the controller.
3. Each RS09 has both input and output cable slots. The signals are cabled in parallel from drive to drive to a maximum of 8.
4. The track address register selects the head according to the folowing bit configuration:







09-0413

Figure 2-3 DECdisk Control Section

The timing generator and control logic receive the A and C track signals and generate all of the system timing and control pulses necessary to carry out the various macro operations (such as shifting the Segment Register and incrementing the Word Track and Disk Address Registers).

The 10-bit Status Register reflects the state of the system after it has performed its specified operation. Any timing or parity errors that have occurred during the operation are indicated here. Table 2-2 summarizes the function of each bit.

2.2.2.2 The Data Transfer Section - The data transfer section, shown in Figure 2-4, has 4 subunits; two 18-bit registers and two controls. During a READ operation, a word is assembled into the Shift Register. If the word has been assembled from the selected address (ADDRESS OK), and the C track indicates that a valid word has been assembled; the contents of the Shift Register are then jammed into the Buffer Register. The computer is notified that a word is ready for transfer, and a multi-cycle data break occurs. At the same time, the Shift Register is assembling the next word. The word count (WC) and current address (CA) for the DECdisk are in location 36_8 and 37_8, respectively.

During the WRITE operation, the computer transfers the word to be written into the Buffer Register where it waits for the ADDRESS OK signal. When this signal arrives, the word is immediately transferred to the Shift Register and is serially shifted from there onto the selected track.

During a WRITE CHECK operation, which is designed to allow the programmer to compare data in memory with corresponding data on the disk, the memory word is fed into the Buffer Register and into the Shift Register, where it is compared bit by bit with the corresponding word being read directly from the disk. If a discrepancy or a parity error exists, the DISK flag is posted.

The instruction set, listed in Table 2-3, allows the computer to clear, load, or read from each of seven registers in the control section. The following points should be noted:

a. The DISK flag is posted under two conditions;

   (1) at the end of the operation, and

   (2) if one of the six error conditions occur.

   The DISK flag causes either a PI or an API interrupt, if these interrupts are enabled in both the controller and the computer.

b. Whenever the DISK flag is posted, the second buffer of the Function Register is cleared, and the operation stops. The first buffer does not clear; and the operation can either be continued by issuing the execute IOT, or altered by changing its code and then issuing the execute instruction.

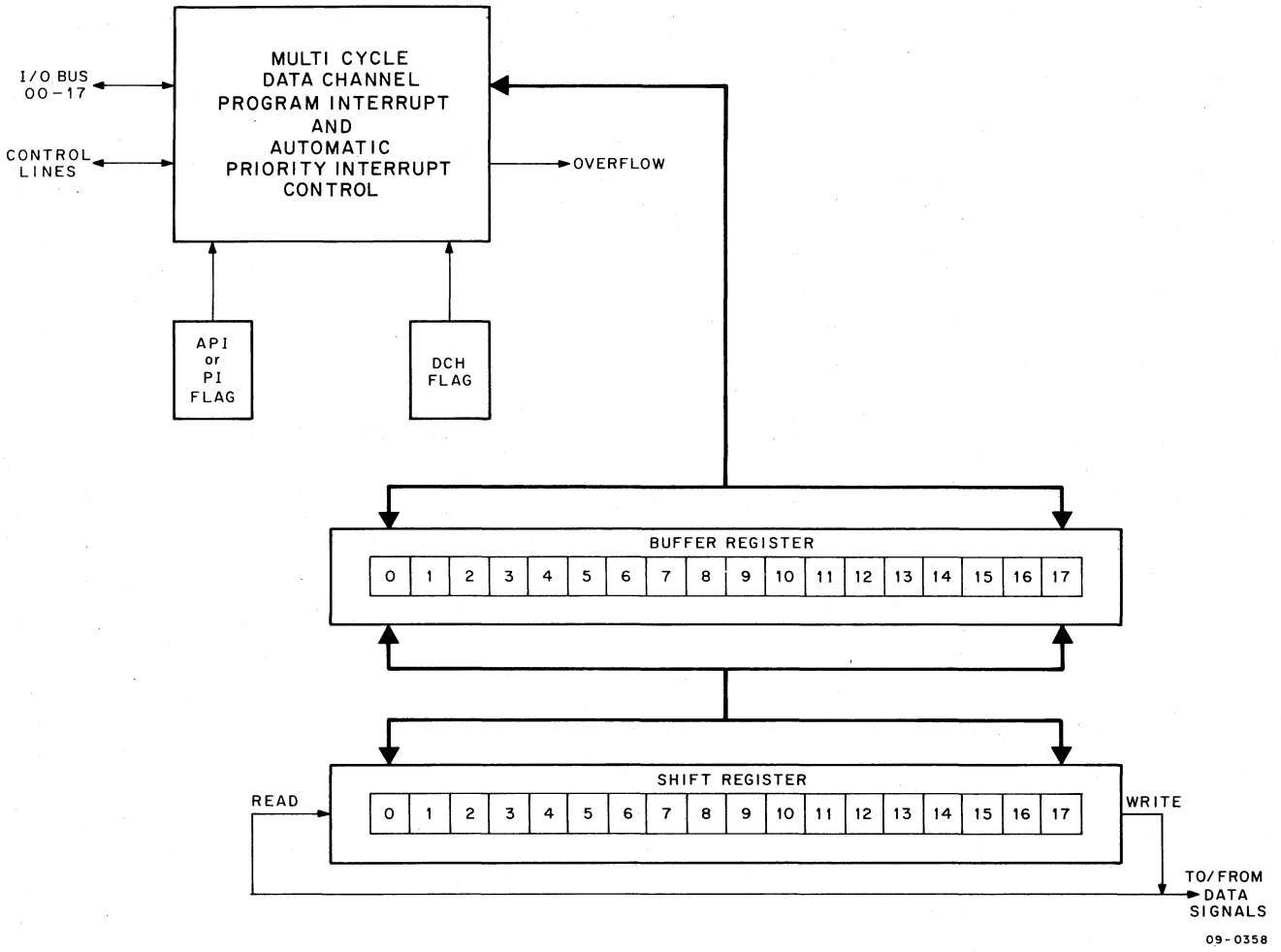


Figure 2-4 DECdisk Data Transfer Section

c. The ADS Register reflects the current position of the disk and not the adjusted address. A program can read its contents and calculate the nearest possible address to which it could transfer its first word (taking into account the speed setting), set the address into the Address Register and, thereby, reduce the initial latency time. (The ADS Register can be one address late.)

d. The disks are not synchronized. When the control transfers from disk to disk, the control itself has no way of knowing the next disk location in its revolution. The ADS Register locates the next disk.

e. During an operation, the Disk, Track, and Word Address registers automatically increment as the system rotates from word to word, track to track, and disk to disk. At all other times, these registers remain constant.

Table 2-2
Status Register Bit Functions

Bit	Flag Name	Function
0	ERR	This ERRor flag is the logical OR of the error conditions of bits 1 to 7. When this bit is set, it causes an interrupt and conditions the skip IOT. It also inhibits the current operation until a continue IOT is issued.
1	HDW	The disk HarDWare Error is set if the control detects missing bits from the A, B, or C track. A set HDW causes the control to freeze for further evaluation. (During a "freeze" condition, writing is stopped and the A timing pulses are inhibited.) A freeze is disabled with an I/O RESET, a CAF, or the DEC-disk clear IOT.
2	APE*	The Address Parity Error flag is set if a parity error occurs when the address is being assembled, provided that the control has been programmed to READ/WRITE or WRITE CHECK. This flag does not set if the disk is idling. APE also freezes the control.
3	MXF	A Missed X (Trans)Fer flag is set if the disk requested a data transfer from the computer and did not get it for 2-3 revolutions. A 130 ms timer triggers to post the MXF flag. Either a data channel failure or a data channel overload initiates this flag. When analyzing an MXF error, the following points should be considered: a. The computer increments its current address in the cycle before it transfers its data. b. The controller increments its disk or track address when it requests a transfer during a read operation, but only after a transfer is acknowledged during a WRITE or WRITE/CHECK operation.

*Note that the hardware is designed to allow only the first of these three errors to set during an operation.

Table 2-2 (Cont)
Status Register Bit Functions

Bit	Flag Name	Function
4	WCE	When the Write Check Error flag is set, the controller has discovered during a WRITE CHECK that the word from memory differs from its corresponding word on the disk. The error flag is raised and all further checking is stopped. The word being checked is in disk location WA-1 (Word Address minus 1), and its corresponding word is in memory address CA-1 (Current Address minus 1).
5	DPE*	The Data Parity Error status bit is set whenever the data parity bit does not agree with the computed parity of the data word just read. The control transfers the data word containing the parity error and raises the error flag. No further transfers occur until the program intervenes. The WA-1 contains the disk address of the word in error. The CA contains the memory address of the word in error.
6	WLO*	The Write LockOut error bit is set when an attempt is made to write into a protected region on the disk. READ or WRITE CHECKING a protected area is permitted.
7	NED	If a disk which does not exist is called for under program control or sequenced into during data transfers, the Non Existence Data flag is raised to signal the error.
8	DCH	The Data CHannel Timing Errors status bit is set whenever the processor has not completed a DCH transfer before the disk control is ready to transfer data. No error flag is raised. This status bit is intended as a warning that the DCH channel is overburdened.
9	PGE	The ProGramming Error status bit is set whenever the program issues an illogical command to the disk. Furthermore, if the command directly conflicts with the operation of the control, the command is ignored. No error flag is raised. This status bit is provided as a warning to the programmer.
10	XFC	When the job requested via the program (either READ, WRITE, or WRCHK) is finished, the (X) TransFer Complete flag indicated by this bit interrupts the processor and conditions the SKIP IOT.

*Note that the hardware is designed to allow only the first of these three errors to set during an operation.

Table 2-3
The DECdisk Instruction Set

Code	Mnemonic	Description
707001	DSSF	Skip if Disk Flag. The Disk flag is raised for either an error condition (ERR) or when transfer is complete (XFC). This flag is indicated on bit 13 of the Input/Output Read Status (IORS) facility. If the Program Interrupt (PIE) and/or Automatic Priority (API) is enabled, the DSSF flag causes the program to be interrupted.
707021	DSCC	Clear the Disk Control and disable the "freeze" status of the control. This IOT is the only command honored by the control when a "freeze" is caused by either a timing track hardware or an Address Parity Error and forces the control to abort the operation in progress. It effectively Power Clears the DISK CONTROL.
707022	DRAL	OR the contents of the Address Pointer 0 (AP0) into the AC. Bits 0 through 6 contain the track address and bits 7 through 17 contain the word address of the next word to be transferred.
707062	DRAH	OR the contents of the Disk Number (AP1) into the AC. Bits 15, 16, and 17 contain the Disk Number. Bit 14 is read back if a data transfer has exceeded the capacity of the Disk Control (causes a NED error status).
707024	DLAL	Load the contents of the AC into the AP0.
707064	DLAH	Load the contents of the AC (15, 16, 17) into the Disk Number (AP1).
707041	DSCF*	Clear the Function Register, Interrupt Mode.
707042	DSFX*	XOR the contents of AC bits 15-17 into the Function Register (FR). The use of each bit is the same as described for bits 15-17 of the Status Register.
707044	DSCN*	Execute the condition held in the FR. Since the AP contains the next available word (because it is incremental), this IOT can be used to continue after having changed the Word Count (WC) and Current Address (CA) held in core memory, or it can be microcoded with the Clear (DSCF) and XOR (DSFX) instructions to execute a new function at different address.
707202	DLOK	OR the contents of the 11-bit Disk Segment Address (ADS) into the AC. The ADS Register contains the real-time segment address, which is useful for minimizing access times. The address read always indicates the physical position of the disk (that is, one address of 2048 for one revolution (360°) of the disk, independent of the transfer rate being used).

*These instructions may be microcoded in any combination.

Table 2-3 (Cont)
The DECdisk Instruction Set

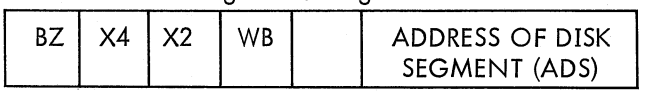
Code	Mnemonic	Description
707202 (Cont)		Register Configuration  When reading the ADS Register, the most significant four bits contain the status condition explained below. AC Bit / Name / Function 0 BZ — Busy. The disk has been commanded to transfer data and it is not finished. When reading the ADS Register, this is an indication that if the Address Pointer is used by the programmer to determine the Track Address (TA), the Track Address may not be valid if the ADS Register contains 3777 (since the TA may be changing at this time). 1 X4 — The control is set to transfer every fourth word. The effective transfer rate is, therefore, 64 μs per word. 2 X2 — The control is set to transfer every other word. The effective transfer rate is, therefore, 32 μs per word. If neither X4 nor X2 is set, the control is operating at its highest rate or 16 μs per word. AC Bit / Name / Function 3 WB — Write Bit. This bit is used primarily for maintenance purposes. It is the intermediate storage location for the data being transferred to the disk during WRITE.
707242	DSCD	Clear the Status Register and Disk Flag.
707262	DSRS	OR the contents of the Disk Status Register with the Accumulator (AC). Status at the point of interrupt is as follows: AC 0 — Error (ERR) 1 — Disk Hardware Error (HDW) 2 — Address Parity Error (APE)

Table 2-3 (Cont)
The DECdisk Instruction Set

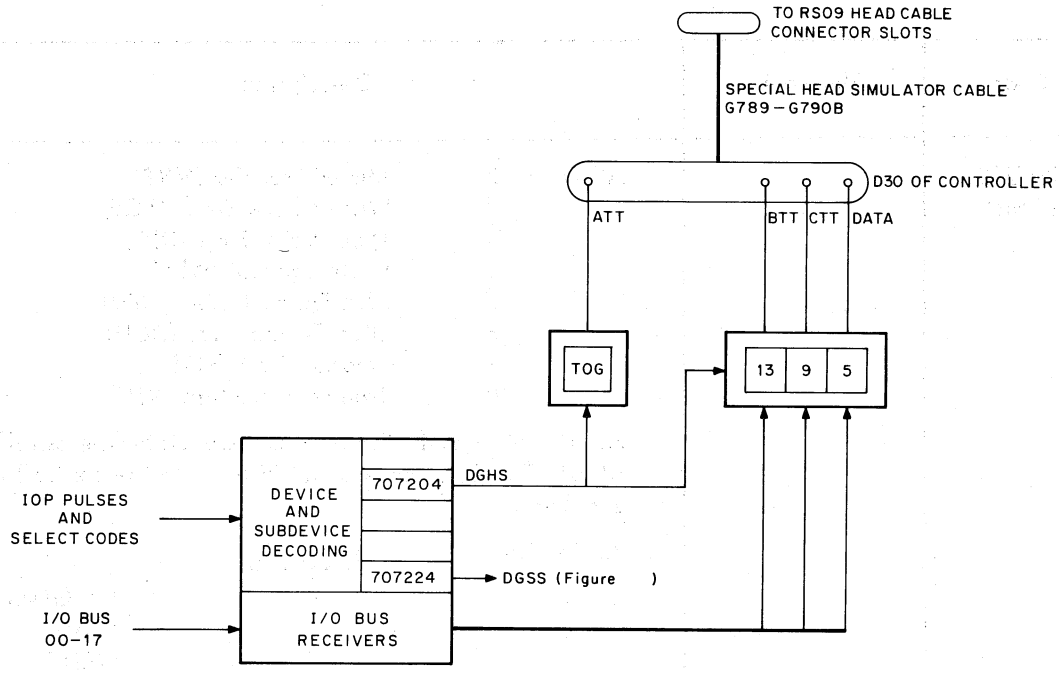
Code	Mnemonic	Description
707262 (Cont)		AC 3 Missed Transfer (MXF) 4 Write Check Error (WCE) 5 Data Parity Error (DPE) 6 Write Lockout (WLO) 7 Non Existent Disk (NED) 8 DCH Timing Error (DCH) 9 Program Error (PGE) 10 Transfer Complete (XFC) AC 15, 16, and 17 Function Register states are as follows: (If Bit 17 is a 1, the API and PI logic in the controller is enabled.) Bit 15 (F0) / Bit 16 (F1) 0 / 0 — No Effect 0 / 1 — READ 1 / 0 — WRITE 1 / 1 — WRCHK

2.2.2.3 Maintenance Section - The Maintenance section provides a means to test each unit of the DECdisk System without running the other units. Signals that usually come from the read/write heads of the disk surface can be simulated by the controller under IOT control with the logic shown in Figure 2-5. Similarly, signals from the RS09 output cables can be simulated by the controller with the logic shown in Figure 2-6. In this way, the controller can be tested without the disk drive, and the RS09 electronics can be tested without the disk surface.

The Buffer Register, which is normally available to the data channel alone, can be accessed from the Central processor under the control of maintenance IOTs.

A design feature of the control is that signals transmitted over cables between the controller and the RS09 disks perform active functions while they are themselves active. Therefore, if a wire in the cable is broken, a function is disabled rather than uncontrollably activated.

Table 2-4 lists the maintenance IOTs, and Figures 2-5 and 2-6 show simplified versions of some of the maintenance logic for the simulator section.



NOTES:
1. TOG is complemented when DGHS is released.
2. Bits 13, 9, and 5 are set from corresponding AC bits.
3. The letters ABCD indicate which track is simulated. D is for all data tracks.

Figure 2-5 Simulating the Disk Surface with the Maintenance Logic

Table 2-4
Maintenance IOTs

Code	Mnemonic	Description
707204	DGHS	Generate Simulated Head signals. This maintenance IOT causes the control to generate analog signals that simulate the disk head signals, as received directly from the head. The AC is used to determine the sequence of pulses to be generated, and the bit rate is controlled by the diagnostic program. Each IOT, in effect, is treated as though it were one cell space on the disk. The function of the AC bits is shown in Figure 2-5.

Table 2-4 (Cont)
Maintenance IOTs

Code	Mnemonic	Description
707204	DGHS	The bits are arranged as shown to provide for data packing, since only the bits that appear in the AC in bit cell 1 position are used when the IOT is generated. An RAR can then be used to position the data for the next Simulated Head Signal. When either of the maintenance IOTs (707204 or 707224) is used, a Maintenance Control flip-flop is set that inhibits the effect of control delay timeouts, which are a result of the lower data rates encountered under program simulation. If subdevice Bit 0 (MB12) is used when issuing the above IOTs, the Maintenance Control flip-flop is cleared.
707224	DGSS	Generate Simulated Disk signals. This IOT causes the control to generate Simulated Disk Interface signals within the control. No disk is necessary. The AC is used to determine the sequence of pulses to be generated and the bit rate is controlled by the diagnostic program. Each IOT, in effect, is treated as though it were one cell space on the disk. The function of the AC bits is shown in Figure 2-8. The bits are arranged as shown to provide for data packing, since only the bits which appear in the AC in bit cell 1 position are used when the IOT is generated. An RAR can be used to position the data for the next Simulated Head Signal.
707002	DRBR	OR the contents of the Buffer Register with the AC. This is a function normally performed by the data channel.
707004	DLBR	Load the contents of the AC into the Buffer Register. This is a function normally performed by the data channel.

2.3 THE OPERATOR'S CONTROLS

There are three groups of operator controls on the disk system. These include a three-position switch to select the transfer rate, a jumper panel to assign the address of each disk, and a series of write lockout switches to protect regions from being written onto each disk. The first two controls are part of the controller itself, and the write lockout switches are available on each disk drive.

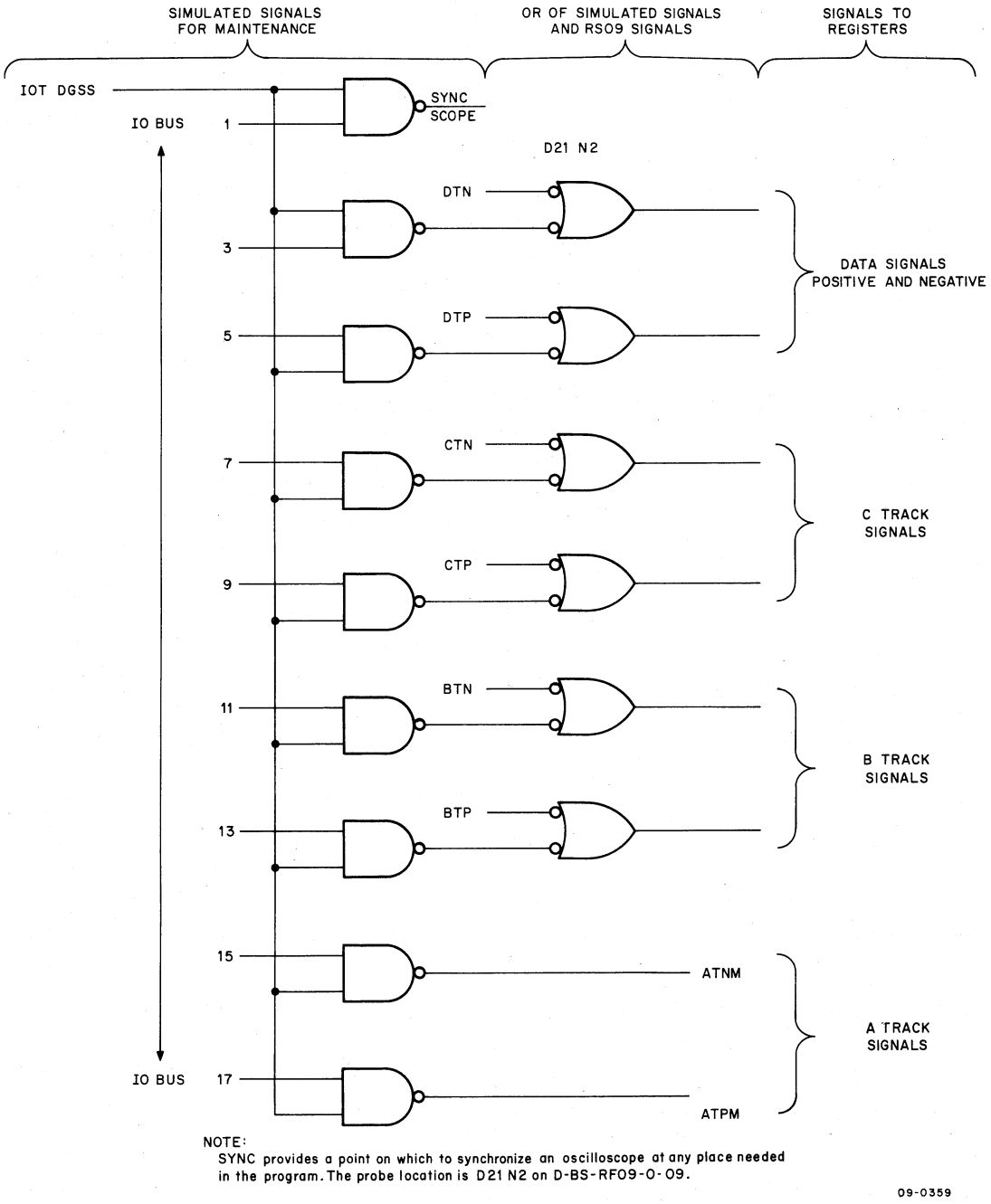


Figure 2-6 Simulating the RS09 with the Maintenance Logic

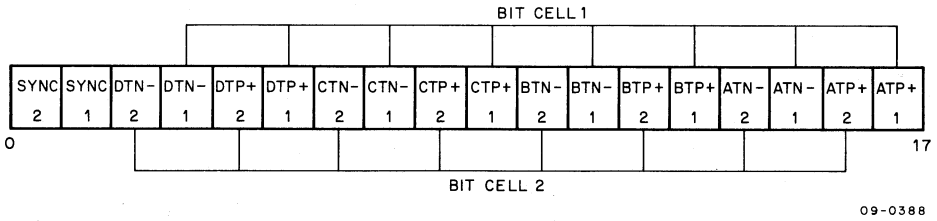


Figure 2-7 AC Bit Usage for IOT DGSS

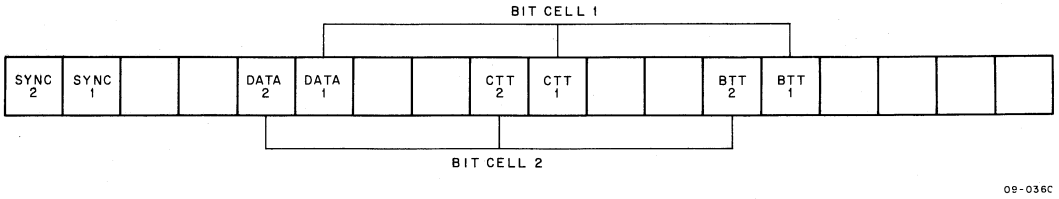


Figure 2-8 AC Bit Usage for IOT DGHS

2.3.1 Transfer Rate Selection

The operator can select a high, medium, or low transfer rate by positioning the rate selection switch at HI, MED, or LOW.

At the high speed, data are transferred to or from each word on a disk channel every 16 μs.

At the medium speed, the rate is halved to every 32 μs, not by slowing down the disk, but by requiring the disk to rotate twice in order to fill a channel completely. During the first rotation, every second address is read from or written into; in the second rotation the remaining addresses are used. All this is done automatically without extra coding. However, the programmer must ensure that the disk is read at the same speed at which it was written, or the data become unintelligible.

At the low speed, every fourth address is used on the first revolution, and the remaining addresses are picked up on the successive three turns. The transfer rate is one word every (4 x 16) 64 μs. The programmer is not required to do any extra coding, as the hardware completes the operation. The programmer must, however, ensure that the data are read and written at the same speed.

2.3.2 Disk Address Selection Jacks

The jacks shown in Figure 2-9, which are part of the controller, are used to assign selection numbers to the RS09 Disk Drive. The select wire of each drive is wired to an individual plug in the DISK bank.

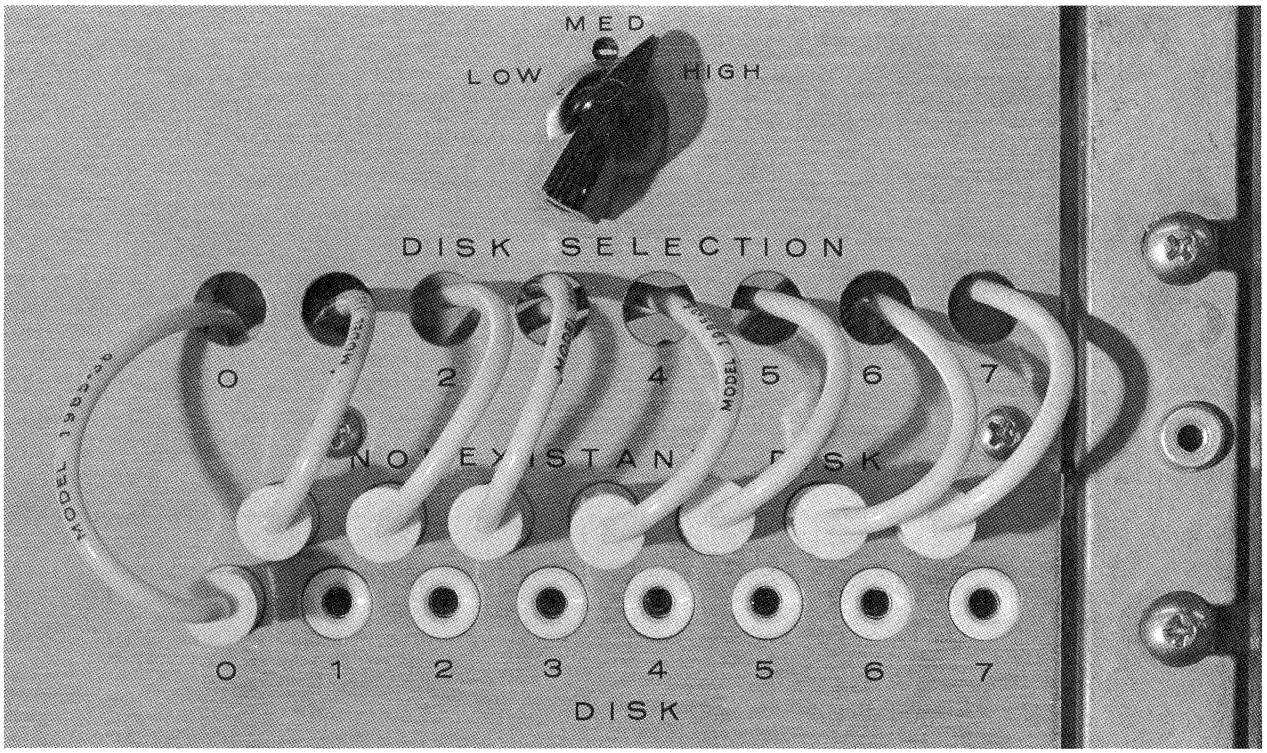


Figure 2-9 Transfer Rate Selection Switch and
Disk Address Select Jacks

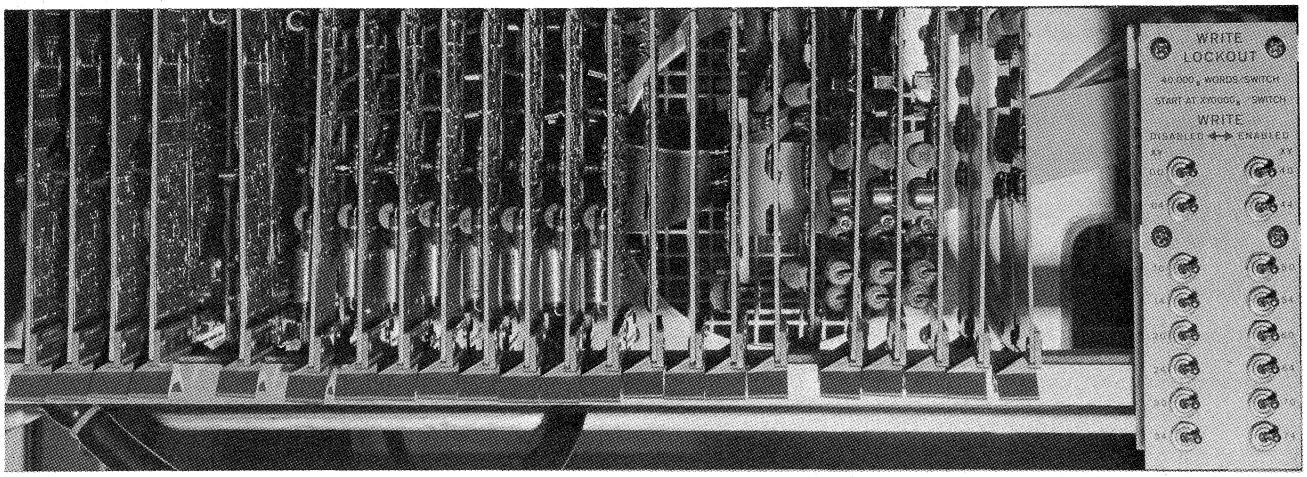


Figure 2-10 Write Lock Out Switches

The select decoder of the controller is wired to the DISK SELECTION jacks. Each disk can be assigned any address by plugging the appropriate DISK SELECTION jack into that disk's plug of the DISK bank. Any jacks that are not assigned should be plugged into one of the NONEXISTENT DISK plugs; a selected nonexistent-disk error can then be detected.

2.3.3 Write Lockout Switches

There are 16 lockout switches on each disk drive (labeled 00 through 74). Each switch protects 8 tracks on the disk. Switch 00 protects tracks 0 to 7_8, switch 04 protects tracks 10_8 through 17_8, and so on, up to track 177_8. When any one of these switches is set to DISABLED, the 8 tracks that they protect cannot be written on. If the program tries to write in such a protected area, the WLO flag (Write LockOut) is posted and writing is inhibited. Figure 2-10 shows the lockout switches. Note that switch 00 actually protects the first head of each shoe, switch 04 the second head, etc. For the programmer, this translates into successive tracks in blocks of 8_{10}.

2.4 THE OPERATOR'S INDICATORS

The operator has at his disposal an extensive indicator panel that reflects the state of the DECdisk System (see Figure 2-11). If a light on the panel is lit, the bit it reflects is set. Table 2-5 summarizes the meaning of each light.

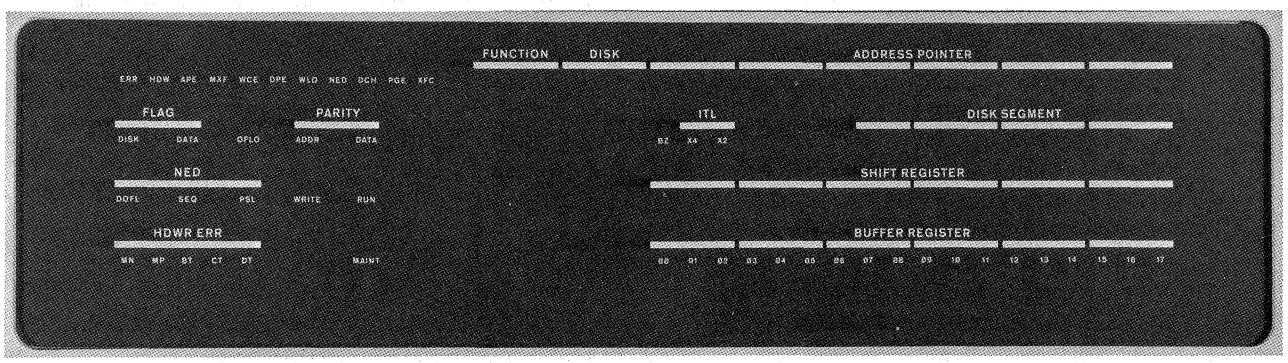


Figure 2-11 Indicator Panel

Table 2-5
The Indicator Panel

Indicator Group	Indicator Name	Indication When Lit
STATUS	ERR, HDW, APE, MXF, WCE, DPE, WLO, NED, DCH, PGE, XFC	The Status Register bits described in Table 2-2 are set.
FUNCTION	-	Three bits of the Function Register decoded in Table 2-1 are set.
DISK	-	Three bits of the Disk Selection Register are set.
ADDRESS POINTER	-	The first 7 bits from left to right indicate the contents of the Track Address Register, and the following 11 bits indicate the content of the Word Address Register.
FLAG	DISK	This level is the logical OR of the two conditions that cause an API or PI break - the ERROR flag and the TRANSFER COMPLETE flag.
	DATA	The flag that requests a multicycle data break is set.
	OFLO	The computer has overflowed its Word Count Register and has set this flag to stop further transfers.
PARITY	ADDR	A parity error on the B or address track has been detected.
	DATA	A parity error on the current data track has been detected.
	BZ	The disk is presently BUSY and engaged in a data transfer.
ITL	X4	When this bit is set, the operator has selected the LOW transfer rate, and every fourth bit is being transferred. (ITL = interlace.)
	X2	Same indication as X4, except that the operator has selected the MED transfer rate.
NED	DOFL	During a transfer, the control sequenced into the ninth disk, which does not exist. (NED = Non Existent Disk.)
	SEQ	During a transfer, the control sequenced into a disk unit that does not exist. The difference between DOFL and SEQ is that SEQ a disk could be added, i.e., the system capacity was not exceeded. With DOFL, the control asked for a disk address greater than 78.

Table 2-5 (Cont)
The Indicator Panel

Indicator Group	Indicator Name	Indication When Lit
NED (Cont)	PSL	A nonexistent disk unit was specified by the program. It was not sequenced under a transfer; the error was a direct programming mistake.
	WRITE	A WRITE operation is taking place.
	RUN	The control is busy and properly synchronized.
	MN	A missing negative pulse or extra positive pulse from the ATT track bipolar signal pair was detected.
HDWR ERR	MP	A missing positive or extra negative pulse from the ATT track bipolar signal pair was detected.
	BT	Any pulse of the bipolar signal pair from the BTT track was detected as missing or extra.
	CT	Any pulse of the bipolar signal pair from the CTT track was detected as missing or extra.
	DT	Any pulse of the bipolar signal pair from the addressed data track was detected as missing or extra.
	MAINT	The controller is in Maintenance mode.

2.5 PROGRAMMING EXAMPLES

2.5.1 Calling Sequence Table

The following program can be used to read, write, or write check any number of the words that can be accommodated in core memory. The program is set up from a calling sequence table that lists the word count, current address, disk number, track number, the address of the first word in the track, and the function to be performed. The execute subroutine that follows enters these variables in their respective registers and commands the disk to execute.

In this sample program, a pointer (DO) is set into auto-increment register 10. Each time the register is indirectly addressed, it is initially incremented by one. The effective address for the first entry is the WC; for the second, the CA; and so on, down the calling sequence table to the FR. With this technique, the execution subroutine sets up the disk and the multicycle data break to carry out the prescribed operation.

```
                                    /Calling Sequence Table
CALTAB      JMS DO                  /Jump to execute subroutine
            0 (WC)                  /2's complement of number of words to be transferred
            0 (CA)                  /Start of memory core data table less 1
            0 (AP0)                 /Disk starting word address and track
            0 (AP1)                 /Disk number
            0 (FR)                  /Function (read, write check) desired
            X                       /Continue program sequence
                                    /Execution Subroutine
DO          0                       /Enter execution subroutine
            LAC  DO                 /Fetch pointer
            DAC  10                 /Deposit pointer in auto index register
            LAC* DO                 /Fetch word count
            DAC  36                 /Deposit in word count register
            LAC* 10                 /Fetch current address
            DAC  37                 /Deposit it in CA register
            LAC* 10                 /Fetch disk starting word and track address
            DLAL                    /Deposit it into its registers
            LAC* 10                 /Fetch disk number
            DLAH                    /Load into disk number register
            LAC* 10                 /Fetch the function
          ( DSCF                    /Clear the function register
*         { DSFX                    /XOR the function register
          ( DSCN                    /Execute the condition held in the function register
            JMP* 10                 /Exit the disk subroutine
```

*Note that these instructions are usually microprogrammed into 707047.

2.5.2 Disk Flag Tests

The disk flag is posted if an error occurs during the transfer or when the transfer is successfully completed. The disk flag causes a PI or API break (if they are enabled) to locations 0_8 or 63_8, respectively, and the program tests for an error or sets up the next transfer from the selected location.

2.5.2.1 Use of IORS - The input/output read status (IORS) facility provides for programmed interrogation of I/O device status (e.g., DECdisk) by groups. When the IORS instruction is executed, the states of all device flags are entered into specific bit positions of the AC. The DECdisk flag is entered into AC bit position 13. Testing the AC contents by groups saves time in locating a specific flag. The following subroutine illustrates the use of IORS to test for the DECdisk flag.

```
FLAGS       0
            IORS
            DAC     SAVE            /Save flags
            AND     MSK1            /Test with group 1 mask
            SNA                     /Skip if flag is located
            JMP     GRP2            /Go to group 2 device test
            JMP     TST1            /Test group 1 device flags
                    .
                    .
                    .
```

```
TST1    LAC     SAVE    /Get group 1 flags
        AND     DMSK    /Mask AC with disk mask
        SNA             /Skip if disk flag is located
        JMP     GRP1A   /Test next subgroup 1A
        JMS     DISK    /Handle disk
          .
          .
MSK1    000777          /Group 1 mask
DMSK    000020          /DECdisk mask
```

2.5.2.2 Skip Chain - The disk flag can be tested by the DSSF instruction if PI or API are not used. The following subroutine lists this procedure.

```
PI      0               /Store the link, extend mode (PDP-9) protect and PC+1
        JMP     FLGS    /
FLGS    IOT SKPA        /Skip if device A flag
        SKP             /Go to next device
        JMS     DEVA    /Handle device A
        IOT     SKPB    /Skip if device B flag
        SKP             /Go to next device
        JMS     DEVB    /Handle device B
          .
          .
          .
        DSSF            /Skip if disk flag
        SKP             /Go to next device
        JMS     DISK    /Handle disk
          .
          .
          .
        ION             /Turn PIE on
        JMP*    PI      /Return to main program
```

The program is now aware that the disk flag has been set. To determine if a successful transfer has taken place, read the status word into the AC by the DSRS IOT. AC bit 0 is the logical OR of all significant error conditions, and it can be quickly tested by the skip on positive accumulator (SPA) instruction. If no skip occurs, an error exists; the next step is accomplished to determine the error and take the required action. The following program illustrates these points.

```
DISK    0               /Store PC + 1, link, EXD (PDP-9) and protect
        DSRS            /Disk read status
        DAC     SAVE    /Save the status
        SPA             /Test for an error condition
        JMP     ER      /Go to error routine
        JMP     XFC     /Go to transfer complete test.  Set program flag
          .
          .
          .
        DBR             /API debreak and restore command
        JMP*    DISK    /
```

The API and PI subroutines usually differ in that the API is kept as short as possible so that it does not tie up the API channel and delay other devices. Techniques for programming the API are explained in the appropriate manual. For simplicity, the same handler for both PI and API is used in this manual.

2.5.3 Error Flag Tests

The error flags that cause an interrupt or API break are classified in three categories, according to the action that should be initiated when they occur. HDW (APE, MXF, and timing errors such as BT, CT, etc.) indicates hardware malfunctions. WLO and NED show that either an operator error was made when the system was initiated, or that the data transferred exceed the capacity of the system to store it. The operator can correct this situation. If WCE or DPE occur, the program itself can take corrective action by determining if the error persists, and subsequently rewriting the erroneous data. Only if a parity error persists should the operator be notified.

The first two classes of error flags should be tested first. If they caused the interrupt (HDW, APE, and MXF; WLO and NED), the program is stopped, and the status is left in the accumulator for the operator to interpret. If the last set of errors occurs, further action can be expected from the program.

EXAMPLE:

```
ER      LAC STATUS      /Get the status
        AND (346000     /Mask out all but the first two classes
        SNA             /Skip if an error occurs
        JMP REWRIT      /It was a soft error, go to rewrite
        LAC STATUS      /It was  a hard error, store the status and notify
                        /the operator
```

Note that the error flags are arranged in descending order of importance so that they can also be tested by successive RTLs skips on link and SMAs.

```
REWRIT          /The parity error was discovered during a read or a write check.
                /The program can either halt, go back and repeat the operation
                /several times to see if the error is still there, or go back and
                /rewrite all the data that has been written erroneously and
                /then retest it, or both.
```

2.5.4 Programming with the ADS Register

The contents of the ADS register reflect the current position of the disk surface. This information is available to the program through the IOT instruction DLOK and can reduce file transfer time between the disk and core memory. Consider the following example, which is illustrated in Figure 2-12.

Example:

> Assume that a file 1777_8 words long is to be transferred from core memory onto a disk. Let the current address (CA) and the word count (WC) be 2000_8 and the 2's complement of 1777_8 (1024_{10}), respectively. Let address pointer 0 (AP0) = 050000 and let address pointer 1 (AP1) = 0. The function is set to WRITE and the transfer rate to HIGH.
>
> Assume further that after the calling sequence has been set up by the program, the ADS register is read into the AC and found to be set to location 660_8. The program would then determine the nearest address to which it could begin transferring data, taking into account the amount of coding that must be processed before the start IOT is issued and the time it takes to switch tracks, if a track must be switched (200 μs), plus set up time. About 240 μs or 15_{10} addresses later is a reasonable figure. In this example, the projected ADS address (PADS) is, therefore, 677_8.

The PADS falls within the area on the disk where the file is to be transferred. The program can now make one of two decisions; it can wait until the disk rotates to location 0 before it starts to transfer data, or it can begin transferring the file at location 677_8. One way to manage the transfer is to divide the file in core into two subfiles. The first subfile starts at core location 2677_8 and transfers to disk location 677_8; it overflows 1101_8 words later. The second subfile starts at the original CA location 2000_8, transfers to disk location 0, and overflows 677_8 words later. This procedure transfers the file in one revolution in its proper sequence, and the time saved from the previous method is approximately the time it takes for one-quarter of a revolution. (See Figure 2-12.)

The more general problem of calculating the two subfiles and determining the PADS address for all three transfer rates is somewhat more complicated. Recall that the ADS Register does not give the adjusted address of the Disk Segment register during medium- or low-speed transfers. During medium transfer rates, this adjusted address can be found by rotating the ADS Register (the 11 least significant bits) one to the right. During low-speed transfers, the adjusted address is calculated by rotating the 11 least significant bits two places to the right. If the first address calculated does not fall into one of the revolutions where the data is s tored for this file, the next address should be tried, and if the transfer rate is LOW, then the following two addresses. This continues until all four revolutions are exhausted, or until a valid PADS is determined. The flow diagram of Figure 2-13 illustrates the process. Assume, for example, that a PADS is calculated and falls into the section shown in Table 2-6. When the transfer rate is HIGH, any address from 74 to 105 is acceptable to determine whether the PADS falls within the file area. When the transfer rate is medium, then it is possible that only every second address belongs to the file. The second line converts the high-speed addresses to their appropriate medium-speed address. If, for example, address 75 converted to 2036 does not fall within the data file, then the program must go on to address 76. Converted, this address is 37 to the medium-speed transfer. If 37 does fall within the file area, the program can begin transferring its file at that point. When low-speed transfer rates are used, four addresses, one for each revolution, may have to be tested for valid PADS points.

Table 2-6
Adjusted ADS Register for Medium and Low Transfer Rates

Transfer Rate	Address									
HIGH	74	75	76	77	100	101	102	103	104	105
MEDIUM	36	2036	37	2037	40	2040	41	2041	42	2042
LOW	17	1017	2017	3017	20	1029	2020	3020	21	1021

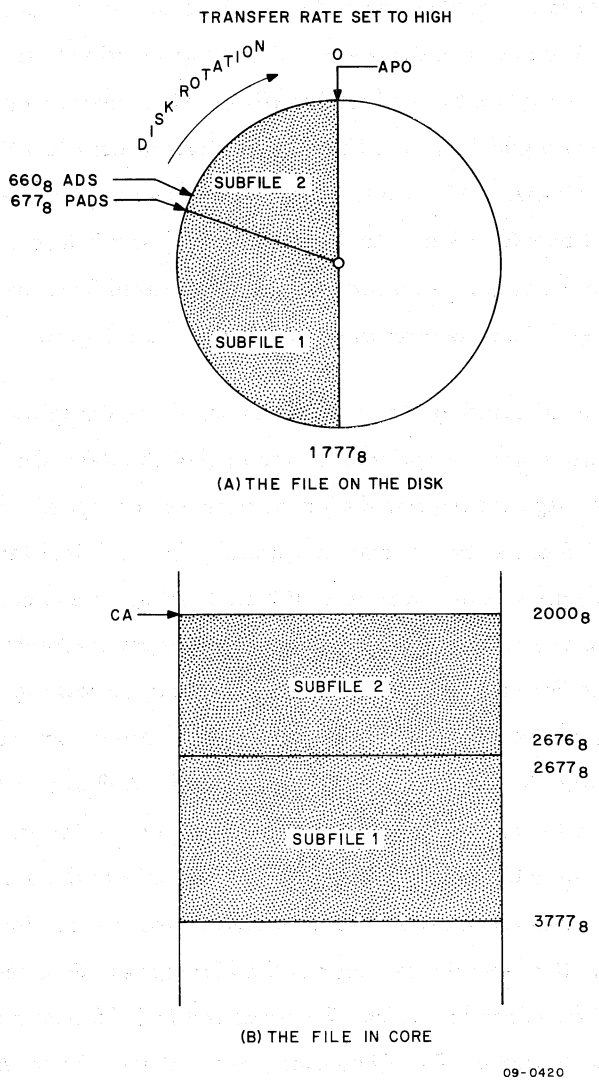


Figure 2-12 Calculating Fast Access Calling

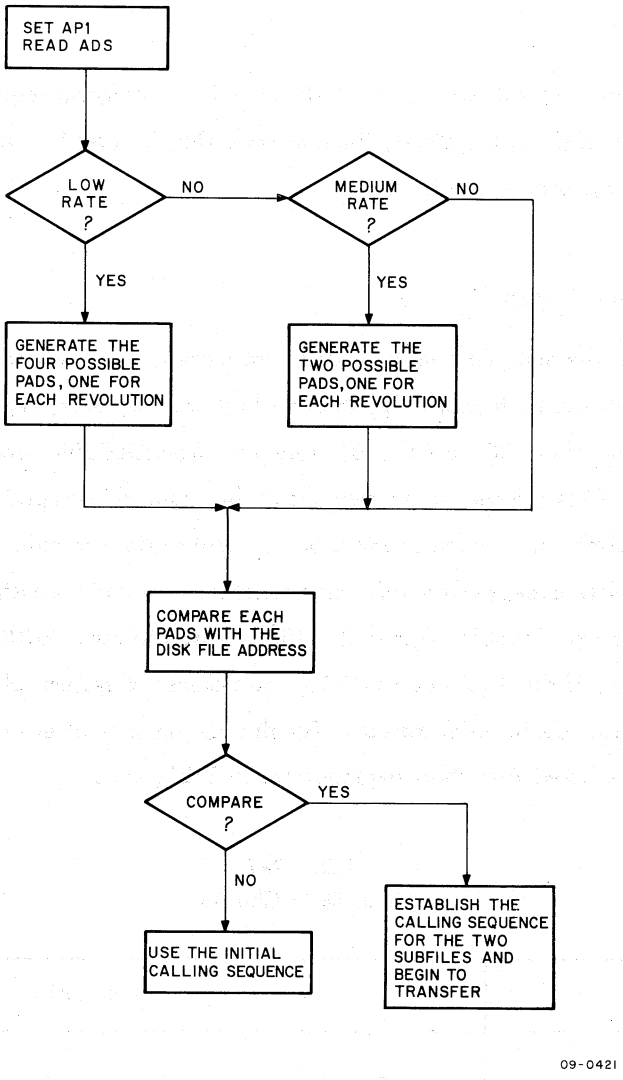


Figure 2-13 Flow Diagram of the Subroutine That Uses the ADS Register

2.5.5 Programming Multiple-Disk Systems

Sequencing from track-to-track and disk-to-disk is program-transparent except for the latencies that occur when switching from disk-to-disk. The disks are not synchronized. The latency can be reduced by using the ADS register and the techniques described in Paragraph 2.5.4. Ensure that the ADS register is read and the correct disk has been selected; i.e., that address pointer 1 has been properly set up (see Figure 2-13).

If API is set to a disk that does not exist in the system, or if the program sequences into a nonexistent disk (such as disk 9 in an eight disk system), then an error flag is posted. Note that the eighth disk does not overflow and wrap around to disk 0.

2.5.6 Using DECdisk in a System

DECdisk is as reliable as the main core memory and considerably more reliable than industry compatible tape. The disk should always be supported by another bulk memory unit, typically DECtape or industry-compatible tape. It takes about 30s to fill a DECtape reel from DECdisk, and each disk surface fills two such reels. As data files are generated they should be regularly dumped from the disk into its support memory. How often this is done depends on the application; in most systems, this job can become a background activity except when very important files are under construction. DECdisk may cause approximately one irretrievable error in 2×10^9 bits transferred. With most information, this is not a problem. However, if the error occurs during the transfer of system software or the accumulation of a payroll file, the result could be disastrous. For this reason several error detecting techniques have been devised. These are listed with short explanations in Table 2-7.

Table 2-7
Disk Data Checks

Name	Explanation
Lateral Parity Checking	This test is automatically performed by the hardware each time a word is read, written, or write checked.
WRITE CHECK	This function checks the disk itself. It compares the file in core with the file as it should have been written in memory. The checking is done at the controller, however; consequently, consistent errors in the data paths are not detected.

Table 2-7 (Cont)
Disk Data Checks

Name	Explanation
WRITE then READ	This technique tests the disk, the data paths, and core memory itself. It copies the file onto the disk, and then reads it back into core into a different area of core memory. The two files are then checked for consistency. (The overhead is high.)
Longitudinal Parity Check	This technique tests the disk and the data paths and core. When a table or file is built, a longitudinal checksum is calculated with it. Whenever the table is transferred, the checksum is recalculated and compared with the original. (The overhead is very high.)
Error Detecting and Correcting Codes	Hamming codes that can be generated for each word automatically and correct some errors when they occur. The overhead when this is done with software is usually prohibitive.

One additional short test can be run on a file after it has been transferred: add the original word count to the original APO and compare the result to the APO just after the transfer. The two values should be identical.

2.6 SUMMARY OF DECDISK CHARACTERISTICS

The following is a summary of the DECdisk System characteristics:

a. Storage Information
   (1) fixed head
   (2) serial, random access
   (3) 8 disks per controller
   (4) 128 data tracks per disk
   (5) 2048 eighteen-bit words per track
   (6) 262,144 eighteen-bit words per disk
   (7) 2,097,152 eighteen-bit words per disk system

b. System Transfer Rates

   Three switch-selectable speeds: 16 μs per word;
   32 μs per word;
   64 μs per word.

c. Protection

   Tracks on each disk are protected from a write operation in groups of eight (a total of 16,384 words).

d. Access

(1) 16.7 ms (average) when the ADS register is not used

(2) 200 μs if the ADS register is used

e. Reliability

Six recoverable errors and one nonrecoverable error in 2×10^9 bits transferred. (A recoverable error is defined as an error that occurs only once in four successive reads.)

f. Core Locations

Automatic Priority Interrupt	63 on level 1
Data Channel	36, 37

Chapter 3
The DECtape System

3.1 INTRODUCTION

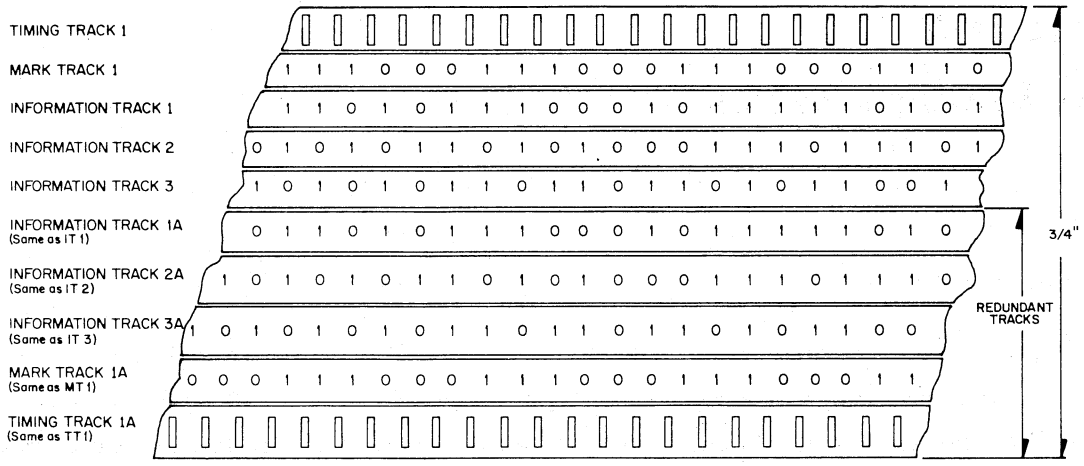
The DECtape System is a magnetic tape data storage facility. The system, consisting of up to four TU56 Dual DECtape Transports, or eight TU55 single DECtape Transports, and a TC15 DECtape Control, stores information at fixed positions on magnetic tape as in magnetic disk or drum storage devices, rather than at unknown or variable positions in conventional magnetic tape systems. This feature allows replacement of blocks of data on tape in an ordered fashion without disturbing other previously recorded information. In particular, during the writing of information on tape, the system reads format (mark) and timing information from the tape and uses this information to determine the exact position at which to record the information to be written. Similarly, in reading, the same mark and timing information is used to locate data to be played back from the tape.

This system has a number of features to improve its reliability and make it exceptionally useful for program updating and program editing applications. These features are: phase- or polarity-sensed recording on redundant tracks, bidirectional reading and writing, and a simple drive mechanism that uses aerodynamically lubricated tape guiding (the magnetic tape surface floats on air and does not touch any metal surfaces except the head).

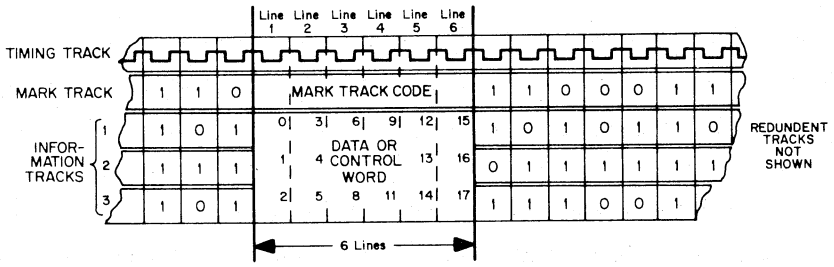
3.2 DECTAPE FORMAT

DECtape utilizes 10-tracks with a read/write head for each track. Tracks are arranged in five non-adjacent, redundant channels: a timing channel, a mark channel, and three information channels (see Figure 3-1). Redundant recording of each character bit on nonadjacent tracks materially reduces bit drop-out and minimizes the effect of skew. Series connection of corresponding track heads within a channel and the use of Manchester phase recording techniques, rather than amplitude sensing techniques, virtually eliminate dropouts. The timing and mark channels are preformatted prior to recording all normal data read and write functions.

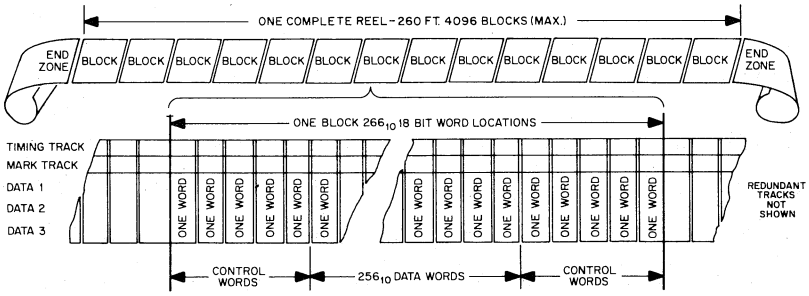
A reel of DECtape is 260 ft long and is divided into two end zones of 10 ft each and a recording zone of 240 ft. The end zones are merely used as leader-trailer to wind the tape around the heads and onto



Track Allocation Showing Redundantly Paired Tracks



Basic Six Line Tape Unit



Control and Data Word Assignments

Figure 3-1 DECtape Format

the take-up reel. Timing and mark channels are recorded prior to recording the data. The 240-ft recording zone is divided into blocks. Each block, which contains a specified number of data words, is separated by a number of control words on the mark track. The number of data words per block is determined by information on the mark track when the tape is preformatted. The standard PDP-15 DECtape format is 576 blocks of 256 18-bit data words. Each 18-bit data word uses six data lines, and each data line contains three bits. The total length of tape is equivalent to 884,736 data lines which can be divided into any number of blocks up to 4096. Blocks are normally limited to a total of 4096 to provide compatibility with 12-bit computers such as the PDP-8.

3.2.1 Timing Track

The timing of operations performed by the tape drive and some control functions are determined by the information on the timing channel. Therefore, wide variations in the speed of tape motion do not affect system performance.

3.2.2 Mark-Track Format

The mark track is reserved for control codes (see Table 3-1 and Figure 3-2) which are stored serially. The codes are 6-bits long and are used for initiating controls to raise flags in the program, requesting data breaks, detecting block mark numbering and block ends, and protecting control portions of tape. The TC15 DECtape Controller automatically identifies these codes to control transmission of data. The mark track also provides for automatic bidirectional compatibility, variable block formatting, and end-of-tape sensing.

During all tape processing functions except recording of the timing and mark track, a single mark-track bit is read from each line of tape, regardless of whether the information is being read from or written onto the data tracks; and each tape line in both the information and mark tracks is positioned at the center of the correct polarization in the timing track, as shown in Figure 3-1. The six lines on the tape that contain the mark code in the mark track are designated as a mark frame. A mark frame is a group of six tape lines. The six bits in the mark track are called a mark code, as shown in Figure 3-1.

A given change of polarization on tape read in one direction produces a pulse opposite in polarity to that produced by the same change when tape is read in the opposite direction. Consequently, a mark code read in reverse of the direction in which it was recorded has the order of bits reversed and the bits complemented. For example, a mark code read forward as 100010 is read as 101110 in reverse (see Figure 3-3). This correspondence is termed the complement obverse or the complement image. Every 6-bit code has one and only one complement obverse, which is constructed by complementing all bits and reversing their order.

Table 3-1
Mark Track Coding

Mark	Octal Code	Function
Reverse End Mark	55	Indicates the tape is progressing from the end zone toward the control words and data blocks.
Expand Mark	25	Allows DECtape compatibility between 18- and 36-bit machines.
Forward Block Mark	26	Signifies the start of a block. Block number is stored in memory, and, when the TC15 controller decodes code 26, the computer obtains the forward block number.
Reverse Guard Mark	32	Allows time for the computer to decide what to do with the forward block number.
Lock Mark	10	Indicates first of four octal 10 cells and allows for computer-decision time.
Reverse PCC Mark	10	Indicates last cell before a data word, and is used to initiate parity checksum logic. For write operations, the first 18-bit word to be written onto tape should be transferred from the computer to the controller during reverse PCC mark.
Reverse Final Mark	10	Indicates first data word.
Reverse Prefinal Mark	10	Indicates second data word.
Data Mark	70	Indicates a data word is contained in data track.
Prefinal Mark	73	Indicates next to last data word of block.
Final Mark	73	Indicates last data word in block and shows that next cell begins with the forward longitudinal parity check.
PCC Mark	73	For a write operation, the parity checksum is deposited in this cell. For a read operation, the value written in this cell is compared with the value of the checksum due to the read operation.
Reverse Lock Mark	73	Provides additional time to protect records in the event of mark track errors.
Guard Mark	51	Performs same function as reverse lock mark (code 73).
Reverse Block Mark	45	Complement obverse of forward block mark and is used when tape is travelling in reverse direction.
Expand Mark	25	Same as expand mark previously described.
End Mark	22	Identifies end zone and indicates transport is running out of tape.

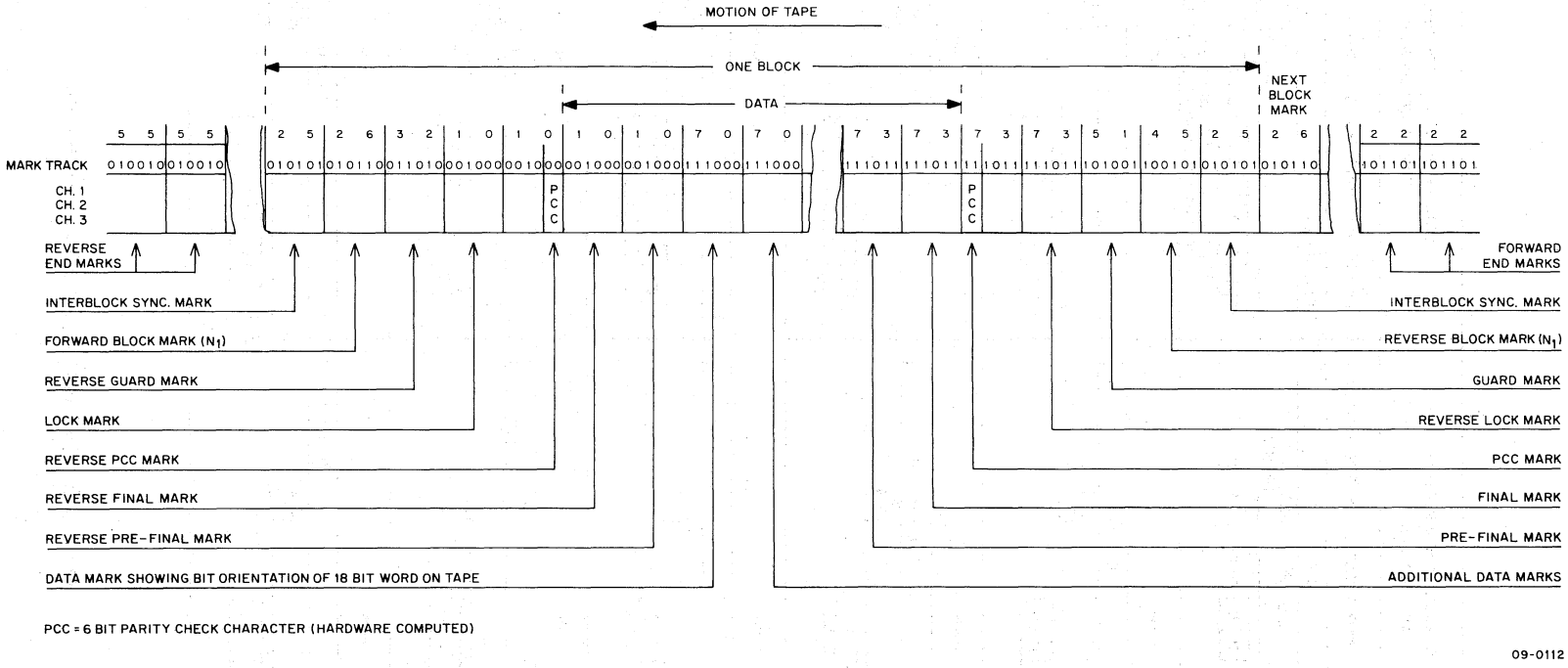


Figure 3-2 Mark-Track Format

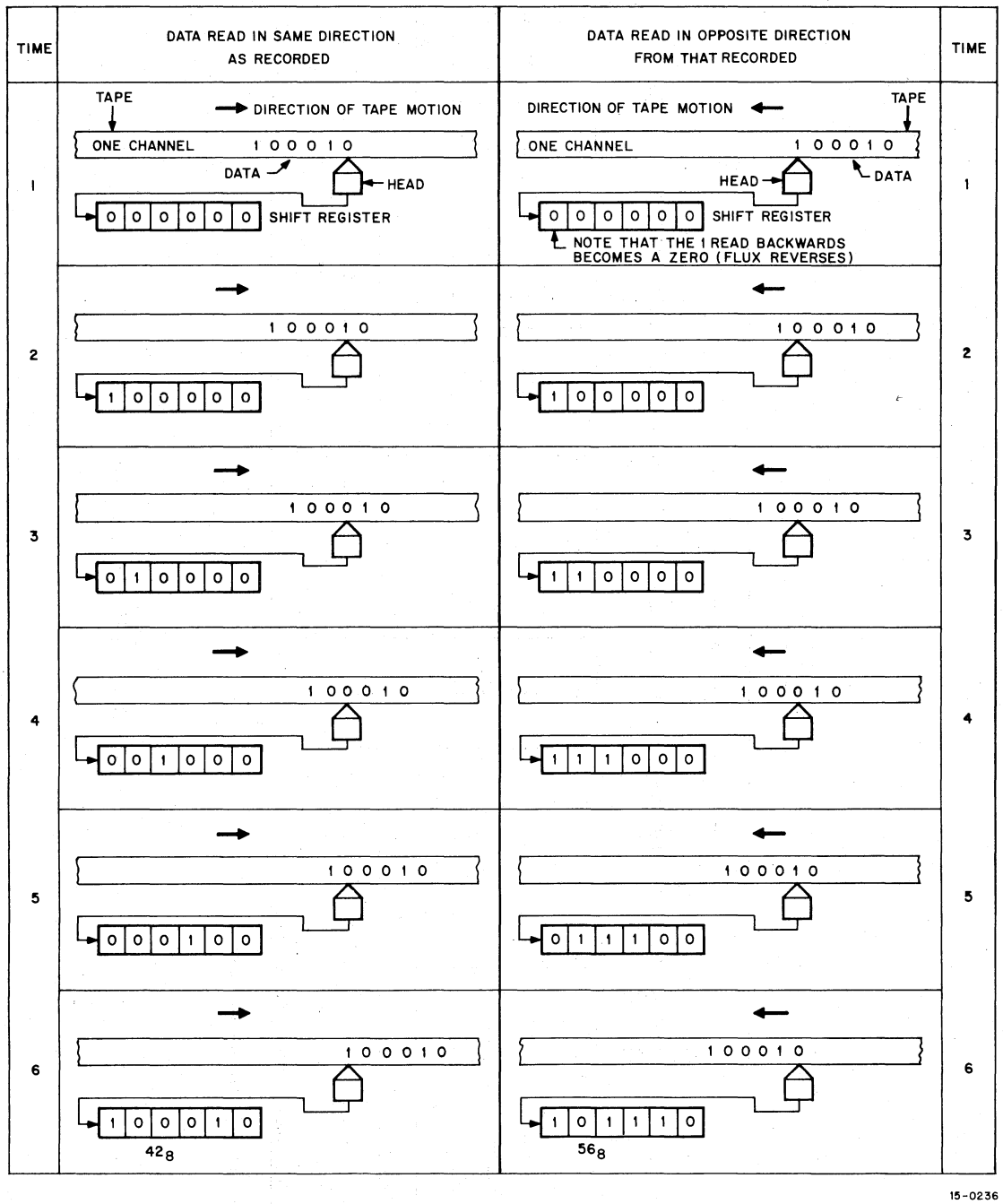


Figure 3-3 Bidirectional Reading and Writing

There are eight octal codes of two digits that are their own complement obverses: 07, 13, 25, 31, 46, 52, 64, and 70. All other possible combinations of two octal digits (there are 56) are different from their complement obverses.

Because the DECtape System allows reading and writing in both directions of tape motion, the mark track must be coded to present the same information when entering a block from either direction. The marks at the end of a block are the complement obverses of the marks at the beginning, in reverse order. For example, assume the control reads the marks 25, 26, and 32 as the first three marks beginning a block in forward motion. In reverse motion, the control sees the obverse complement of the contents of the mark track. The first information when reading the block in reverse is therefore, also read as 25, 26, and 32. Mark codes to be read in when tape is moving in reverse direction are recorded in obverse complement. Thus, the mark track appears symmetrical.

All mark codes used in the standard DECtape format are listed in Table 3-1. Only 10 valid codes exist even though a given code may have different designations, such as 10 and 73.

The standard mark track uses the serial code of 6-bit characters to divide the tape into words. Codes are written on the mark track opposite word locations to identify the type of information stored at that location on tape. Block addresses are written for both forward and reverse directions and identified by two types of mark codes (the second is the complement obverse of the first). A checksum is written at each end of the block. The hardware computed checksum is the 6-bit logical equivalent (i.e., the complement of the exclusive OR) of each six bits written on tape, plus the reverse checksum previously recorded. By including the reverse checksum in the computation, the block may be read in either direction at a later time without an error. The control uses the final marks to establish synchronism and raise block-end flags. Data marks locate data words.

3.2.3 Data Blocks

A tape contains a series of data blocks that can be of any length that is an even number of 18-bit words. Block length is determined by information on the mark channel. Usually a uniform block length (256_{10} for the PDP-15) is established over the entire length of a reel of tape by a program that writes mark and timing information at specific locations. The ability to write variable-length blocks is useful for certain data formats. For example, small blocks containing index or tag information can be alternated with large blocks of data. The maximum number of blocks addressable is 4096 when 12-bit machine compatibility is required. Otherwise, the number of blocks can be increased to 13,928.

A block is identified by a number recorded during tape formatting on the data tracks just before and after the area where data are stored in the block. This number is recorded at either end of a block; thus, it can be read from either direction.

Block numbers normally occur on tape in sequence from 0 to N-1, where N is the number of blocks. The total length of tape is equivalent to 884,736 data lines per tape, which can be divided into any number of blocks by prerecording of the mark track. However, 576_{10} blocks of 256_{10} words are considered to be standard format for a PDP-15 DECtape.

3.3 TU55 AND TU56 DECTAPE TRANSPORTS

The TU55 DECtape Transport is a bidirectional magnetic-tape transport consisting of a read/write head for recording and playback of information on DECtape. The TU56 Transport is similar to the TU55 but is a dual transport (two transports mounted side-by-side on a single chassis). Connections from the read/write heads in the transport are made directly to the TC15 DECtape Control, which contains the associated read and write amplifiers. The logic circuits in the DECtape Transports control tape movement in either direction over the read/write head.

Tape drive motor control is accomplished by regulating the torque of two motors that transport the tape across the head according to the established function of the device, i.e., go, stop, forward, or reverse. In normal tape movement, full torque is applied to the forward or leading motor, and a reduced torque is applied to the reverse or trailing motor to keep proper tension on the tape. Tape motion is bidirectional; as a result, each motor serves as either the leading or trailing drive for the tape, depending on the forward or reverse control status of the DECtape transport.

Tape movement can be controlled by commands originating in the computer and applied to the TU55 or TU56 through the TC15 DECtape Control, or can be controlled by commands generated by manual operation of rocker switches on the front panel of the transport. Manual control is used to mount new reels of tape on the transport, or to perform a quick maintenance check for proper operation of the control logic in moving the tape.

3.4 TC15 DECTAPE CONTROL

A maximum of eight TU55 DECtape or four TU56 transports can be connected to one TC15. Of the four data channels available, DECtape is assigned to channel 0 (i.e., core memory locations 30 and 31).

C(30) = Word Count (in 2's complement form) - WC
C(31) = Current Address Register - CA

Data transfers can occur to or from only one transport at any given time at a rate of one word every 200 μs ± 60 μs (1 block of 256_{10} words every 53 ms), after the desired block has been found.

The CA is incremented before the data transfer (except in SEARCH where the CA is not incremented); thus, the initial contents should be set to the desired initial address minus one. The WC is also incremented before each transfer and must be set to the 2's complement of the desired number of data words to be transferred. In this way, the word transfer that causes the word count overflow is the last transfer to take place.

3.5 DECTAPE INSTRUCTION SET

The number of IOTs required for the TC15 is minimized by the scheme of transferring all necessary DECtape control data (i.e., unit, function, mode, direction, etc.) from the AC to Status Register A in the DECtape Controller, using one set of IOTs. Similarly all status information (i.e., all above information plus status bits, error flags, etc.) can be read into the AC from Status Register B in the DECtape Controller via a second set of IOTs (see Table 3-2).

A bit of each status register is set or cleared by its corresponding bit in the AC when the proper load status IOT is issued. Similarly, when the read status IOT is issued, each bit of the status register addressed is read into its corresponding accumulator bit.

Table 3-2
TC15 Control IOT Instructions

Mnemonic	Octal Code	Description
DTCA	707541	Clear status register A. The DECtape control and error flags are undisturbed (DTF and EF).
DTRA	707552	Read status register A. The AC is cleared and the content of status register A is ORed into the accumulator.
DTXA	707544	XOR status register A. The exclusive OR of the content of bits 0 through 9 of the accumulator and status A is loaded into status register A, and bits 10 and 11 of the accumulator are sampled to control clearing of the error and DECtape flags, respectively. Any time this command is given with AC bits 0-4 set to 1, the select delay of 120 ms will be incurred.
DTLA	707545	Load status register A. Combines action of DTCA and DTXA to load AC0-9 into status register A. Bits 10 and 11 control clearing of error and DECtape flags, respectively.
DTEF	707561	Skip on error flag. The state of the error flag (EF) is sampled. If it is set to 1 the contents of the PC is incremented by one to skip the next sequential instruction.

Table 3-2 (Cont)
TC15 Control IOT Instructions

Mnemonic	Octal Code	Description
DTRB	707572	Read status B. The AC is cleared and the content of status B is ORed into the accumulator.
DTDF	707601	Skip on DECtape flag. The state of the DECtape flag (DTF) is sampled. If it is set to a 1, the content of the PC is incremented by one to skip the next sequential instruction.

All 10 comand bits (0 to 9) of status register A may be sensed, set, or changed via IOTs (see Table 3-3 for Status Register A bit assignments). Bits 10 and 11 of the AC are not retained by status A, but enable or disable the clearing of the DECtape and ERROR flags. To issue a DECtape command, the command bits 0-9 of status register A are set as desired by bits 0-9 of the AC with bits 10 and 11 set to 0. The bits in status register B can be sensed and cleared by IOTs (see Table 3-4 for Status Register B bit assignments). Bit 11 of register B is set when a DTF occurs and must be cleared before the next DTF to avoid a timing error. If any error occurs, bit 0 of register B and the corresponding bit (1-5 depending on the error) will be set. This bit must be cleared to avoid further interrupts on the same condition. All error flags (i.e., status register B) are cleared by issuing a DTXA instruction with AC bit 10 set to 0. The DONE flag is cleared by issuing a DTXA instruction with AC bit 11 set to 0.

Table 3-3
Register A Bit Assignments

Function	Bit	Conditions
Transport unit select (decodes one of eight DECtape transports)	0-2	Octal Code / Transport Unit 000 — 8 or 0 001 — 1 010 — 2 011 — 3 100 — 4 101 — 5 110 — 6 111 — 7
Motion (determines forward or reverse motion)	3	0 = Forward (FWD) 1 = Reverse (REV)
STOP/START	4	0 = Stop (STOP) 1 = Start motion (GO)

Table 3-3 (Cont)
Register A Bit Assignments

Function	Bit	Conditions
MODE	5	0 = Normal Mode (NM) 1 = Continuous Mode (CM)
Function	6-8	Octal Code — Operations 000 — Move 001 — Search 010 — Read Data 011 — Read All 100 — Write Data 101 — Write All 110 — Write Timing 111 — Unused (causes select error)
Interrupt Disable	9	1 = Enables DECtape flag or error flag to initiate PI or API interrupt. 0 = DECtape flag or error flag inhibited from initiating PI or API interrupt.
Error Flag	10	0 = Clear all error flags 1 = Error flags undisturbed
DECtape flag	11	0 = Clear DECtape DONE flag. 1 = DECtape DONE flag undisturbed

Table 3-4
Status Register B Bit Assignments

Function	Bit	Conditions
Error Flag	0	Set if error occurred (inclusive OR of bits 1-5)
Error Indicators	1-5	Bit 1 = Mark Track Error Bit 2 = End of tape error Bit 3 = Select Error Bit 4 = Parity Error Bit 5 = Timing Error
	6-10	Not used
DECtape Flag	11	DECtape flag set at completion of selected function

3.6 DATA FLOW

Data are transferred between the central processor and DECtape via a double-buffered (DECtape buffer and data buffer) arrangement in the TC15 DECtape Controller. To write data onto DECtape, a three-cycle data channel request is initiated, which causes the 18-bit data word to be transferred from the central processor to the DECtape buffer and then to the data buffer. The data word is subdivided into groups of three bits - each bit going to a specific data track write head. Longitudinal parity is generated as the data are transferred to the write heads and written onto the tape by the hardware.

To transfer data from DECtape to the central processor, the reverse process occurs. The three read heads transfer the data in three-bit bytes to the data buffer. The contents of the data buffer, when full, are transferred to the DECtape buffer. A three-cycle data channel request is initiated to transfer the contents of the DECtape buffer to the central processor. This transfer must take place within 200 μs (± 60 μs) or the contents of the data buffer will be transferred to the DECtape buffer before the DECtape buffer has transferred the preceding word.

3.7 DECTAPE PROGRAMMING CONSIDERATIONS

3.7.1 Control Functions

The seven functions available with the TC15 and their octal codes as specified by the bits 6-8 of the AC are as follows:

Function	Octal Code
MOVE	0
SEARCH	1
READ DATA	2
READ ALL	3
WRITE DATA	4
WRITE ALL	5
WRITING TIMING AND MARK TRACK	6
Unused at present (select error if given)	7

All functions take place in either direction and in either normal mode (NM) or continuous mode (CM). NM differs from CM only in the fact that the DECtape flag (DTF) occurs at more frequent intervals in NM. The DTF settings that occur in NM are eliminated in the CM until word count overflow (WC) has occurred.

3.7.2 MOVE Function

The MOVE function, with the GO bit set to 1, sets the selected unit in motion (forward or reverse). NM and CM have no meaning and are ignored in this function. When the tape enters either end zone* (i.e., beginning of tape (BOT) and end of tape (EOT)) and the unit in question is selected, then:

a. The error flag (EF) is set.
b. The EOT bit (bit 2 of status register B) is set.
c. An interrupt occurs**.

A program check on the forward/reverse motion bit (AC bit 3) of the status register will determine whether EOT or BOT occurred. However, if the unit is deselected while in motion, the tape runs off the reel with no flags raised and no interrupt. To stop a selected unit at any time, the GO bit (AC bit 4) must be set to 0. When setting the GO bit to 0, the forward/reverse motion bit and unit selection bits should not be changed from their current status. The hardware accepts the change in the TC15 (i.e., status A bit 3 changes from its former state) without error indication, but does not pass this change on to the transport. Programming confusion can result. After a unit is deselected, status information pertaining to that unit is no longer accessible unless it was saved by the program prior to deselection.

3.7.3 SEARCH Function

The SEARCH function permits random access of data blocks on DECtape. This function is used to locate the number of block to (or from) which data transfer will occur. In normal mode at each block mark until EOT occurs, the DTF is raised and an interrupt occurs. The block number is automatically transferred by the hardware into the memory location specified by the CA. The CA must have been set previously by the program, but the contents are not incremented. The WC is incremented at each DTF; the program must clear the DTF bit in the status register and check the block number until the desired one is found.

In continuous mode, the WC is set to the 2's complement of the number of block numbers to skip. At each block mark, the block number is read into the memory location specified by the CA which is not incremented. The DTF is raised only at the block mark at which the WC overflows. At that time, an interrupt occurs and the block whose number was just read may be read or written. Continuous mode provides a virtually automatic DECtape search.

---

*If either end zone is entered during turn around or during stopping of tape, the EOT bit is not set and no interrupt occurs.

**All references to the occurrence of interrupts assume both:
1. The program interrupt is on.
2. The DTF and EF have been enabled to the program interrupt or API (i.e., bit 9 of status register A is set to a 1). If either of these is not true, flags are raised and status bits are set (and may be sensed and/or cleared), but no interrupt occurs.

3.7.4 READ DATA Function

READ DATA is used to transfer blocks of data into core memory with the transfer controlled by the standard tape format. The standard block length is 256 18-bit words. For this and all following functions, the CA register must be set initially to the transfer memory location minus one because the CA register is incremented just before each word transfer. The WC register is also incremented prior to each word transfer so it must be set to the 2s complement of the number of words to be transferred prior to the transfer. Data may be transferred in forward or reverse direction.

Any number of words equal to or less than one block may be transferred in NM. The DTF is raised and an interrupt occurs at the end of each block. The DTF must be cleared before the beginning of the next block (1.7 ms) to avoid an erroneous timing error, (see Summary). When partial blocks are transferred, data transmission ends with WC overflow. The word that causes the WC overflow is the last one transferred. However, the remainder of the block is read and parity checked before the DTF and interrupt occur. Tape motion continues until the GO bit is reset to 0 by the program. If the GO bit is not reset to a 0 or a new function is specified before the end of the next block, a timing error will occur. READ DATA in NM is intended primarily for single, 256-word, block transfers. If any other number of words is to be transferred, it is advantageous to use CM. However, if the programmer uses NM for any other number of words, the program must check for WC overflow at each interrupt because there is no other way to determine when to stop the tape or change to another function. When the WC overflow occurs, it is essential that the function be changed or the GO bit set to 0. Otherwise, transfer begins again (the IOT to clear the DTF implicitly specifies the same function again) at the next block (or next word for the ALL functions) because WC equals 0.

Any number of words may be transferred in CM. However, the DTF and an interrupt occur only once after a WC overflow and an end of block. The comments concerning tape continuation apply in CM as well as NM.

3.7.5 READ ALL Function

The READ ALL function allows information to be read from a tape that is not in the normal format. This function reads all data channels recorded on DECtape, regardless of the mark track value. During the READ ALL function, the DECtape control does not distinguish between different marks recorded on the mark track, except to check for mark track errors (MKTK).

In normal mode (NM), the DTF is raised and causes an interrupt at the end of each 18-bit word transfer. Data transfer stops after WC overflow, but tape motion continues until the GO bit is set to 0 or a new function is specified (in both NM and CM). If the DTF is not cleared after each word transfer, a timing error occurs at the end of the next word (200 μs later).

For continuous mode, the DTF is raised and causes an interrupt at WC overflow only. When this interrupt is ignored, no more data transfers occur but tape motion continues to EOT.

3.7.6 WRITE DATA Function

The WRITE ENABLE switch on the transport must be in WRITE ENABLE position for all WRITE functions. All the details of the READ DATA function description apply with the following exceptions.

a. In normal mode, the DTF is set to a 1 at the end of each block. If WC overflow did not occur in the block just ended and no new function is specified, the next block will be written, provided the DTF has been cleared. If WC overflow occurred in the block just ended and no new function is specified, the tape continues to move but the writers are disabled. In both CM and NM, when partial blocks are written, data transfer from core to DECtape stops at WC overflow. 000000s are written in the remaining data words of the block and the parity check character is computed over the entire block and recorded.

b. In continuous mode, the DTF is set at the end of the block in which WC overflow occurred. Therefore, if no new function is specified, the tape continues to move, but the writers are disabled.

c. A six-bit parity check character (PCC) is computed (the XOR of the reverse parity check character and every six bits of every data word) and recorded by the DECtpae control for every block of data recorded during the WRITE DATA function. It is used for automatic parity checking during the READ DATA function.

3.7.7 WRITE ALL Function

All the details of the READ ALL function description apply. The WRITE ALL function is used to write an unusual format (such as block numbers on DECtape after timing and mark tracks have been recorded). The word which causes WC overflow is the last one written in NM or CM. The tape continues to move, but the writers are disabled.

NOTE

Change of function must be delayed for 90 μs to ensure recording of last word. Alternative method: set WC to 1 greater than desired number of word transfers, and change function within 40 μs after WCO.

3.7.8 WRITE TIMING and MARK TRACK Function

This function and only this function may be performed with the selector switch on write timing and mark track (WRTM) on the maintenance control panel. The timing track is actually hardware-recorded during execution of this function. The mark track is generated and recorded by program. The value written in the mark track is determined by bits 0, 3, 6, 9, 12, and 15 of the 18-bit word being written (i.e., the same bits assigned to channel 1).

CM may be used for this function, because the hardware WC provides an automatic counter and interrupt at WC overflow only; in NM, the DTF and interrupt occur at every word until WC overflow. In NM, after WC overflow, if the GO bit or DECtape flag is not cleared, a timing error occurs and no more data are recorded. After WC overflow in CM, if the GO bit is not set to 0, zeros are written on tape.

3.7.9 Disable Interrupt

The disable interrupt feature allows the program to effectively remove DECtape from the interrupt line.

When command bit 9 in the status register is set to a 1, the TC15 is connected to the interrupt system. If this bit is 0, the DTF in the TC15 cannot cause an interrupt even if the interrupt facility in the PDP-15 is ON. Similarly, any of the five error conditions will cause an interrupt if bit 9 is set to 1 in the status register, but cannot cause a program interrupt if bit 9 is a 0.

Whether this bit is set or not does not influence the setting of status bits 0-5 of the status register B on receipt of an error flag (EF) or DTF. Similarly, the result of the I/O skip instruction is independent of the condition of this bit.

3.7.10 Error Conditions

Five types of errors can be detected in the use of DECtape:

a. Timing Error
b. Parity Error
c. Select Error
d. End of Tape
e. Mark Track Error

For all errors the EF is raised, a bit is set in the status register, and an interrupt occurs (if the enable-to-interrupt bit has been set). The DTEF instruction skips on the inclusive OR of those error bits; hence, each status bit must be checked to determine the kind of error. For all but the parity error, the selected transport is stopped, and the EF is raised at the time of error detection. No DTF occurs. For a parity error, the GO bit remains 1 (i.e., motion continues), and the EF is raised simulataneously with the DTF in NM. Only 1 interrupt occurs; hence, the program must check the EF.

A parity error in CM raises the EF at the end of the block in which the parity occurs causing an interrupt (if enabled). If no program action is taken (e.g., stop transport or reverse and re-read), data transfer continues and the DTF is raised. The DTF causes an interrupt at WC overflow and end of final block read.

3.7.10.1 Timing Error - A timing error (program malfunction) is a 'data miss' or program failure to clear DTF status bit. A timing error occurs also if the program switches to READ or WRITE DATA function while the DECtape is currently passing over a data area on tape.

3.7.10.2 Parity Error - A parity error occurs only during the READ DATA function for a hardware computed parity check character (PCC) failure.

3.7.10.3 Select Error* - A select error results from any of the following conditions:

a. Attempt to select unit when two or more DECtape transports are set to the same unit number and both are on REMOTE.
b. Attempt to write on DECtape transport with WRITE ENABLE/WRITE LOCK switch in the WRITE LOCK position.
c. Attempt to select unit for any function with DECtape transport REMOTE/OFF/LOCAL switch in the OFF or LOCAL (off-line) position.
d. Attempt to write timing and mark tracks with the DECtape controls switch in any position other than write timing and mark track.
e. Attempt to perform any function other than write timing and mark tracks with the DECtape control switch in the write timing and mark track position.
f. Attempt to perform any function other than read all with DECtape control switch in the read mark track position.
g. Attempt to execute unused function (7).

3.7.10.4 End of Tape (EOT) - An EOT error occurs when the DECtape enters either end zone with the GO bit = 1 and the forward/reverse direction bit set to continue in the same direction. In NM and CM, data transfer stops at the last legitimate block, the EF is raised, the tape transport stops, and an error interrupt occurs.

3.7.10.5 Mark Track Error - A mark track error occurs when the DECtape control fails to recognize a legitimate mark on the mark track. The error may occur in all but the move or write timing and mark track functions. In both CM and NM, the EF is raised, the tape transport stops, and an interrupt occurs.

---

* No-tape or tape-run-off-reel conditions are not detectable.

3.8 PROGRAMMING EXAMPLES

The following examples illustrate how several DECtape functions can be programmed. In the first example, a specific block is searched out and found, and in the second example, data is read from this block. The subroutines are written in PDP-15 Basic Symbolic Assembler language. Since no indirection is used, the example is intended for Page 0 operation.

3.8.1 Automatic Search

```
BEGIN   LAC (CBLK       /GET THE ADDRESS WHERE THE
                        /BLOCK NO. GOES
        DAC 31          /PUT IT INTO THE CA
        DZM 30          /ZERO OUT THE WC TO AVOID
                        /OVERFLOW
        LAC (JMP SEARCH /GET THE EXIT POINT
        DAC SWITCH      /PUT IT INTO SWITCH
        LAC (321400     /GET THE STATUS A
        DTLA            /LOAD STATUS A REGISTER
44      JMS DECTAP      /API SETUP
DECTAP  0               /SAVE PC, LINK, EXTEND MODE
                        /AND MEM. PROTECT BITS
        DAC ACSAV       /SAVE THE AC
        DTEF            /SKIP ON ERROR FLAG
        SKP             /SKIP
        JMP DECER       /GO TO ERROR ROUTINE (NOT
                        /SHOWN)
        DTDF            /SKIP ON DECTAPE FLAG
        SKP             /SKIP
SWITCH  JMP SEARCH      /GO TO SEARCH ROUTINE
SEARCH  LAC (BLK        /GET THE CURRENT BLOCK NUMBER
        AND (007777     /MASK OUT ALL BUT THE LAST
                        /12 BITS
        SAD RBLK        /SKIP IF DIFFERENT FROM
                        /DESIRED BLOCK
        JMP RBLKS       /SAME BLOCK, GO TO READ DATA
                        /EXAMPLE
        TCA             /DIFFERENT BLOCK, CALCULATE
                        /THE DIFFERENCE BY SUBTRACTING
        DAC TEMP        /THE DESIRED BLOCK FROM THE
        LAC RBLK        /CURRENT BLOCK. IF THE RESULT
        TCA             /IS NEGATIVE, THEN THE TAPE
        ADD TEMP        /DRIVE IS MOVING TOWARD THE
        SMA             /DESIRED BLOCK. THE RESULT OF
                        /THIS CALCULATION IS THE TWO'S
                        /COMPLEMENT OF THE COUNT TO
                        /THE DESIRED BLOCK.
        JMP TEST        /GO TO TEST IF TAPE WAS REVERSED
```

```
GO      DAC 30          /MOTION OK SET UP WC
        LAC (010000     /GET STATUS A CHANGE
        DTXZ            /XOR CHANGE TO STATUS A
                        /(PUT INTO CONTINUOUS)
                        /THE XOR INSTRUCTION IS USED INSTEAD OF THE CLEAR
                        /AND LOAD TO AVOID SETTING THE SELECT DELAY, WHICH
                        /COULD PREVENT ANY ACTION FOR 120 MS
        DBR             /DEBREAK AND RESTORE
        JMP* DECTAP     /RETURN TO MAIN PROGRAM

TEST    ISZ TAG         /CHECK TO SEE IF THE TAG WAS SET
        JMP REV         /IT WAS NOT; GO AND REVERSE
                        /THE TAPE
        TCA             /IT WAS, COMPLEMENT THE AC
                        /TO GENERATE WC
        JMP GO          /GO BACK AND SET UP THE WORD
                        /COUNT

REV     DZM TEMP        /SET UP A DELAY WHICH WILL
        ISZ TEMP        /ALLOW THE TRANSPORT TO MOVE
        JMP .-1         /SEVERAL BLOCKS PAST THE CURRENT
                        /ONE.
        LAC (361400     /GET THE NEW STATUS A WORD
                        /WHICH REVERSES THE TAPE.
        DTLA            /LOAD THE NEW STATUS A WORD
        CLC             /CLEAR AND COMPLEMENT THE AC
        LAC TAG         /PUT RESULTS IN TAG TO REMEMBER
                        /THAT REV HAPPENED.
        JMP* DECTAP     /RETURN TO MAIN PROGRAM UNTIL
                        /THE NEXT NUMBER IS PICKED
                        /UP WHILE THE TAPE IS
                        /TRAVELLING IN THE REVERSE
                        /DIRECTION.

TAG     0
```

Note that it is necessary to remember that the tape had been reversed, and was travelling towards the desired block. In this way the previous subroutine could be used when the tape again found the current block while travelling in the reverse direction.

3.8.2 Read Data

```
RBLKS   LAC ADDR        /GET THE ADDRESS
        DAC 31          /PUT IT INTO CA REGISTER
        LAC WDCNT       /GET THE WORD COUNT
        TCA             /TWO'S COMPLEMENT IT
        DAC 30          /PUT IT INTO THE WC REGISTER
        LAC (003000     /GET THE UPDATED FUNCTION
        DTXA            /XOR IT INTO THE CONTROLLER
        LAC (JMP REDE   /GET THE EXIT POINT
        DAC SWITCH      /PUT IT INTO SWITCH
```

```
        LAC ACSAV       /GET THE AC AND RESTORE IT
        DBR             /DEBREAK AND RESTORE
        JMP* DECTAP     /RETURN TO MAIN PROGRAM
ADDR    ADDRESS
WDCNT   N
```

After the block has been found, the program begins to read the data in the block. The RBLKS subroutine sets up the word count and current address, and XOR's the new function. It also resets the interrupt chain, priming it to jump to REDE, the subroutine which is to handle the data once it was read (not given here).

3.8.3 Bootstrap Loading Technique

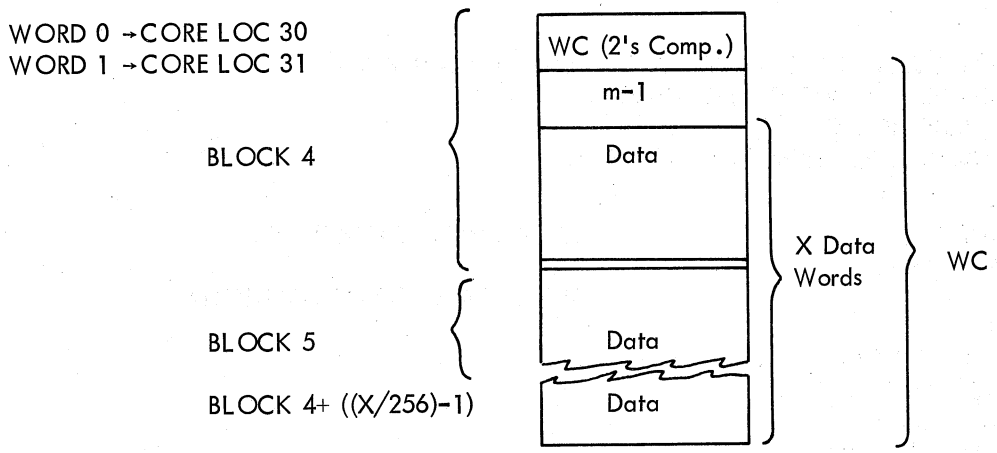
The data channel facility and the design of CONTINUOUS MODE allow for linked loading of data from DECtape, during which the first two DECtape data words determine the core location and amount of data which follows. The address into which data is loaded is specified by CA; thus, if the CA points to the WC-1, the first word transferred specifies the number of words to be transferred. After the first data word transfer, the CA points to itself, and the second word transferred specifies where the following data is to be loaded. No program interrupts, timing, or computation is required to locate the data. Only the TC15 and DCH features are used.

Problem:

Load x data words beginning at DECtape block 4 of unit 3 into core locations M to M+X-1, assuming tape has been positioned at block 4.

```
/BOOTSTRAP EXAMPLE
                .
                .
                .
        DZM*(30         /TO ENSURE NO WC OVERFLOW
        LAC (27         /TO BEGIN LOADING AT REGISTER 30
        DAC*(31         /CA
        LAC IOTD        /IOT DATA
        DTLA            /LOAD STATUS REGISTER
                .
                .
                .
IOTD,   332400          /READ ON UNIT #3, CM, FORWARD, GO (1)
                .       /WITH INTERRUPT ENABLED
                .
                .
```

The following represents the format of the data on the tape starting at block 4.



Where X is an integral multiple of 256.

3.8.4 Writing and Reading in Opposite Directions

As mentioned earlier, it is possible to read data from a DECtape in the opposite direction from which the data was written via program manipulation. However, a re-ordering of both the entire block and individual words is required.

a. Block Re-ordering: A block of words $x_1 \rightarrow x_n$ recorded in one direction is loaded into core as $x_n \rightarrow x_1$ when read in the opposite direction.

b. Word Unscrambling: Data read in backwards comes into core memory locations from the TC02 in the following 18-bit format:

Bits:

$\overline{15}\ \overline{16}\ \overline{17}\ \overline{12}\ \overline{13}\ \overline{14}\ \overline{9}\ \overline{10}\ \overline{11}\ \overline{6}\ \overline{7}\ \overline{8}\ \overline{3}\ \overline{4}\ \overline{5}\ \overline{0}\ \overline{1}\ \overline{2}$

In bit positions:

0 ◄——————————————————————► 17

NOTE

If data is to be re-ordered on the fly, the routine is limited to 140 μs since the word transfer rate = 200 μs (±30%). The probability of such a routine not working is very high if interrupts from other devices are encountered.

The following subroutines may be used for re-ordering the block of words and unscrambling each individual word.

```
/ROUTINE TO UNSCRAMBLE DECTAPE WORD ON REVERSE TRANSFER
/CALLING SEQUENCE:- LAC WORD; JMS UNSCR
/EXIT WITH NEW WORD IN AC
/37 LOCATIONS USED; EXECUTION TIME 47 μSECS.

UNSCR   XX
        CMA                     /COMPLEMENT 2 HOLD WORD
        DAC   UST1
        RCL                     /SWITCH PAIRS OF OCTAL DIGITS
        RTL
        AND   (707070
        DAC   UST2
        LAC   UST1
        AND   (707070
        RCR
        RTR
        XOR   UST2
        DAC   UST1
        RAR                     /MOVE LAST PAIR
        RTR
        RTR
        RTR
        AND   (770000
        DAC   UST2
        LAC   UST1              /MOVE FIRST PAIR
        AND   (770000
        RCL
        RTL
        RTL
        RTL
        XOR   UST2              /COMBINE ENDS
        DAC   UST2
        LAC   UST1              /...2 ADD TO MIDDLE
        AND   (7700
        XOR   UST2
        JMP*  UNSCR
UST1    XX                      /COULD USE OTHER TEMP. STORAGE
UST2    XX
```

3.9 PROGRAMMING NOTES

3.9.1 Modification of Individual Data Words

The only way to modify individual data words within a block is to:

1. Read block in to core buffer
2. Modify Respective Core locations
3. Write core buffer back into original block on tape

3.9.2 Data Transfer - Upper Boundary Protection

The WC controls all data transfers. After WC overflow, no more data transfers take place. Thus, to protect memory when reading a block of unknown length, the WC is set to the 2's complement of the difference between the initial address where data is transferred and the upper boundary.

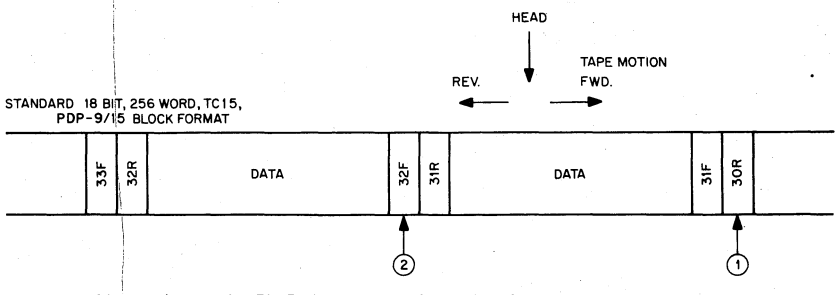
Similar action prevents writing beyond a predetermined point on tape when transferring an unknown number of words from core.

3.9.3 Special Formats on Tape

The user is cautioned always to specify an even number of words in his special format. If he does not, the control will indicate parity errors where none exist.

3.9.4 Turnaround Commands

Programming Note: When a turnaround command is issued (i.e., complement the direction bit while the GO bit remains set to 1), the tape may not be up to speed when the point at which the command was issued is passed (in the new direction). The tape will be up to speed one standard 256-word block length after the turnaround point. Therefore, to find a block in the opposite direction, it is sufficient to delay the turn around one block as shown in the following example:



NOTES: 1.) CONSIDER HEAD FIXED WITH TAPE MOVING PAST IT
2.) 31-R-R REVERSE BLOCK # (ie RECOGNIZED ONLY IN REV.)
3.) 31-F-F FORWARD BLOCK# (ie RECOGNIZED ONLY IN FWD.)

TO FIND BLOCK 32 FORWARD:
1.) SEARCH REVERSE TO BLOCK 30
2.) TURN AROUND AND SEARCH FORWARD FOR BLOCK 32
3.) BLOCK 31 MAY BE FOUND BUT BLOCK 32 IS GUARANTEED TO BE FOUND.

09-0106

With this turn-around specification, finding blocks next to the end zones requires special handling. Block 0 forward may be found if the tape is backed into the end zone twice before turning around. To prevent this special end zone handling, a new formatting program must be written which provides one block length of inter-block zone marks (no-op marks) so that the program can bounce off the end zone and find block 0 (if the tape has the new format on it). The end zone problem is also solved for either format by not using the block next to the end zones (block 0, 1101).

When using non-standard format tape (i.e., not 1102_8 blocks of 400_8 words), a length of tape equal to one 18 bit, 256_{10} (400_8) word block must pass the head before the turn-around command is issued. This length is approximately five in. of tape; however, when calculating the required delay for a non-standard format tape, it should be computed in equivalent standard block lengths.

Example: Turn-around delay calculation for blocks of 94_{10} words.

$$\frac{256 \text{ words/block} \times 1 \text{ block delay}}{92 \text{ words block}} = 2.7 \text{ block delay required}$$

3.10 DECTAPE SUMMARY

3.10.1 DECtape Function Summary

Function	Normal Mode (NM)	Continuous Mode (CM)
0. Move	DTF: No Interrupt CA: Ignored WC: Ignored	Same as NM
1. Search	DTF: Interrupt at each block mark CA: Not incremented WC: Incremented at each block mark	DTF: Interrupt at block mark where WC overflows CA: Not incremented WC: Incremented at each block mark
2. Read Data	DTF: Interrupt at end of each block CA: Incremented at each word transfer WC: Incremented at each word transfer	DTF: Interrupt at WC overflow and end of block CA: Incremented at each word transfer WC: Incremented at each word transfer
3. Read All	DTF: Interrupt at each word transfer CA: Incremented at each word transfer WC: Incremented at each word transfer	DTF: Interrupt at WC overflow CA: Incremented at each word transfer WC: Incremented at each word transfer

Function	Normal Mode (NM)	Continuous Mode (CM)
4. Write Data	Same as 2. Read Data	Same as 2. Read Data
5. Write All	Same as 3. Read All	Same as 3. Read All
6. Write Timing & Mark Tracks	Same as 3. Read All	Same as 3. Read All
7. Unused*	- - - - - - -	
*If used by mistake, the control gives a Select Error (SE).		

3.10.2 DECtape Error Summary

Function	Normal Mode or Continuous Mode
Move	Select Error EOT
Search	Select Error EOT Timing Error MK TRK Error
Read Data	Select Error EOT Timing Error Parity Error MK TRK Error
Read All	Select Error EOT Timing Error MK TRK Error
Write Data	Select Error EOT Timing Error MK TRK Error
Write All	Select Error EOT Timing Error MK TRK Error
Write Timing & Mark Tracks	Select Error Timing Error

3.10.3 DECtape Timing Data on Standard Format (Certified) Tape

Operation	Time
Time to answer data channel request	Up to 200 μs*,
Word Transfer Rate	1 18-bit word every 200 μs*
Block Transfer Rate	1 256 word block every 53 ms*
Start Time	375 ms (±20%)
Stop Time	375 ms (±20%)
Turn Around Time (see Paragraph 3.9.4)	375 ms (±20%)
Search → Read Data Function change for present block	Up to 400 μs*
Search → Write Data Function change for present block	Up to 400 μs*
Read → Search Function change for next block number	Up to 1000 μs*
Write → Search Function change for next block number	Up to 1000 μs*
DTF to beginning of next data block	1.7 ms*
DTF Occurrence:	
Move: NM, CM	Never
Search: NM Read Data: NM Write Data: NM	Every 53 ms*
Search: CM	(WC) X53 ms*
Read Data: CM Write Data: CM	(# blocks) X53 ms*
Read All: NM Write All: NM Write Timing & Mark Tracks: NM	Every 200 μs*
Read All: CM Write All: CM Write Timing & Mark Tracks: CM	(WC) X200 μs*

*(±30%)

Chapter 4
Teletype Controls

4.1 INTRODUCTION

There are three Teletype controls available in a PDP-15 System:

a. An internal Teletype control which supports the console Teletype and is provided as part of the PDP-15 Processor.

b. The LT15A Single-Teletype Control, which supports one additional Teletype and is typically used on PDP-15/30 and PDP-15/40 Background/Foreground Systems. The LT15A Control plugs into the BA15 Option Panel, which also houses the paper tape reader/punch control and the VP15 Display Control.

c. An LT19 Multi-Station Teletype Control which handles from one to five Teletypes. The LT19 is more flexible than the other two in that it will also drive EIA compatible devices. This control requires that a DW15A Positive to Negative Bus Converter be on the PDP-15 System to which it interfaces, because it interfaces to the negative logic bus.

These three Teletype controls support DEC-modified Teletype Models 33 or 35 KSR under IOT control. The internal control and the LT19 Multi-Station Teletype Control also operate with Teletype Models 33 and 35 ASR. Each control operates in full-duplex with an 8-bit code which has one-unit start and two-unit stop codes.

NOTE

The console terminal normally operates as full-duplex with local copy.*

Code variations tolerated by the LT19 Control are described in Paragraph 4.3.

4.2 LT15 SINGLE-TELETYPE CONTROL

The LT15 Single-Teletype Control consists of two functional sections, the transmitter and the receiver.

4.2.1 Transmitter

The transmitter accepts the 8-bit parallel code from the computer I/O bus, converts it to serial form, and sends it to the Teletype printer.

Each time the transmitter has finished serializing the data, it raises its flag and forces an interrupt or either API channel 3 at trap address 74, or on the program interrupt.

---

*A term often confused with and equated to "half-duplex".

4.2.2 Receiver

The receiver accepts the serial code from the Teletype keyboard, converts it to a parallel code, and makes it available to the I/O bus to be fetched under IOT command by the CPU.

Each time the receiver has finished converting the serial data to parallel code, it also raises a flag that requests an interrupt on API channel 3 trap address 75 or a program interrupt.

4.2.3 Instruction Set

The instruction set summarized in Table 4-1 is identical to that for each of the controllers in the LT19. For programming examples, refer to the programming section of the LT19.

Table 4-1
LT15 IOT Instructions

Mnemonic	Octal Code	Description
Transmitter IOTs		
TSF1	704001	Skip on transmitter (Teleprinter) flag.
TCF1	704002	Clear transmitter flag.
TLS1	704004	Load transmitter buffer and transmit.
Receiver IOTs		
KSF1	704101	Skip on receiver flag.
KRB1	704102	Clear receiver flag and read buffer.

4.3 LT19D MULTI-STATION TELETYPE CONTROL

The LT19D Multi-Station Teletype Control interfaces up to five Model 33 or 35 Teletypes or signal-compatible EIA devices to the PDP-15 Computer.

The LT19 connects to the PDP-15 I/O bus through a DW15A Positive to Negative Bus Converter. Operation is through the CPU under IOT control. Each LT19 consists of four subsystems as described below.

4.3.1 LT19D Multiplexer

The LT19D contains multiplexing logic for a total of 5 Teletype controls. The LT19D is also the logic shell into which the other options plug.

4.3.2 LT19E Teletype Control

The LT19E Teletype Control operates with the following characteristics:

a. It uses five- or eight-bit character codes. Eight is standard.
b. It uses a one-unit start code.
c. It uses 1-, 1.5-, or 2-unit stop codes. Two is standard.
d. It operates in full duplex.
e. Transmission and reception speeds are variable, (screwdriver adjustment) to 30,000 baud.
f. Each LT19E will control ASR, KSR RO or SO Teletype units as supplied by DEC.

Up to five LT19E Teletype Controls can be handled by a single LT19D.

4.3.3 LT19F EIA Line Adapter

The LT19F is a group of modules which plugs into the LT19D and converts negative logic LT19E levels to EIA levels.

Each LT19E can support one LT19F. The LT19F supplies EIA logic levels which are compatible with certain types of Dataphones(TM), such as the Bell 103A. Although the LT19E, F combination supplies the necessary data signals to the Dataphone(TM), it does not supply control signals.

4.3.4 LT19H Cable Set

The LT19H is a cable and a set of instructions for interconnecting an LT19D, E, F combination to another LT19D, E, F, or an equivalent PT08. The LT19H takes advantage of the LT19F Bus Drivers and special terminating techniques to provide for a low cost interprocessor data link between PDP-9/15 and PDP-8 computers. The cable comes in five lengths:

LT19HA — 50 ft
LT19HB — 100 ft
LT19HC — 150 ft
LT19HD — 200 ft
LT19HE — 250 ft

4.4 THE OPERATION OF THE LT19 MULTI-STATION TELETYPE CONTROL

4.4.1 LT19D Multiplexer

Figure 4-1 illustrates the structure of the LT19D Multiplexer. The multiplexer supports up to five LT19E, F, and H options, which plug into their appropriate slots. The LT19D supplies API, program

(TM)Dataphone is a registered trademark of Bell Systems.

interrupt (PI), and skip facilities for each of the Teletype controllers. All transmit flags are ORed together and will cause interrupts on either the PI or API facility. The API traps to location 74 on level 3. Each receive flag will also cause a PI or an API interrupt. The API traps to location 75 on level 3. Skip requests are all ORed together to the skip line.

4.4.2 LT19E Teletype Control

Figure 4-2 illustrates the structure of each LT19E Teletype Control. Each LT19E is capable of servicing an ASR, KSR 33 or 35 Teletype, or an equivalent peripheral. Operation is full-duplex. Each LT19E decodes its own transmit or receive IOT and accepts or deposits parallel data onto the I/O bus for transfer to or from the AC. When data is transmitted to the Teletype, the parallel data is strobed into the transmitter, converted into the appropriate serial code, and passed to the printer through its cable. When the controller is receiving data from the Teletype, the serial incoming signal is converted to an appropriate parallel code and presented to the I/O bus lines for transfer under IOT to the AC.

Whenever a word is ready to be transferred into the AC from the receive logic, the appropriate flag is raised and an API or PI interrupt is requested. The program must identify the requesting device in a skip chain.

Similarly, as soon as a word has been transmitted, the transmitter raises a flag to request another word. This flag, when it causes an API or PI request, must also be identified by the program through a skip chain.

4.4.3 The LT19F EIA Line Adapter

The addition of the LT19F option to an LT19E simply adds level converters and a driver to the output of the LT19E. Another cable slot is available to take advantage of the EIA levels and increased drive. These levels are compatible with those accepted by such data sets as the 103A; however, the 103A usually requires additional control logic to be controlled by the computer. This logic must either be supplied by another device or its need removed by special wiring. The appropriate dataphone manual should be consulted.

4.4.4 The LT19H Cable Set

The LT19 System can be used as a low cost interprocessor communications link by interconnecting two LT19D, E, F, or equivalent (PT08) Controllers together with the LT19H. The H option is simply a cable and instructions on how to use it. The cable plugs into the output of an LT19F. The maximum baud rate that such a system can operate is dependent on the entire system. DEC recommends that the total rate handled by a complete LT19 System be no more than 30,000 baud.

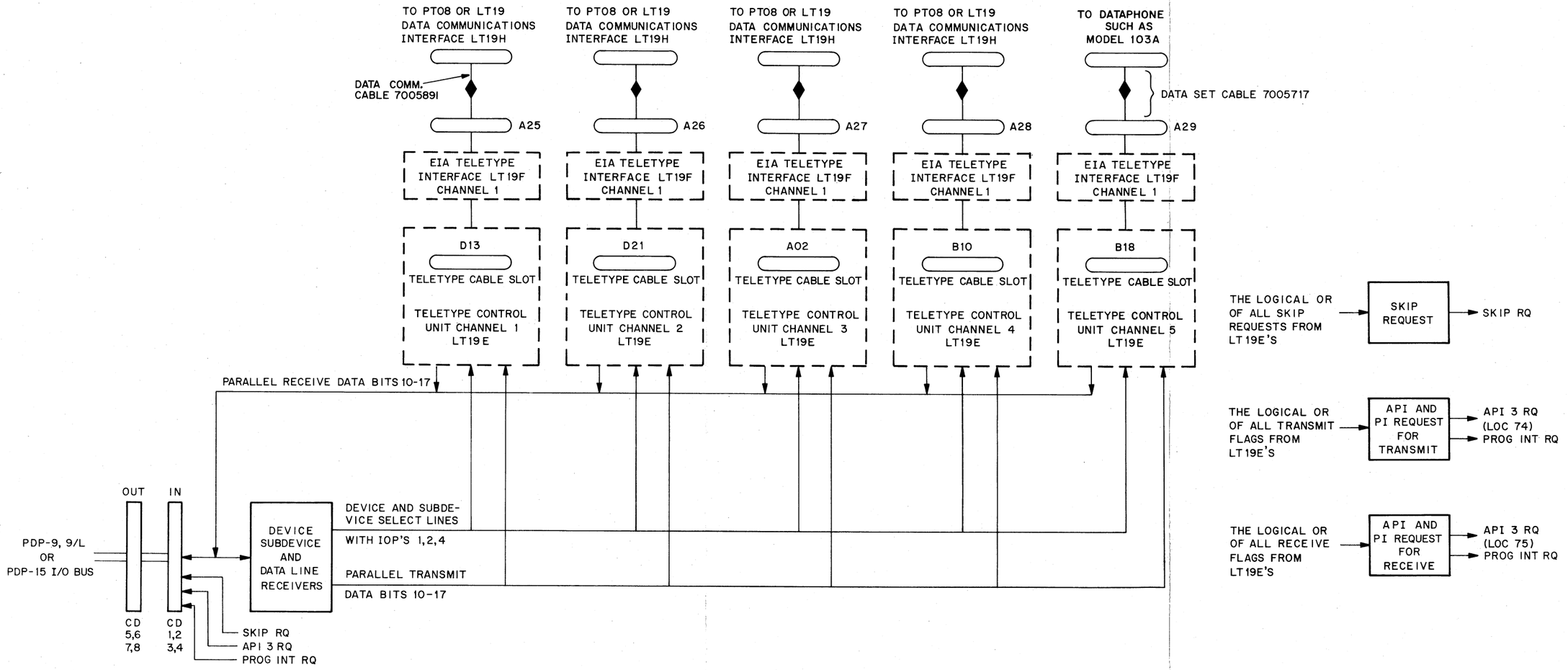


Figure 4-1 LT19E Multi Station Teletype Control Block Diagram

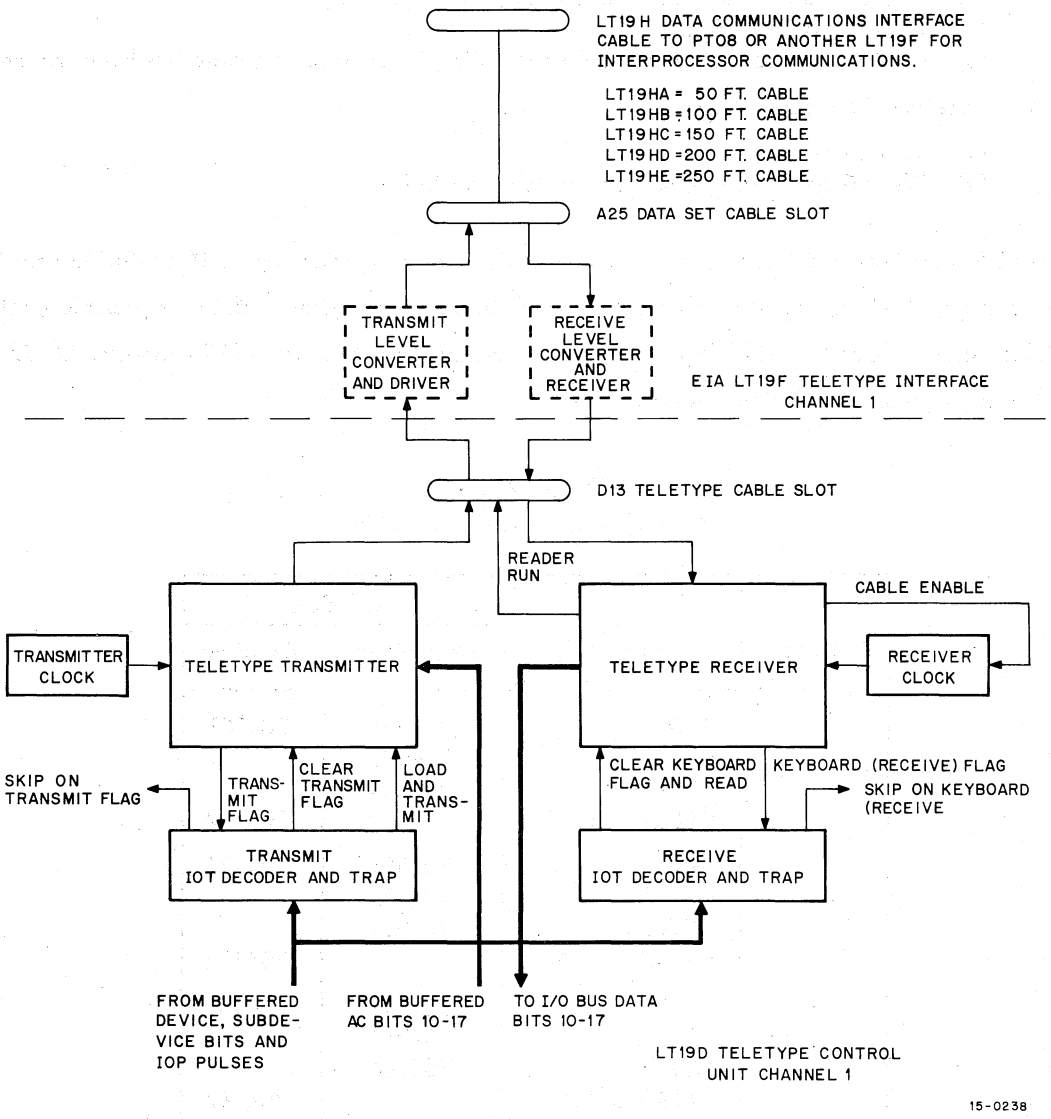


Figure 4-2 LT19E, F, H Teletype Control Interface & Communications

4.5 THE INSTRUCTION SET

Each transmitter and receiver of each LT19E in the system has a unique set of instructions listed below:

a. Transmitter IOTs

(1) Skip on transmitter flag (also called Teleprinter flag).
(2) Clear transmitter flag.
(3) Load transmitter buffer and transmit. Flag is set when the word has been transmitted.

b. Receiver (keyboard) IOTs

(1) Skip on receiver flag.
(2) Clear receiver flag and read the receiver buffer.

The PDP-15 System Software has allocated enough IOTs to accommodate up to 16 LT19Es in four LT19Ds. Tables 4-2 through 4-5 list the IOTs that are assigned to each controller in the four possible configurations. Note that if there is an LT15A already on the system when the first LT19 is added, LT19E Unit #1 is not used because the LT15 has used its IOTs.

Table 4-2
IOT Assignments for One LT19

Unit No.	Function	IOT Code	
		Transmitter	Receiver
1	SKIP	704001	704101
	CLEAR	704002	
	LOAD	704004	
	CLEAR & READ		704102
2	SKIP	704021	704121
	CLEAR	704022	
	LOAD	704024	
	CLEAR & READ		704122
3	SKIP	704041	704141
	CLEAR	704042	
	LOAD	704044	
	CLEAR & READ		704142
4	SKIP	704061	704161
	CLEAR	704062	
	LOAD	704064	
	CLEAR & READ		704162
5	SKIP	704201	704301
	CLEAR	704202	
	LOAD	704204	
	CLEAR & READ		704302

Table 4-3
IOT Assignment for Two LT19s

Unit No.	Function	LT19#1 IOT Code		LT19#2 IOT Code	
		Transmitter	Receiver	Transmitter	Receiver
1	SKIP	704001	704101	704201	704301
	CLEAR	704002		704202	
	LOAD	704004		704204	
	CLEAR & READ		704102		704302
2	SKIP	704021	704121	704221	704321
	CLEAR	704022		704222	
	LOAD	704024		704224	
	CLEAR & READ		704122		704322
3	SKIP	704041	704141	704241	704341
	CLEAR	704042		704242	
	LOAD	704044		704244	
	CLEAR & READ		704142		704342
4	SKIP	704061	704161	704261	704361
	CLEAR	704062		704262	
	LOAD	704064		704264	
	CLEAR & READ		704162		704362
5	SKIP	704401	704501	704421	704521
	CLEAR	704402		704422	
	LOAD	704404		704424	
	CLEAR & READ		704502		704522

Table 4-4
IOT Assignments for Three LT19s

Unit No.	Function	LT19#1 IOT Code		LT19#2 IOT Code		LT19#3 IOT Code	
		Transmitter	Receiver	Transmitter	Receiver	Transmitter	Receiver
1	SKIP	704001	704101	704201	704301	704401	704501
	CLEAR	704002		704202		704402	
	LOAD	704004		704204		704404	
	CLEAR & READ		704102		704302		704502
2	SKIP	704021	704121	704221	704321	704421	704521
	CLEAR	704022		704222		704422	
	LOAD	704024		704224		704424	
	CLEAR & READ		704122		704322		704522
3	SKIP	704041	704141	704241	704341	704441	704541
	CLEAR	704042		704242		704442	
	LOAD	704044		704244		704444	
	CLEAR & READ		704142		704342		704542

Table 4-4 (Cont)
IOT Assignments for Three LT19s

Unit No.	Function	LT19#1 IOT Code		LT19#2 IOT Code		LT19#3 IOT Code	
		Transmitter	Receiver	Transmitter	Receiver	Transmitter	Receiver
4	SKIP	704061	704161	704261	704361	704461	704561
	CLEAR	704062		704262		704462	
	LOAD	704064		704264		704464	
	CLEAR & READ		704162		704362		704562
5	SKIP	704601	704701	704621	704721	704641	704741
	CLEAR	704602		704622		704642	
	LOAD	704604		704624		704644	
	CLEAR & READ		704702		704722		704742

Table 4-5
IOT Assignments for Four LT19s

Unit No.	Function	LT19#1 IOT Code		LT19#2 IOT Code		LT19#3 IOT Code		LT19#4 IOT Code	
		Transmitter	Receiver	Transmitter	Receiver	Transmitter	Receiver	Transmitter	Receiver
1	SKIP	704001	704101	704201	704301	704401	704501	7040_	7040_
	CLEAR	704002		704202		704402		7040_	
	LOAD	704004		704204		704404		7040_	
	CLEAR & READ		704102		704302		704502		7040_
2	SKIP	704021	704121	704221	704321	704421	704521	7040_	7040_
	CLEAR	704022		704222		704422		7040_	
	LOAD	704024		704224		704424		7040_	
	CLEAR & READ		704122		704322		704522		7040_
3	SKIP	704041	704141	704241	704341	704441	704541	7040_	7040_
	CLEAR	704042		704242		704442		7040_	
	LOAD	704044		704244		704444		7040_	
	CLEAR & READ		704142		704342		704542		7040_
4	SKIP	704061	704161	704261	704361	704461	704561	7040_	7040_
	CLEAR	704062		704262		704462		7040_	
	LOAD	704064		704264		704464		7040_	
	CLEAR & READ		704162		704362		704562		7040_
5	SKIP	704601	704701	704621	704721	704641	704741	704661	704761
	CLEAR	704602		704622		704642		704662	
	LOAD	704604		704624		704644		704664	
	CLEAR & READ		704702		704722		704742		704762

4.5.1 Programming Examples

The following program assumes that the system has one LT19D and two LT19Es. It also assumes that API is available.

```
PI          0                   /STORE LINK, PAGE/BANK MODE, PC &
            JMP FLGS            /MEM. PROT. BITS.

            IOT SKPA            /SKIP IF DEVICE A FLAG
            SKP                 /GO TO NEXT DEVICE
            JMS DEVA            /HANDLE DEVICE A
                .
                .
                .
                .
            IOT SKPT            /SKIP IF LT19 TRANSMITTER FLAG
            SKP                 /GO TO NEXT DEVICE
            JMS MULTT           /HANDLE TRANSMITTER FLAG
            IOT SKPR            /SKIP IF LT19 RECEIVER FLAG
            SKP                 /GO TO NEXT DEVICE
            JMS MULTR           /HANDLE RECEIVER FLAG
                .
                .
                .
            ION                 /TURN PIC ON
            RES                 /RESTORE
            JMP* PI             /RETURN TO MAIN PROGRAM

74          JMS MULTR           /RECEIVER API SERVICE

75          JMS MULTT           /TRANSMITTER API SERVICE

MULTR       0                   /STORE LINK, PAGE/BANK MODE, MEM. PROT. & PC.
            DAC RAC             /SAVE AC
            IOT SKPR1           /SKIP IF RECEIVER 1 FLAG
            SKP                 /GO TO NEXT DEVICE
            JMS TTR1            /HANDLE RECEIVER 1 FLAG
            IOT SKPR2           /SKIP IF RECEIVER 2 FLAG
            SKP                 /GO TO NEXT DEVICE
            JMS TTR2            /HANDLE RECEIVER 2 FLAG
            LAC RAC             /RESTORE AC
            DBR                 /DEBREAK AND RESTORE
            JMP* MULTR          /RETURN TO MAIN SKIP CHAIN

TTR1        0                   /STORE LINK, PC, PAGE/BANK MODE & MEM.
                                /PROT. BITS.
            IOT CLRR1           /CLEAR FLAG AND READ RECEIVER 1
            DAC* PTR1           /PUT CHARACTER INTO FILE
            ISZ PTR1            /INCREMENT POINTER
            JMP* TTR1           /RETURN TO LT19 SKIP CHAIN

TTR2        0                   /STORE LINK, PC, PAGE/BANK MODE & MEM.
                                /PROT. BITS.
            IOT CLRR2           /CLEAR FLAG AND READ RECEIVER 2
            DAC* PTR2           /PUT CHARACTER IN FILE
            ISZ PTR2            /INCREMENT FILE POINTER
            JMP* TTR2           /RETURN TO LT19 SKIP CHAIN
```

After determining that the interrupting flag belonged to an LT19 receiver, the program then identifies the particular receiver. This operation must be done with another skip chain, which then branches the program to the appropriate subroutine to fetch and store the character in a file referred to by the pointer PTR1 or PTR2. The pointer is incremented for the next word. In a more comprehensive program, each file would also need a counter which would overflow when the file filled. The program would then dump the file on tape, disk, or a similar device, or pack the words and then dump them.

```
MULTT   0                   /STORE LINK, PC, PAGE/BANK MODE & MEM.
                            /PROT. BITS
        DAC TAC             /SAVE AC
        IOT SKPT1           /SKIP IF TRANSMITTER 1 FLAG
        SKP                 /GO TO NEXT DEVICE
        JMS TTT1            /HANDLE TRANSMITTER 1 FLAG
        IOT SKPT 2          /SKIP IF TRANSMITTER 2 FLAG
        JMS TTT2            /HANDLE TRANSMITTER 2 FLAG
        LAC TAC             /RESTORE AC
        DBR                 /DEBREAK AND RESTORE
        JMP* MULTT          /RETURN TO MAIN SKIP CHAIN
TTT1    0                   /STORE LINK, PC, PAGE/BANK MODE & MEM.
                            /PROT. BITS.
        LAC* PTT1           /GET TRANSMITTER 1 CHARACTER
        ISZ PTT1            /INCREMENT PLOTTER
        IOT CLDT1           /DEPOSIT WORD IN TRANSMITTER AND
                            /CLEAR TRANSMITTER FLAG
        JMP* TTT1           /RETURN TO LT19 SKIP CHAIN
TTT2    0                   /STORE LINK, PC, PAGE/BANK MODE & MEM.
                            /PROT. BITS.
        LAC* PTT2           /GET TRANSMITTER 2 CHARACTER
        ISZ PTT2            /INCREMENT POINTER
        IOT CLDT2           /DEPOSIT WORD IN TRANSMITTER 2
                            /AND CLEAR TRANSMITTER FLAG
        JMP* TTT2           /RETURN TO LT19 SKIP CHAIN
```

This routine parallels the previous coding for the receiver flag. Only the pointers and IOTs have been changed to protect the files.

Chapter 5
Line Printers

5.1 INTRODUCTION

The line printers provide the PDP-15 with a selection of hard-copy output devices. The characteristics of the five options are listed in Table 5-1.

Table 5-1
Line Printer Characteristics

Option	Lines/Min	Characters/Line	Number of Printing Characters
LP15C	1000	132	64
LP15F	356	80	64
LP15H	253	80	96
LP15J	245	132	64
LP15K	173	132	96

All of these printers may use the same systems programs, allowing for the differences in line lengths and that the 64 character printers convert all lower case character codes to upper case before printing. The diagnostic program for the LP15C differs from the others because of differences in the printer and its control. The LP15C differs from the other printers in the way it handles vertical format characters (see Paragraphs 5.6 and 5.7). For most applications, however, these differences do not affect the user.

Once started by an LPP1 or LPPM command, the line printers use the 3-cycle data channel facility of the PDP-15 to access a character buffer in core. This buffer contains up to 256 lines of characters, each line terminated by a line feed or other control character. In this way, up to 256 lines may be printed without further attention from the program.

5.2 CHANNEL AND BUFFER SETUP

The line printer is assigned to data channel locations 34 and 35. The word count location, 34, is not used and can be ignored. The current address location, 35, should be initialized to the address immediately preceding the start of the data buffer. Location 35 will be modified as printing progresses.

The data buffer area must begin with a 2-word header in the format shown in Figure 5-1.

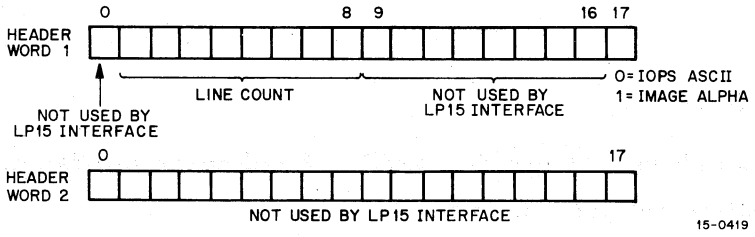


Figure 5-1 Data Buffer Header Format

Bit 0 and bits 9-16 of header word 1 are not used by the hardware. Bit 0 indicates the line printer mode of operation to the program. It is set in multi-line mode and cleared in single-line mode. Bits 9-16 contain flags and parity bits used by the Advanced Monitor System software. The line count is used in multi-line mode (see Paragraph 5.5), and bit 17 of the first header word selects the format of the data words to follow.

5.3 DATA WORD FORMATS

Following the header words are words containing the characters to be printed in one of the two formats as indicated by the header.

5.3.1 IOPS ASCII

In the IOPS ASCII format, five 7-bit character codes are contained in two consecutive words, as shown in Figure 5-2.

In IMAGE ALPHA format, a single 7-bit character code, right justified, is located in each 18-bit word, as shown in Figure 5-3.

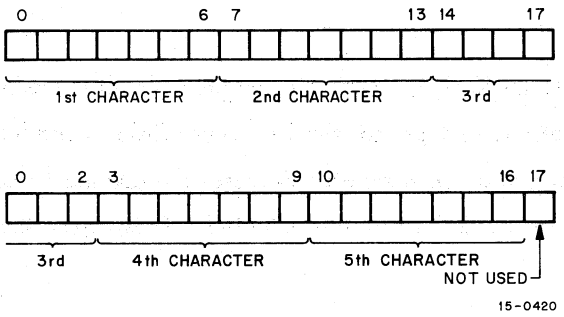


Figure 5-2 5/7 ASCII Packing Scheme

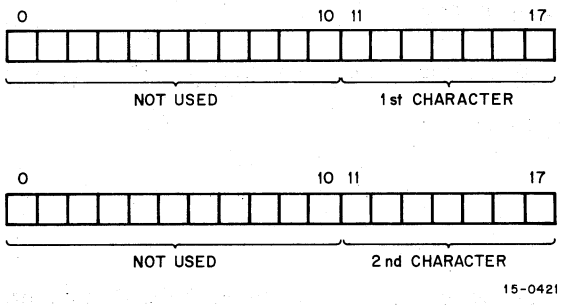


Figure 5-3 IMAGE ALPHA Format

5.4 SINGLE LINE OPERATION

On receiving the LPP1 IOT (706541_8), the printer prints a single line of text. First, two consecutive data channel requests are used to obtain the header words. The line count is ignored, but the data format selection bit is saved. The data channel is again used to bring in the first two data words (for efficiency, even IMAGE ALPHA mode words are obtained in pairs). According to the selection format, the characters are unpacked and sent one-by-one to the printer.

When the data buffer is exhausted, another pair is brought in and the process continues.

As each character is unpacked, it is checked for being one of the vertical control characters (such as Line Feed) listed in Paragraphs 5.6 and 5.7. When one is found, it terminates the current line, causes the appropriate control action (advancing the paper), and sets the Done flag. If more characters are received than the line can hold (80 or 132), with no control characters, the Line Overflow flag is set, indicating the error (see Paragraph 5.8.3).

5.5 MULTI-LINE OPERATION

On receiving the LPPM IOT (706521_8), the printer enters multi-line mode. This mode of operation is similar to single-line operation, except that the line count field of the header word is stored in the line counter. Each control character encountered causes the counter to decrement, and only when it reaches 0 is the Done flag raised and printing terminated. Thus, up to 256 lines may be printed from one LPPM.

Note that only lines of print are counted. Every control character (except horizontal tab) counts as one line, even though it may advance the form from an entire page to not at all.

5.6 LP15C CONTROL CHARACTERS

The LP15C recognizes HT, CR, LF, FF, DLE, DC1 through DC4, and ALT MODE as control characters. Their functions are as follows:

5.6.1 Horizontal Tab (HT)

The HT control character does not end a line or decrement the line counter. Permanent tab stops are located every eight columns, starting with column 9. Receiving an HT causes the controller to generate spaces until the next stop is reached. At least one space will be generated. Thus, if the sequence A HT B is received, and the A is printed in column 7, the B will be printed in column 9. If the A appears in column 8, however, the B cannot be in column 9, and is put instead in column 17. If more than 128 characters have been sent to the printer when the tab is received, there are no more stops on the line and the ILL HT flag is raised to signal the error (see Paragraph 5.8).

5.6.2 ALT MODE and Carriage Return (CR)

ALT MODE and CR have identical effects; they terminate the current line, start another at the left margin, but do not advance the paper. This allows for overstriking. The line counter is decremented.

5.6.3 Vertical Format Unit (VFU) Characters

All the remaining control characters use the VFU located in the printer to govern form advance. This unit contains a punched paper tape loop, synchronized with the paper feed. Each of the various control characters selects one of the eight channels across the paper tape, and the form is advanced until a hole is sensed in that channel. Line Feed, for instance, uses channel 8 which normally has a hole in every line except near the paper perforation which should be skipped over. See Table 5-2 for other assignments. All of these characters decrement the line counter by 1.

Table 5-2
Control Character Assignments

ASCII	Character	VFU Channel	Conventional Meaning*
012	LF Line Feed	8	1 line
013	VT Vertical Tab	7	1/3 page
014	FF Form Feed	1	Top of next page
020	DLE Device Control	2	1/2 page
021	DC1 Device Control	3	2 lines
022	DC2 Device Control	4	3 lines
023	DC3 Device Control	5	1 line
024	DC4 Device Control	6	1/6 page

*Using "normal" VFU tape, other spacing may be obtained by using specially prepared VFU tape.

5.7 CONTROL CHARACTERS FOR LP15, F, H, J, AND K

Because these printers lack a mechanical VFU, some of the control characters must be handled in a different manner. HT, CR, and ALT MODE have identical effects to those described in Paragraph 5.5. Form Feed causes the paper to advance to the top of the next page. The rest of the control characters cause the form to advance a fixed number of lines:

Character	Number of Lines
LF Line Feed	1
DLE Device Control	30
DC1 Device Control	2
DC2 Device Control	3
DC3 Device Control	1
DC4 Device Control	10
VT Vertical Tab	20

5.8 IOT INSTRUCTIONS AND FLAGS

Table 5-3 describes the IOTs used in normal printer operations. Other IOTs, useful only during maintenance, are described in the LP15C and LP15F Maintenance Manuals.

5.8.1 Error Flag

This flag is raised by an inclusive OR of the LP Alarm, Line Overflow, Illegal HT, and Interlock flags. When the Error flag is raised, the printer prints out all characters received before the error, if possible, and the Done flag is raised.

Table 5-3
LP15 IOT Instructions

Mnemonic	Octal Code	Description
LPP1	706541	The line printer prints a single line of text, as described in Paragraph 5.4.
LPPM	706521	The line printer enters multi-line mode, as described in Paragraph 5.5.
LPSF	706501	Skip if done or error. Skips the following instruction if the printer's Done or Error flag is set.
LPEI	706544	Enable interrupt system. Connects the printer to the PDP-15 priority interrupt system. Either the Done or Error flag will cause an interrupt if enabled. If the API is in use, the interrupt will be on level 3, channel 56.
LPCD	706621	Clear Done flag.
LPCF	706641	Clear status register and Error flag.
LPRS	706642	Read status register. This IOT reads into the AC a register made up of the following system flags: Bit — Flag 0 — Error 1 — LP Alarm 2 — Line Overflow 3 — Illegal HT 4 — Busy 5 — Done 6 — Interlock

The following paragraphs describe each of the status register flags.

5.8.2 LP Alarm Flag

This flag indicates some error condition in the printer itself. This may be due to printer power off, insufficient paper supply, printer off line, printer yoke open, or some electrical or mechanical malfunction.

5.8.3 Line Overflow Flag

This flag is set when more than 132 (or 80) characters are sent in a single line (i.e., without vertical control characters). A software error is indicated.

5.8.4 Illegal Horizontal Tab (ILL HT)

This flag indicates that an HT has been sent after the last tab stop on the line has been passed. This, too, is a software error.

5.8.5 Busy Flag

This flag is set at the start of a print operation and is cleared by raising the Done flag.

5.8.6 Done Flag

This flag is set by encountering the first control character in single-line mode, the final control character in multi-line mode, or completing printing following an error. It is cleared by IOT 6621 (LPCD).

5.8.7 Interlock Flag

This flag is set if there is no printer connected to the controller. Check the printer cable connections.

5.9 PROGRAMMING EXAMPLE

This programming example does not use the interrupt system, and assumes that interrupts are disabled. At START, the Form Feed contained in IBUFF is printer. This ensures a fresh form, and causes the Done flag to raise upon completion. PRINTM may now be called any number of times; its argument is a pointer to the buffer containing the characters to be printed. After calling PRINTM, the program may continue without waiting for the printer to complete its operation. PRINTM waits, if necessary, for the last task to finish, checks for errors, then starts the new print operation.

```
                .LOC 35              /INITIALIZE DATA CHANNEL ADDRESS
                IBUFF-1
                .LOC 200
START           LPP1                 /PRINT FORM FEED FROM IBUFF
                .
                .
                .
                .
                JMS PRINTM           /CALL TO MULTIPLE PRINT ROUTINE
                CBUFF-1              /POINTER TO CHARACTER BUFFER
                .
                .
                .
                HLT
```

```
IBUFF       000001      /HEADER- IMAGE ALPHA FORMAT
            000000      /HEADER
            000014      /CODE FOR FORM FEED
CBUFF       033000      /HEADER, 33 LINES, IOPS ASCII FORMAT
            000000      /HEADER
                .
                .
                .
                .
                .
                .
.
.
.
.                       /DATA WORDS--33 PRINT LINES
.
.
PRINTM      0           /MULTIPLE LINE PRINT ROUTINE
            LPSF        /CHECK IF PRINTER DONE
            JMP .-1     /NO, WAIT
            LPRS        /DONE, CHECK FOR ERROR (BIT 0)
            SPA
            JMS ERR     /TYPE OUT ERROR MESSAGE
            LAC* PRINTM /NO ERROR, GET BUFFER POINTER
            DAC* (35    /LEAD DATA CHANNEL ADDRESS
            LPPM        /START PRINTER
            ISZ PRINTM  /RETURN TO CALL +2
            JMP* PRINTM
```

Digital Equipment Corporation
Maynard, Massachusetts

digital

printed in U.S.A.