

PDP-7 PROGRAM LIBRARY

1. IDENTIFICATION

- 1.1 Digital-7-20-IO-FB PDP-7 DECTOG
1.2 Leonard M. Hantman
1.3 2-2-65



2. ABSTRACT

2.1 Purpose

To allow the user to perform various functions using DECtape and the toggle switches of a PDP-7 computer. The programs are designed to detect any errors in the DECtape Control. Separate binary tapes are available for either control 1 or for a second control.

General Description

The user simply sets the accumulator switches as required and starts the computer at the correct starting address for the particular program. The appropriate messages will be typed when the function is completed or when any error occurs. A brief description of each subroutine is given in the following paragraphs; however, a list of the current subroutines and their starting addresses follows:

<u>Starting Address</u>	<u>Mnemonic</u>	<u>Function</u>
a) 100	TSTMM	Writes, reads, and compares up to 2000 ₈ words written from 3600.
b) 101	WRMKTR	Writes mark track and block mark numbers for unmodified controls with serial numbers 1 through 5.
c) 102	WRVIRG	Writes virgin tape in the forward direction.
d) 103	SUMF	Sum checks tape forward.
e) 104	SUMR	Sum checks tape in reverse.
f) 105	MMGEN	Generates 2000 ₈ words of desired pattern beginning at 3600.
g) 106	MMGENA	Generates 2000 ₈ words beginning at 3600 in arithmetic progression.
h) 107	MMWRT	Writes up to 2000 ₈ words from 3600.
i) 110	MMREAD	Reads up to 2000 ₈ words into core beginning at 5600.
j) 111	WRVIRR	Writes virgin tape in reverse direction.
k) 112	EXER	Exercises requested tapes by writing a virgin tape pattern and sum checking in both directions continuously.
l) 113	MMGENR	Generates 2000 ₈ random numbers beginning at 3600.
m) 114	ROCKER	Moves tape in both directions for constant time in variable modes.

<u>Starting Address</u>	<u>Mnemonic</u>	<u>Function</u>
n) 115	BLROCK	Reads tape in both directions beginning at designated block number for constant number of blocks.
o) 116	RWMKTR	Revised mark track writer for PDP-compatible DECtapes (550 Control serial number 6 and higher). Used prior to WRNVT below.
p) 117	WRNVT	Second pass virgin tape program for revised mark track writer. (To be used for RWMKTR above, only.)
q) 120	STAP	Stop adjustment program. Provides suitable running and stopping delays for correct adjustment of DECtape delays.
r) 121	RBMN	Reads block numbers from requested tape into successive locations, beginning at 3600.
s) 122	SPEXER	Special exerciser. Operates as does the normal exerciser (see k) except that a predescribed pattern is written in place of the virgin tape pattern.

3. REQUIREMENTS

3.3 Equipment

Paper Tape Reader, Teleprinter, DECtapes

4. USAGE

4.5 Errors

- 1) All error messages except the compare error from the TSTM program have the following format:

ERR XXX ZZZ (Y)*

where ERR is the code for the error, XXX is the block mark number of the block involved and ZZZ is the nine bits of the report of the DECtape status using the MMRS instruction. The bits can be interpreted as follows:

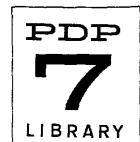
*If an error occurs in the exerciser, Y will indicate the offending unit number (in octal).

Bit 0 = Data flag up
1 = Block flag up
2 = Error flag up
3 = End of tape
4 = Timing error
5 = Reverse
6 = Go
7 = Mark track error
8 = Unable

Error Codes are as follows:

- FMT = Format Error. Check ACS (block number will indicate block requested).
- NTF = Not found.
- ERS = Error flag for other than end of tape, during search.
- ERR = Error flag during reading.
- SUM = Sum check error.
- ERW = Error flag during writing.
- BMW = Block mark read does not correspond to block mark predicted during writing.
- BMC = Block mark read does not correspond to block mark predicted during sum checking.
- INT = Program interrupt occurred from other than error or data flags of DECTape unit.
- FLC = The block end flag occurred before or after 400 data flags had been found. Can occur only during writing virgin tape.* Register MMWA2 contains the count of the number of flags read; i.e., 400400 or 377377 etc.
- NFL = No flags. Program has stopped receiving program interrupts from DECTape unit.
- BUF = Contents of DECTape buffer were wrong after issuing MMWR command and checking by MMRD. Occurs during writing virgin tape only.
- NWR = Actual check sums do not agree with predicted check sums using exerciser, indicating that probably nothing was written. In the exerciser, unit 10 (octal) will always appear with this message. Can also occur if for any reason the first block in a pass was not written (for example, NTF etc.).
- NZB = No zero block. WRNVT could not successfully write the first zero block. Rerun WRNVT (see p).

*Assuming 400₈ word blocks, FLC cannot occur if SPEXER is used.



- 2) Comparison errors will appear as follows:

CMP O ZZZ XXXXXX YYYYYYY

where the first 3 words are as in 1) above, XXXXXX indicates the word written and YYYYYYY indicates the word read. The block number indicators will always be zero.

- 3) When errors occur, the following registers may be of importance.

MMRVC	(3321)	The reverse check sum as read from the block.
MMRDC	(3320)	The forward check sum as read from the block.
MMCC	(3317)	The actual check sum calculated during reading. This should be 7777777.
MMWC	(3316)	The forward check sum as written on the tape during writing. This should agree with MMRDC if the same direction is used.
MMWDC	(3305)	The word count expressed as -N+1. After all words have been written or read, this should say +0.

NOTE: The forward check sum is the one read or written after the data and the reverse check sum is the one read or written before the data. Therefore, both are dependent on the direction being used.

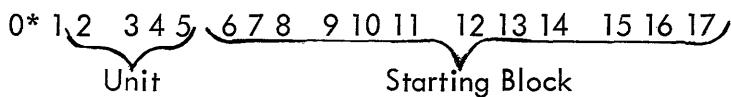
6. DESCRIPTION

a) TSTMM

Writes up to 2000₈ words beginning at 3600 into the indicated blocks in the forward direction, reads it back in the forward direction into 5600 and compares it against the original, word by word. Any error except a comparison error or a sum check error with ACS₀ = 1 will stop the program. In case of a comparison error, the program will continue until all words have been checked.

Use:

- 1) Set ACS as follows:



- 2) Start at 100

*If ACS Bit 0 = 1, sum check errors will be ignored.

- 3) At HLT set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17,
Number of Words

- 4) Press CONTINUE.

b) WRMKTR

Writes the timing and mark tracks and block mark numbers on the tape. (Note that block mark numbers will have the 6-bit mark track code in the high-order six bits. These must be ANDed out when reading block marks through a user's program.) The program can create blocks of various constant lengths. The tape must be placed as close as possible to the beginning of the reel, manually, before starting. This program should be used for unmodified DECtape controls serial numbered 1-5 only.

Use:

- 1) Set ACS as follows: Block number should indicate one higher than the highest block which can be found by the tape subroutines, i.e., if the routines should search for blocks up to 1100, the number 1101 should be used.

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17,
Unit Highest Block Number+1

- 2) Start at 101.

- 3) At HLT, set ACS as follows to indicate the actual number of data words desired per block. (Do not confuse with the number of data marks per block.)

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17,
Number of Words per Block

- 4) Press CONTINUE.

NOTE: This program will work only if the switch to allow writing on the mark track is on. The switch should be turned off immediately after the program is completed.

c) WRVIRG

Writes blocks with correct check sum, whose data portions consist of the numbers 0, 1001, 2002 etc., through 377377, in the forward direction on the tape.* Any error stops the program.

*If other than a 400₈ word block is written, the constant at MMVK (1756) must be changed so as to place the number of words desired in each half of the word. For example, for a 300₈ word block, make the constant 300300, etc. The check sum of a 400₈ word standard virgin tape block is 100100.

Use:

- 1) Set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Unit Starting Block

- 2) Start at 102.

- 3) When HLT occurs set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Number of Blocks

- 4) Press CONTINUE.

d) SUMF

Checks the validity of the data and block marks on the tape by sum checking the tape and checking the intervening block mark numbers in the forward direction. If an error occurs, the program will type a message and continue beginning with the next block until all indicated blocks have been read.

Use:

- 1) Set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Unit Starting Block

- 2) Start at 103.

- 3) When HLT occurs set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Number of Blocks

- 4) Press CONTINUE.

e) SUMR

Same as SUMF (see d) except that the tape is checked in reverse direction.

Use:

- 1) Set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Unit Starting Block

- 2) Start at 104.
- 3) When HLT occurs set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Number of Blocks

- 4) Press CONTINUE.

f) MMGEN

Will spread the pattern set in the ACS into memory from 3600 through 5577.

Use:

- 1) Set ACS to pattern desired.
- 2) Start at 105.

g) MMGENA

Spreads a pattern beginning with 0 and incremented by the contents of the ACS, into memory from 3600 to 5577. For example, if the ACS = 1, the numbers 0 through 1777 will be spread.

Use:

- 1) Set ACS to increment desired.
- 2) Start at 106.

h) MMWRT

Writes up to 2000_8 words in the forward direction from core locations 3600 through 5577.

Use:

- 1) Set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Unit Starting Block

- 2) Start at 107.
- 3) At HLT, set number of words to be written in ACS as follows:

(Max = 2000_8)
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Number of Words

- 4) Press CONTINUE.

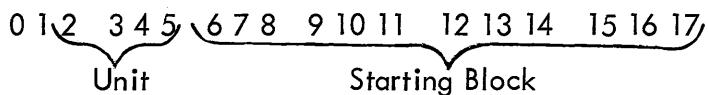
i)

MMREAD

Reads up to 2000_8 words in the forward direction into core locations 5600 through 7577.

Use:

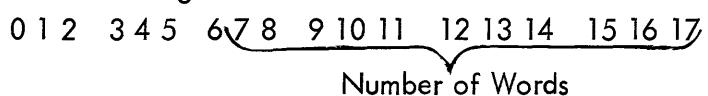
- 1) Set ACS as follows:



- 2) Start at 110.

- 3) At HLT, set number of words to read in ACS as follows:

(Max = 2000_8)



- 4) Press CONTINUE.

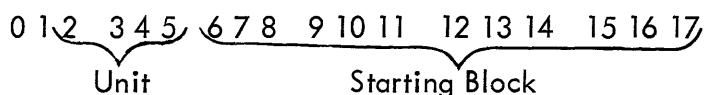
j)

WRVIRR

Same as WRVIRG (see c) except that blocks are written in the reverse direction.

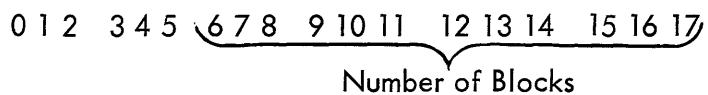
Use:

- 1) Set ACS as follows:



- 2) Start at 111.

- 3) When HLT occurs set ACS as follows:



- 4) Press CONTINUE.

k)

EXER

Exercises a succession of tapes by writing modified virgin tape in the forward direction on all desired tapes, sum checking in the reverse direction on all tapes, sum checking in the forward direction on all tapes, writing modified virgin tape in the reverse direction on all tapes, sum checking in the forward direction on all tapes, and finally sum checking in

the reverse direction on all tapes. The entire cycle will be repeated continuously until ACS bit 0 is made a 1. A message is written at the end of each cycle giving the number of the cycle. The pattern written during each pass is incremented by the pass number. If an error occurs, the program will type a message and continue, beginning with the next block. All error messages will indicate the unit number at the end of the message. The NWR message will stop the program only if it occurs eight times.

Use:

- 1) Set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Starting Block

- 2) Start at 112.

- 3) When HLT occurs set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Number of Blocks

- 4) Press CONTINUE.

- 5) When HLT occurs, set AC switches 1-8 to a 1 to indicate which units numbered 1-8, respectively, are to be tested. For example, to exercise units 1-4 set only switches 1-4 to a 1; to exercise unit 3 only, set switch 3 to a 1 and all others to a 0. Any combination of bits may be used. Only bits 1-8 are examined. These switches can be reset at any time during the running of the program to add or delete units. If $ACS_0 = 1$, the program will stop at the end of a pass.

- 6) Press CONTINUE.

I) MMGENR

Spreads 2000₈ random numbers into locations 3600 through 5577. A different block will be created each time the program is used unless the program is reloaded or the location called RNK (1042) is reset to 736425.

Use:

- 1) Start at 113.

m) ROCKER

The program will move the tape in the mode indicated for multiples of approximately 5-second intervals in both directions. The switches can be changed at any time during the running of the program. The ERROR FLAG is cleared when a Block End Flag is detected.

Meaning of the AC bits:

12 = Go Flip-Flop On

13 = Reverse Flip-Flop On

14 = Not used

15 - 17 = Mode: 0 = Move

1 = Search

2 = Read

3 = Write

4 = Spare

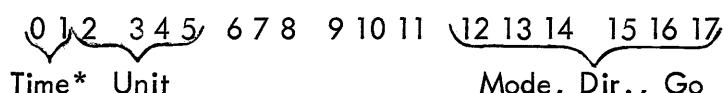
5 = Read through block ends

6 = Write through block ends

7 = Write timing track

Use:

- 1) Set ACS as follows:



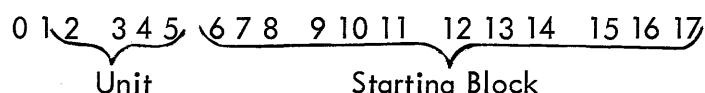
- 2) Start at 114.

n) BLROCK

Reads DECTape blocks beginning at the block specified, for a variable number of blocks in either direction. If end of tape is reached, repositioning at the starting block occurs automatically. All other errors are cleared when a Block End Flag is detected. Read Mode only, is used. To stop the program, set bit 0 of the ACS to 1.

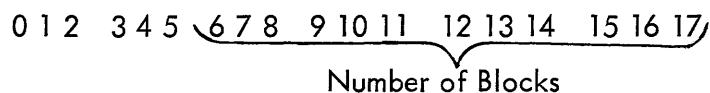
Use:

- 1) Set ACS as follows:



- 2) Start at 115.

- 3) At HLT, set ACS as follows:



- 4) Press CONTINUE.

*00 = 5 seconds, 01 = 10 seconds, 10 = 15 seconds, 11 = 20 seconds approximately.

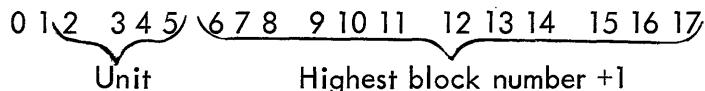
RWMKTR

For those tape controls using the revised bit format (serial number 6 and later), two new programs have been designed to write the mark track, virgin tape pattern, and to check the tape. These are described fully below: (WRMKTR should still be used for unmodified controls with serial numbers 1-5 as described in paragraph b, page 6.)

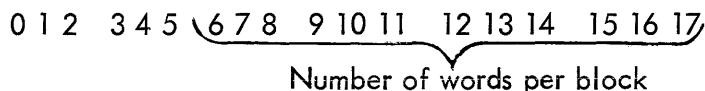
RWMKTR writes only the timing and mark tracks on the tape. The program can create any number of blocks of any desired constant length.

Use:

- 1) Place the reel as close as possible to the physical start of the tape.
 - 2) Set the switch which controls writing to ON.
 - 3) Set ACS, as follows, to one higher than the highest block number to be legally found by the subroutines:



- 4) Start at 116.
 - 5) At HLT, set ACS as follows to indicate the actual number of data words per block:



- 6) Press CONTINUE.
 - 7) When program is finished, turn mark track switch off, remount tape if necessary, and continue with the virgin tape program (WRNVT) explained below.

p) WRNVT

Used to write the virgin tape pattern and the block mark numbers on a tape with a newly created timing and mark track. This program should never be used except after RWMKTR above. It is not a normal virgin tape program, but one which writes through block ends. After the block mark numbers and virgin tape have been written in the forward direction, the tape is automatically sum checked in reverse, showing all errors. In addition, the first block on the tape is rewritten to guarantee a 0 as the first block mark number. (There are two blocks numbered 0 at the front of the tape.)

Use:

- 1) Set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Unit

- 2) Start at 117.

3) The block mark track will be used to determine the length of each block, and the end of tape will be used to determine the number of blocks. Since block mark numbers are now put on in a second pass, it is no longer necessary to AND out bits 0-5 when reading block mark numbers produced with this program.

q) STAP

The program allows visual inspection and manual regulation of the DECtape Stop Adjustment by the following routine: If $ACS_0 = 0$ the tape will move forward (in move mode) from its present position for 1 second, stop for 1 second, move in reverse for 1 second, stop for 1 second, then repeat the cycle.

If $ACS_0 = 1$, the tape will move forward for 0.12 seconds, stop for 1 second, move in reverse for 0.12 seconds, stop for 1 second, then repeat the cycle.

Use:

- 1) Set ACS as follows:

0* 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Unit

- 2) Start at 120.

Any of the accumulator switches can be changed at any time. Care should be taken that the tape unit used is not presently in the end zones, as stopping will be premature.

r) RBMN

Rewinds the indicated tape unit, then searches forward reading all block mark numbers and placing them (without masking) in successive locations beginning at 3600. When the forward end of the tape is reached, the remainder of the area up through 7577 is filled with minus zeros (777777), thus making clear what the last block mark number found was. The area can be checked by using the Examine Key. For the standard tape, the first block number 0 will be in location 3600 and number 1101 will be in location 4702.

*Move for 0.12 second if on.

Use:

- 1) Set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Unit

- 2) Start at 121.

s) SPEXER

This special exercise program operates for the most part as does the normal exercise program (see k) except for the following differences:

- 1) The user must generate the pattern to be written rather than use the standard virgin tape pattern.
- 2) The check for flag count errors (FLC) is omitted as it depends on a check of the pattern used.

Use:

- 1) Generate the pattern desired by using MMGEN (see f), MMGENA (see g) or MMGENR (see l). If no pattern is generated, the block written will contain the information currently in memory beginning at location 3600.

- 2) Set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Starting Block

- 3) Start at 122.

- 4) When HLT occurs, set ACS as follows:

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Number of Blocks

- 5) Press CONTINUE.

- 6) When HLT occurs, set Accumulator switches, bits 1-8, to a 1 to indicate which units numbered, 1-8 respectively, are to be tested. For example, to exercise units 1-4, set bits 1-4 only to a 1, to exercise units 1, 3, 5 and 8, set bits 1, 3, 5 and 8 to a 1 etc. Any combination of bits may be used. These switches can be reset at any time during the running of the program to add or delete units. Only bits 1-8 are examined. If $ACS_0 = 1$, the program will stop at the end of a pass.

- 7) Press CONTINUE.

9. PROGRAM

9.4 Listing

```
EWIN
DEC TOG
MMAUTO=17
21/
    CLA
    MMLC
    TIN
    LAW MSG1
    TSR
    LAC 20
    TAD (-0)
    AND (7777)
    JMS TWZ6
X1,
    HLT
    JMP .-1
50/
    CLA
    MMLC
    JMP X1
DISMIS=JMP .
    LAC n
    RAL
    LAC A#CSAVE
    ION
    JMP 1 n
100/
    JMP TSTM /READ, WRITE AND CHECK BLOCKS, 100
    JMP WRMKTR /WRITE MARK TRACK, 101
    JMP WRVIRG /WRITE VIRGIN TAPE, 102
    JMP SUMF /SUMCHECK FWD, 103
    JMP SUMR /SUMCHECK REV, 104
    JMP MMGEN /GENERATE 1 PATTERN BLOCKS, 105
    JMP MMGENA /GENERATE ASCENDING BLOCKS, 106
    JMP MMWRT /WRITE BLOCKS, 107
    JMP MMREAD /READ BLOCKS, 110
    JMP WRVIRR /WRITE VIRGIN TAPE REVERSE, 111
    JMP EXER /WVF,SCR,SCF,WVR,SCF,SCR, 112
    JMP MMGENR /GENERATE RANDOM BLOCKS, 113
    JMP ROCKER /MOVES TAPE IN TWO DIRECTIONS FOR FIXED TIME
    JMP BLROCK /READS TAPE BIDIRECTIONALLY FOR FIXED LENGTH
    JMP RWMKTR /REVISED MARK TRACK WRITER, 116
    JMP WRNVT /VIRGIN TAPE FOR NEW TAPE ONLY, 117
```

JMP STAP /STOP ADJUSTMENT PROGRAM, 120
JMP RBMN /RECORD BLOCK MARK NUMBERS, 121
JMP SPEXER /EXERCISE WITH PREDETERMINED PATTERN, 122

130/
/TEST DECTAPE SUBROUTINES

MMWRA=3600

MMRDA=MMWRA+2000

TSTMM, JMS CLRFLG

LAS /SELECTION AND BLOCK NUMBER

DAC #K1

DAC TSTM2

DAC TSTM3

SMA

JMP .+3

LAC (JMP MMRD4+10)

DAC MMOD1+6

JMS TSTNWD /PICK UP NUMBER OF WORDS

LAC C1

CMA

ADD (MMWRA)

DAC TSTM2+2

ADD (MMRDA-MMWRA)

DAC TSTM3+2

JMS MMWRS

LAC K1 /BLOCK NUMBER

JMP TSTM6 /ERROR RETURN

TSTM2, 0 /UNIT SELECTION

MMWRA

0 /CORE END

JMS TSTM8 /MULTI-PROGRAM

JMS MMRDS

LAC K1 /BLOCK NUMBER

JMP TSTM6 /ERROR RETURN

TSTM3, 0 /UNIT SELECTION

MMRDA /CORE START

0 /CORE END

JMS TSTM8 /MULTI-PROGRAM

LAW MMWRA-1

DAC 10

LAW MMRDA-1

DAC 11

LAC C1

DAC #C2

TSTM4, LAC I 10

SAD I 11

SKP

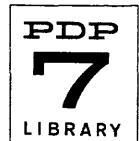
```
JMP TSTMM7 /COMPARE ERROR
ISZ C2
JMP TSTMM4
JMS TYPEND
LAS
SMA
JMP X1 /DONE
JMS LWRFLG
JMP TSTMM2-3

TSTMM6, JMS DECERR
JMP X1

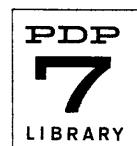
TSTMM7, LAW
JMS DECERR
TSP
LAC 10
DAC #C3
LAC I C3
JMS TW6
TSP
LAC 11

DAC C3
LAC I C3
JMS TW6
JMP TSTMM4+4 /CONTINUE

TSTMM8, @
LAC MMWA1
ISZ IF
JMP .+4
IOF
LAW 1300
JMP MMERRX-1
ISZ MMDONE
JMP TSTMM8+1
IOF
JMP I TSTMM8
/FOR INTERRUPT
TSTMM9, DAC ACSAVE
DZM IF
MMEF
SKP
JMP MMERR
MMDF
SKP
JMP MMDATA
LAW 1100
JMP MMERRX-1
```



/CONSTANTS AND SUBROUTINES
.TYPOK, %
 TIN
 LAC (FLEX OK)
 TY3
 JMP I TYPOK
TYPEND, %
 TIN
 LAC (FLEX END)
 TY3
 JMP I TYPEND
ERRTAB, FLEX CMP FLEX FMT FLEX NTF
 FLFX ERS FLEX ERR FLEX SUM
 FLEX ERW FLEX BMW FLEX BMC
 FLEX INT FLEX FLC FLEX NFL
 FLFX BUF FLEX NWR
ERRTAB+3W
DECERR, %
 DAC ERRWA
 SAD (LAW)
 DZM MMWA1
 SAD (LAW 100)
 JMP .+4
 SAD (LAW 200)
 SKP
 JMP .+3
 LAC MMBLKM
 DAC MMWA1
 TIN
 LAC' ERRWA
 RTR
 RTR
 RTR ,
 AND (77)
 ADD (LAC ERRTAB)
 DAC .+1
ERRWA, LAC ERRTAB /MODIFIED
TY3
TSP
LAC MMWA1
JMS TWZ6
TSP
LAC MMRSA
TWORD
3
JMP I DECERR



TW6, 0
TWORD
6
JMP I TW6

TWZ6, 0
TWORDZ
6
JMP I TWZ6

MSG1, FLEX CAL FLEX FR FLEX M T-2

/WRITE VIRGIN TAPE

WRVIRG, LAC (JMS MMVT)
DAC WRV1-2
JMS CLRFLG
LAS /UNIT SELECTION AND BLOCK NUMBER
DAC K1
DAC WRV1+2
HLT
LAS /NUMBER OF BLOCKS
AND (7777)
CMA
ADD (1)
DAC WRV1
LAC (1001)
JMS GENRLA
0 /NUMBER TO START AT

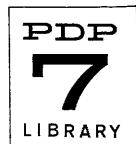
JMS MMVT
LAC K1 /BLOCK
WRV1, 0 /NO OF BLOCKS
JMP TSTM6 /ERROR
0 /UNIT
JMS TSTM8

WRV2, JMS TYP0K
JMP X1

WRVIRR, LAC (JMS MMVTR)
JMP WRVIRG+1

/SUM CHECK PROGRAMS

SUMF, LAC (JMS MMSCF)
DAC SUMF1
JMS CLRFLG
LAS /UNIT AND BLOCK NUMBER
DAC SUMF1+4
AND (7777)
DAC K1
HLT



```
        LAS      /NUMBER OF BLOCKS
        AND (7777)
        CMA
        ADD (1)
        DAC SUMF1+2
SUMF1,    JMS MMSCF /OR MMSCR, MODIFIED
        LAC K1   /BLOCK
        0      /NUMBER OF BLOCKS, -N+1
        JMP SUMF2 /ERROR
        0      /UNIT
        JMS TSTM8
SUMF1A,   JMS TYPEND
        JMP X1
SUMF2,   JMS DECERR
        LAC MMWA1
        CMA
        ADD K1
        SPA
        CMA
        ADD SUMF1+2
        ADD (1)
        SMA
        JMP SUMF1A
        DAC SUMF1+2
        LAC MMWA1
        TAD MMSCFK
        DAC K1
        JMS LWRFLG
        JMP SUMF1
SUMR,     LAC (JMS MMSCR)
        JMP SUMF+1

/GENERATE BLOCKS
MMGEN,   LAC (CLA)
        DAC MMGEN1
        LAC (LAS)
        DAC MMGEN1-3
        LAW MMWRA-1
        DAC MMAUTO
        LAM -MMRDA+MMWRA+1
        DAC MMWA2
        LAS
        DAC MMWA1
        DAC I MMAUTO
MMGEN1,   CLA      /OR LAS
        ADD MMWA1
        DAC MMWA1
        ISZ MMWA2
        JMP MMGEN1-1
        JMP wRV2
```



```
MMGENA, LAC (LAS)
DAC MMGEN1
LAC (CLA)
JMP MMGEN+3

/READ BLOCKS
MMREAD, LAC (JMS MMRDS)
DAC MREAD2
LAW MMRDA
DAC MREAD2+4
LAS /UNIT AND BLOCK NUMBER
DAC MREAD2+3
JMS TSTNWD /PICK UP NO OF WDS
LAC C1
CMA
ADD MRFA02+4
DAC MREAD2+5
JMS CLRFLG

MREAD2, JMS MMRDS /OR JMS MMWRS
LAC .+2 /BLOCK NUMBER
JMP TSTM6 /ERROR
@ /UNIT
LAW MMRDA /OR LAW MMWRA, CORE START
@ /CORE END, MODIFIED

JMS TSTM8
JMP WRV2

/WRITE BLOCKS
MMWRT, LAC (JMS MMWRS)
DAC MREAD2
LAW MMWRA
JMP MMREAD+3

/CLEAR FLAGS, SET UP INTERRUPT, AND MODIFY READ AND WRITE RTNES
CLRFLG, @
    IOF      DCF      CRRB
    CPCF     LPCF     LSCE
    700102   PCF      KRB
    TCF      MSI      CLOF
    CLA      707604   701604 /MMLC FOR 1ST AND 2ND CONTROLS
    LAC (JMP MMMOD1)
    DAC MMRD2-1
    LAC (JMP MMMOD1+3)
    DAC MMRD4+3
    LAC (JMP MMMOD1+7)
```

```
DAC MMWR4+4
LAC (JMP TSTM9)
DAC 1
DZM #IF
LAC (LAC MMVK)
DAC MMVWR3+10
LAC (JMP MMRD4+4)
DAC MMMOD1+6
LAC (JMS TYPFND)
DAC SUMF1A
JMP I CLRFLG

MMMOD1,   DAC MMSUM
           DAC M#MRVC
           JMP MMRD2

           DAC M#MRDC
           ADD MMSUM
           DAC M#NCC
           JMP MMRD4+4           /OR JMP MMRD4+10

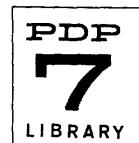
           DAC M#MWG
           MMWR
           JMP MMWR4+5

TSTNWD,   %
           HLT
           LAS      /NUMBER OF WORDS
           AND (7777)
           CMA
           ADD (1)
           DAC #C1
           ADD (MMRDA-MMWRA)
           SMA      /TOO MANY WORDS
           JMP I TSTNWD
           LAM -MMRDA+MMWRA+1
           JMP .-5

MMOD1=MMMOD1

START

EXFR,     CONSECUTIVE UNIT EXERCISER
           JMS CLRFLG
           LAC (JMP MVT2)
           DAC MMVWR3+10
           LAC (JMS GENBLA)
           DAC EXER1+3
           LAS      /BLOCK NUMBER
```



```
AND (7777)
DAC C2
HLT
LAS      /N
AND (7777)
DAC C3
CMA
ADD (1)
DAC E#X1
LAC C2
ADD C3
TAD (-0)
DAC C3      /ENDING BLOCK
DZM TYPOK
LAM -7
DAC W#A6
HLT      /SET UP UNITS TO WORK ON

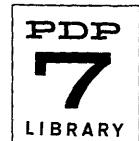
EXER1,    LAC TYPOK
          DAC .+3
          LAC (1001)
          JMS GENBLA
          N      /MODIFIED, NUMBER TO START WITH

          LAC (JMS EXVT)
          JMS CONSEC
          LAC (JMS SUMSBR)
          JMS CONSEC
          JMS MWRCHK
          LAC (JMS SUMSBF)
          JMS CONSEC

          LAC (JMS EXVTR)
          JMS CONSEC
          LAC (JMS SUMSBF)
          JMS CONSEC
          JMS MWRCHK
          LAC (JMS SUMSBR)
          JMS CONSEC

          ISZ TYPOK
          JMS TYPEND
          TSP
          LAC TYPOK
          JMS TW#6

          LAS
          SPA
          JMP X1
          JMS LWRFLG
          JMP EXFR1.
```



CONSEC, Ø
DAC CONSC2
LAC -7
DAC W#A4
LAS
DAC W#A5
CLA
CONSC1, ADD (10000)
DAC SUMSRF+13
LAC WA5

RAL
DAC WA5
SPA
CONSC2, JMS /MODIFIED
LAC SUMSRF+13
ISZ WA4
JMP CONSC1
JMP I CONSEC

SUMSRF, Ø
LAC (JMS MMSCF)
DAC .+5
LAC C2
DAC K1
LAC EX1
DAC .+3
JMS MMSCF /OR MMSCR, MMVT, MMVTR
LAC K1 /BLOCK NUMBER
Ø /-N+1
JMP EXERR /ERROR
Ø /UNIT
JMS TSTMM8
JMP I SUMSRF

SUMSBR, Ø
LAC .-1
DAC SUMSRF
LAC (JMS MMSCR)
DAC I SUMSRF+2
LAC C3
JMP SUMSRF+4

EXVT, Ø
LAC .-1
DAC SUMSRF
LAC (JMS MMVT)
JMP SUMSRF+2

EXVTR, 0
LAC .-1
DAC SUMSBF
LAC (JMS MMVTR)
JMP SUMSRR+4

LWRFLG, 0
TSF
JMP .-1
TCF
JMP I LWRFLG

EXERR, JMS DECERR
TSP
LAC SUMSBF+13
TWORDZ
2
LAC MMWA1
CMA
ADD K1
SPA
CMA
ADD SUMSBF+11
ADD (1)
SMA
JMP .+7 /DONE IN THIS DIRECTION

DAC SUMSBF+11
LAC MMWA1
TAD MMSCFK
DAC K1
JMS LWRFLG
JMP SUMSBF+7

JMS LWRFLG
JMP I SUMSBF

RN, 0
LAC RNK
CLL!RAR
SZL
XOR (400000)
XOR (335671)
ADD (335671)
DAC RNK
JMP I RN
736425

RNK,



MMGENR, LAW MMWRA-1 DAC MMAUTO
 LAM -MMRDA+MMWRA+1
 DAC MMWA2
 JMS RN
 DAC I MMAUTO
 ISZ MMWA2
 JMP .-3
 JMP WRV2

MWRCHK, 0
 LAC MMRVC
 SAD MMWC
 JMP I MWRCHK
 LAC (JMP X1)
 ISZ WA6
 LAC .-3
 DAC MVT2-1
 LAW 1500 /NWR
 JMS DECERR
 TSP
 LAC SUMSRF+13
 TWORDZ
 2
 JMS LWRFLG
 JMP I MWRCHK /OR JMP X1

MVT2, LAC TYPOK
 ADD MMVK
 JMP MMVWR3+11

/INTERNAL BLOCK GENERATORS

/GENERATE BLOCKS WITH SAME CONTENTS
/FORMAT LAC CONTENTS
/
JMS GENBL

GENBL, 0
 DAC MMWA1
 LAM -MMRDA+MMWRA+1
 DAC MMWA2
 LAW MMWRA-1
 DAC MMAUTO
 LAC MMWA1
 DAC I MMAUTO
 ISZ MMWA2
 JMP .-2
 JMP I GENBL

/GENERATE BLOCKS IN ARITHMETIC PROGRESSION
/FORMAT LAC FACTOR /NUMBER TO ADD
/ JMS GENBLA
/ X /NUMBER TO START WITH

GENBLA, Ø

DAC MMWA1
LAM -MMRDA+MMWRA+1
DAC MMWA2
LAW MMWRA-1
DAC MMAUTO
LAC I GENBLA
DAC I MMAUTO
ADD MMWA1
ISZ MMWA2
JMP .-3
ISZ GENBLA
JMP I GENBLA

SPEXER, JMS CLRFLG
LAC MMVWR3+12
DAC MMVWR3+10
LAC (SKP)
JMP EXER+4

START

ROCKER PROGRAM
ROCKER, JMS CLRFLG
DZM C1
JMS MMWAIT
LAS
XOR C1
MMSE
MMLC
RAL
RTL
AND (3)
CMA
DAC C3

```
ROCK1,    LAC (DECIMAL -89285 OCTAL)      /56 US LOOP
          DAC C2
          MMBF
          JMP .+5
          MMEF
          JMP .+3
          LAS
          MMSE
          MMDF
          SKP
          MMRD
          ISZ C2
          JMP ROCK1+2
          ISZ C3
          JMP ROCK1

          LAC C1
          XOR (20) /CHG DIRECTION
          DAC C1
          JMP ROCKER+2

/ROCK FOR CONSTANT NUMBER OF BLOCKS
BLROCK,   JMS CLRFLG
          DZM C1
          LAS      /UNIT AND BLOCK NUMBER
          DAC BLRCK1
          HLT
          LAS      /NUMBER OF BLOCKS
          AND (7777)
          CMA
          ADD (1)
          DAC C2      /BLOCK COUNTER
BLRCKA,   CLA
          MMLC
          JMS MMSCH1
          LAC .+3      /BLOCK NUMBER
          JMP .-2      /ERROR, SEARCH AGAIN
          DISMIS
BLRCK1,   0      /UNIT
          LAC MMWA1 /PRESENT BLOCK LOCATION
          ISZ MMDONE
          JMP .-2

          IOF
BLRCK2,   LAC C2
          DAC C3
          LAW 42      /READ FORWARD
          XOR C1      /GET DIRECTION
          MMLC      /READ FORWARD OR REVERSE
BLRCK3,   MMEF
          SKP
```

```
JMP BLRCK5           /ERROR FLAG
MMDF
SKP
MMRD
MMRF
JMP BLRCK3
MMRD
MMEF
JMP .+3
LAC BLRCK1
MMSE
ISZ C3
JMP BLRCK3

BLRCK4, LAS
SPA
JMP 50
LAC C1
XOR (20)
DAC C1
JMP BLRCK2

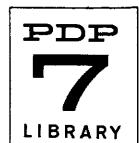
BLRCK5, MMRS
AND (40000)
SAD (40000)          /EOT
JMP BLRCKA           /REPOSITION TAPE
JMP BLRCK3+3

START
```

PDP-7 DECTAPE SUBROUTINES, CONTROL 1, LMH 12-2-64
/PDP-7 DECTAPE SEARCH SUBROUTINE
/DISMIS MUST BE DEFINED AS JMP TO DISMISS INTERRUPT ROUTINE

MMWR=707504
MMLC=707604
MMSE=707644
MMRS=707612
MMDF=707501
MMBF=707601
MMEF=707541
MMRD=707512
SKP7=703341

/FORMAT JMS MMSCH /OR MMSCH1 OR MMSCHR
/ LAW B /OR LAC (B), BLOCK NUMBER
/ JMP X /ERROR RETURN
/ JMP Y /SEARCH COMPLETED RETURN
/ ZZ0000 /UNIT SELECTION
/ MULTI-PROGRAM RETURN

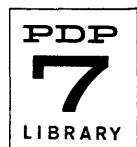


/LEAVE IN SEARCH REVERSE MODE
MMSCHR, 0
LAC .-1
DAC MMSCH1
LAC (JMP MMSCH6+2)
DAC MMSCH3+1
CLA
JMP MMSCH1+4

/LEAVE IN FORWARD DIRECTION WITH TAPE STOPPED
MMSCH, 0
LAC .-1
DAC MMSCH1
LAC (JMP MMSCH6)
JMP MMSCH1+2

/LEAVE IN SEARCH FORWARD MODE
MMSCH1, 0
LAC (JMP MMSCH6+2)
DAC MMSCH3+1
CLC
DAC M#MSRK
TAD (1)
DAC M#MSFK
LAW 61
DAC M#MWA3 /CURRENT DIRECTION
XCT I MMSCH1 /PICK UP BLOCK NUMBER
ISZ MMSCH1 /POINTS TO ERROR RETURN
AND (7777)
DAC M#MBLK^M /BLOCK TO SEARCH FOR
SNA
JMP MMSCH4
ADD MMEK
SMA
JMP MMSCH4 /FORMAT ERROR
LAM -7
DAC M#MSUM
LAC I MMSCH1 /CHG OF DIRECTION COUNTER
DAC MMERRX
ISZ MMSCH1 /ERROR RETURN
LAC I MMSCH1 /COMPLETION RETURN
DAC MMSCH7
ISZ MMSCH1
JMS MMWAIT /CHECK IF DELAY IS NECESSARY
LAC I MMSCH1 /UNIT SELECTION

MMSE
ISZ MMSCH1 /POINTS TO MULTI-PROGRAM RETURN
LAC (NOP)
DAC MMSAVE
ION
MMTURN, ISZ MMSUM
JMP MMERRX+2
LAW 200 /NOT FOUND
JMP MMEK+1
MMERRX, JMP . /ERROR EXIT
HLT /ERROR EXIT WAS NOT JMP INSTR
LAW 41
SAD MMWA3
JMP MMREV
DAC MMWA3
MMLC
LAC (SMA)
DAC MMSCH2
LAC MMBLK
TAD MMSFK
DAC M#MW A2 /BLOCK TO LOOK FOR IN THIS DIRECTION
NZM M#MIDONE
MMSAVE, NOP /UR DISMIS
LAC (DISMIS)
DAC MMSAVE
JMP I MMSCH1 /CONTINUE MULTI-PROGRAMMING
MMREV, LAW 61
DAC MMWA3
MMLC
LAC (SPA)
DAC MMSCH2
LAC MMBLK
TAD MMSBK
JMP MMSAVE-2
MMERR, MMPS
AND (40000)
SAD (40000)
JMP MMTURN
LAW 300 /NON-EOT ERROR DURING SEARCH
JMP MMERRX-1
MMDATA, MMRD
AND (7777)
DAC M#MWA1
SAD MMWA2
JMP MMSCH3
CMA
ADD MMWA2

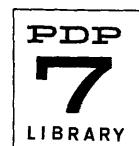


MMSCH2,	SMA	/OR SPA FOR REVERSE
	JMP MMSAVE-1	/KEEP GOING
	JMP MMTURN	/TURN AROUND
MMSCH3,	SAD MMBLKM	
	JMP MMSCH6	/OR MMSCH6+2
	JMP MMTURN	
MMSCH4,	LAW 100	/FORMAT ERROR
	MMLC	
	JMP I MMSCH1	
MMSCH5,	LAW 100	
	JMP MMERRX-1	/FORMAT ERROR
MMSCH6,	CLA	
	MMLC	
	CLC	
	DAC MMDONE	
MMSCH7,	JMP .	/EXIT
MMEK,	DECIMAL -576 OCTAL	

DAC MMSCH
MMRS
DAC M#MRSA
LAC MMSCH
MMLC
JMP MMERRX

/35 MILLISECOND SELECT DELAY LOOP

MMWAIT,	0	
	XCT I MMWAIT	/PICK UP SELECT
	AND (170000)	/CHECK SELECT ONLY
	SAD MMCHK-1	
	JMP I MMWAIT	/SAME SELECT
	DAC MMCHK-1	/SAVE SELECT
	CLA	
	MMSE	/SELECT UNIT ZERO
	LAM DECIMAL -5000 OCTAL	
	SKP7	/IS THIS A PDP-7?
	LAM DECIMAL -1094+1 OCTAL	/COUNT 35 MS
	DAC MMSCH	
	ISZ I .-1	
	JMP .-1	
	JMP I MMWAIT	
	0	/SAVE SELECTION



/PDP-7 DECTAPE READ AND WRITE FORWARD SUBROUTINES
/USES AUTO-INDEX REGISTER NAMED MMAUTO WHICH MUST BE DEFINED

/COMMON ROUTINE FOR PICKING UP CONSTANTS AND SEARCHING FOR BLOCK
MMCHK, 0
ADD (-1)
DAC MMAUTO
LAC I MMAUTO /BLOCK NUMBER
DAC MMCHK1+1
LAC I MMAUTO /ERROR RETURN
DAC MMERRX
DAC MMCHK1+2
LAC I MMAUTO /UNIT SELECTION
DAC MMCHK1+4
CLC
TAD I MMAUTO /STARTING ADDRESS
AND (17777)
DAC M#MW4
CLC
TAD I MMAUTO /ENDING ADDRESS
AND (17777)
CMA
ADD MMWA4
SMA
JMP MMSCH5 /ILLEGAL FORMAT
DAC M#MWDC /WORD COUNT
MMCHK1,
JMS MMSCH1
LAW . /BLOCK NUMBER, MODIFIED
JMP . /ERROR RETURN, MODIFIED
JMP MMCHK2 /END RETURN
Ø /UNIT SELECTION, MODIFIED
MMCHK2,
JMP I MMAUTO /MULTIPROCESS WITH MAIN PROGRAM
LAC MMWA4
DAC MMAUTO
LAC (DISMIS)
DAC MMSCH7
JMP I MMCHK

/DECTAPE READ SUBROUTINE
/FORMAT JMS MMRDS
/
/ LAW R /OR LAC (B), BLOCK NUMBER
/
/ JMP X /ERROR RETURN
/
/ ZZ0000 /UNIT SELECTION
/
/ C1 /CORE STARTING ADDRESS
/
/ C2 /CORE ENDING ADDRESS, INCLUSIVE
/
/ MULTI-PROGRAM RETURN

MMRDS, 0
LAC MMRDS
JMS MMCHK
LAW 42 /READ FORWARD
MMLC
LAC (DAC I MMAUTO)
DAC MMRD3
MMRD1, MMEF
JMP .+3
LAW 400
JMP MMERRX-1 /ERROR FLAG DURING READING
MMDF
JMP MMRD1
MMRD
DAC MMSUM
MMRD2, MMEF
SKP
JMP MMRD1+2 /ERROR FLAG DURING READING
MMDF
JMP MMRD4
MMRD
MMRD3, DAC I MMAUTO /OR NOP
ADD MMSUM
DAC MMSUM
ISZ MMWDC
JMP MMRD2
LAC (NOP)
DAC MMRD3
JMP MMRD2
MMRF
JMP MMRD2
MMRD
ADD MMSUM
SAD (-0)
JMP .+3
LAW 500 /SUM CHECK READING
JMP MMERRX-1
ISZ MMWA1 /UPDATE CURRENT BLOCK ADDRESS
LAC (DAC I MMAUTO)
SAD MMRD3
JMP MMRD1
JMP MMSCH6 /GOOD EXIT

/DECTAPE WRITE SUBROUTINE

/FORMAT JMS MMWR\$
/ LAW R /OR LAC (B), BLOCK NUMBER
/ JMP X /ERROR RETURN
/ ZZ0000 /UNIT SELECTION
/ C1 /CORE STARTING ADDRESS



/ C2 /CORE ENDING ADDRESS, INCLUSIVE
/ MULTI-PROGRAM RETURN

MMWRS, Ø
LAC MMWRS
JMS MMCHK
LAC (LAC I MMAUTO)
DAC MMWR3

MMWR1, CLC
DAC MMSUM
LAW 43 /WRITE FORWARD
MMLC

MMWR2, MMEF
JMP .+3
LAW 600 /ERROR FLAG DURING WRITING
JMP MMERRX-1
MMDF
JMP MMWR4

MMWR3, LAC I MMAUTO /OR CLA
MMWR
ADD MMSUM
DAC MMSUM
ISZ MMWDC
JMP MMWR2
LAC (CLA)
DAC MMWR3
JMP MMWR2

MMWR4, MMBF
JMP MMWR2
LAC MMSUM
CMA
MMWR
LAW 41 /SEARCH FORWARD
MMLC
MMFF
SKP
JMP MMWR2+2 /ERROR DURING WRITING
MMDF
JMP .-4
MMRD
ISZ MMWA1 /UPDATE CURRENT BLOCK ADDRESS
AND (7777)
SAD MMWA1
JMP .+3
LAW 700 /BLOCK MARK ERROR DURING WRITING
JMP MMERRX-1
LAC (LAC I MMAUTO)
SAD MMWR3
JMP MMWR1
JMP MMSCH6 /GOOD EXIT

START



DECTAPE VIRGIN TAPE AND SUMCHECK ROUTINES

```
/VIRGIN TAPE SUBROUTINE
/FORMAT    JMS MMVT  /OR MMVTR
/          LAW B      /OR LAC (B), BLOCK NUMBER
/          -N+1       /NUMBER OF BLOCKS
/          JMP X      /ERROR RETURN
/          ZZ0000     /UNIT SELECTION
/MULTIPROGRAM RETURN

MMVT,      0
          LAC (JMS MMVCHK)
          DAC MMVWR1-1
          LAC (1)
          DAC MMSCFK
          LAW 41      /SEARCH FWD
MMVTR2,    DAC MMVWR3+6
          ADD (2)
          DAC MMVWR1
          LAC MMVT
          JMS MMVCHK           /OR MMVCHR
MMVWR1,    LAW 43      /WRITE FWD OR WRITE REV
          MMLC
          CLC
          DAC MMSUM
          LAC (SAD MMWRA-1)
          DAC MMAUTO
MMVWR2,    LAC I MMAUTO
          MMEF
          JMP .+3
          LAW 600.   /ERW, ERROR DURING WRITING
          JMP MMERRX-1
          MMDF
          JMP MMVWR3
          MMWR
          MMRD           /CHECK IF BUFFER IS CORRECT
          XCT MMAUTO
          JMP .+3
          LAW 1400   /RUF, BUFFER INCORRECT
          JMP MMERRX-1
          ADD MMSUM
          DAC MMSUM
          JMP MMVWR2
MMVWR3,    MMBF
          JMP MMVWR2+1
          LAC MMSUM
          CMA
```

```
DAC MMWC
MMWR
LAW 41      /SEARCH FORWARD OR REVERSE
MMLC
LAC MMVK  /MODIFIED BY EXERCISERS
XCT MMAUTO
JMP .+3
LAW 1200  /FLC, INCORRECT NUMBER OF FLAGS
JMP MMERRX-1
LAC MMWA1
TAD MMSCFK
DAC MMWA1
MMEF
SKP
JMP MMVWR2+3          /ERROR DURING WRITING
MMDF
JMP .-4
MMRD
AND (7777)

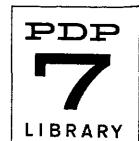
SAD MMWA1
JMP .+3
LAW 700  /BMW, BLOCK MARK INCORRECT
JMP MMERRX-1
ISZ MMWDC
JMP MMVWR1          /CONTINUE
JMP MMSCH6          /EXIT

MMVTR,      0
LAC .-1
DAC MMVT
LAC (JMS MMVCHR)
DAC MMVWR1-1
CLC
DAC MMSCFK
LAW 61      /SEARCH REV
JMP MMVTR2

MMVK,      400400      /LAST VIRGIN TAPE WORD +1
```

/DECTAPE SUM CHECK PROGRAMS

```
/FORMAT JMS MMSCF /OR MMSCR
/
/      LAW B      /OR LAC (B), BLOCK NUMBER
/      -N+1      /N=NUMBER OF BLOCKS
/      JMP X      /ERROR RETURN
/      ZZ0000      /UNIT SELECTION
/      MULTI-PROGRAM RETURN
```



MMSCF, 0
 LAC (JMS MMVCHK)
 DAC MMSCF1
 LAC (1)
 DAC MMSCFK
 LAW 41 /SEARCH FORWARD
MMSCR2,
 DAC MMSCF5
 ADD (1) /READ MODE
 DAC MMSCF1+1
 LAC MMSCF
 JMS MMVCHK /OR MMVCHR
 LAW 42 /READ FWD OR REV, MODIFIED
 MMLC
MMSCF2,
 MMEF
 JMP .+3
 LAW 400 /ERROR DURING READING, ERR
 JMP MMERRX-1
 MMDF
 JMP MMSCF2
 MMRD /READ REVERSE CHECKSUM
 DAC MMSUM
 DAC MMRVC
MMSCF3,
 MMEF
 SKP
 JMP MMSCF2+2 /ERROR DURING READING
 MMDF
 JMP MMSCF4
 MMRD /DATA
 ADD MMSUM
 DAC MMSUM
 JMP MMSCF3 .
MMSCF4,
 MMBF
 JMP MMSCF3
 MMRD
 DAC MMRDC
 ADD MMSUM
 DAC MMCC
 SAD (-0)
 JMP .+3
 LAW 500 /SUM CHECK
 JMP MMERRX-1
MMSCF5,
 LAW 41 /SEARCH FWD OR REVERSE, MODIFIED
 MMLC
 LAC MMWA1
 TAD MMSCFK
 DAC MMWA1
 MMEF

```
SKP
JMP MMSCF2+2           /ERROR DURING READING
MMDF
JMP .-4
MMRD
AND (7777)
DAC M#MRDB
SAD MMWA1

JMP .+3
LAW 1000 /BLOCK MARK ERROR DURING READING, BME
JMP MMERRX-1
ISZ MMWDC /BLOCK COUNTER
JMP MMSCF1+1           /READ NXT BLOCK
JMP MMSCH6             /GOOD EXIT
MMSCFK, 1              /-0 FOR REVERSE, MODIFIED

/SUM CHECK REVERSE
MMSCR, 0
LAC .-1
DAC MMSCF
LAC (JMS MMVCHR)
DAC MMSCF1
CLC
DAC MMSCFK
LAW 61 /SEARCH REVERSE
JMP MMSCR2

/COMMON ROUTINE FOR PICKING UP VIRGIN TAPE CONSTANTS ETC
MMVCHK,
ADD (-1)
DAC MMAUTO
LAC (JMS MMSCH1)
DAC MMVCH1
LAC I MMAUTO           /BLOCK NUMBER
DAC MMVCH1+1
LAC I MMAUTO           /NUMBER OF BLOCKS
DAC MMWDC
LAC I MMAUTO           /ERROR RETURN
DAC MMVCH1+2
LAC I MMAUTO           /UNIT SELECTION
DAC MMVCH1+4
MMVCH1, JMS MMSCH1     /OR MMSCHR
LAW .      /BLOCK NUMBER, MODIFIED
JMP .      /ERROR RETURN, MODIFIED
JMP MMVCH2             /END RETURN
```

```
0           /UNIT SELECTION
JMP I MMAUTO      /CONTINUE MULTIPROGRAMMING
MMVCH2,
LAC (DISMIS)
DAC MMSCH7
JMP I MMVCHK

MMVCHR,
0
ADD (-1)
DAC MMAUTO
LAC MMVCHR
DAC MMVCHK
LAC (JMS MMSCHR)
JMP MMVCHK+4

START
```

88

MARK TRACK PROGRAM FOR DECTOG

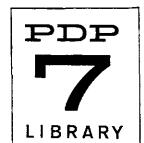
M=320000
B=670000
G=450000
D=070000
GBAR=260000
BBAR=040000
MBAR=510000
E=220000
EBAR=550000

/NOTE THAT THESE MARKS ARE THE REVERSE OF THE WAY THEY ACTUALLY APPEAR ON
/THE TAPE

```
WRMKTR,   JMS CLRFLG
          LAC (NOP)
          DAC WRMB2
EK,       LAM DECIMAL -7143 OCTAL      /NO OF END MARKS
          DAC C1      /NO OF E TO WRITE
          JMS MMWAIT
          LAS        /UNIT SELECTION AND HIGHEST BLOCK NUMBER
          MMSE
          AND (7777)
          CMA
          DAC INIT+2
          HLT
          LAS        /NO OF DATA WORDS PER BLOCK
          AND (7777)
          CMA
          ADD (5)
          DAC wRD
```

LAC (47) /WRITE BLOCK MARK TRACK, FORWARD
MMLC
WRF,
LAC (EBAR)
MMDF
JMP .-1
MMWR
ISZ C1
JMP WRF+1
INIT,
LAC (M)
DAC MM
LAM DECIMAL -577 OCTAL /MODIFIED
DAC C1 /NO OF BLOCKS TO WRITE -1
LAC (JMP WRDATA)
DAC 1
WRM,
MMFF
JMP .+11
LAC MM
AND (7777)
DAC MMWA1
MMRS
DAC MMRSA
LAW 600 /ERW
MMLC
JMP TSTMMS6
LAC MM
MMDF
JMP .-1
MMWR
WRGBAR,
LAC (GBAR)
MMDF
JMP .-1
MMWR
WRBBAR,
LAM -3
DAC C2
LAC (BBAR)
WRBB,
MMDF
JMP .-1
MMWR

ISZ C2
JMP WRBB
WRD,
LAM DECIMAL -251 OCTAL /NO OF DATA FLAGS -1, MODIFIED
DAC C2
ION.
/CALCULATE OBVERSE MARK, 1800US
C01,
LAC MM
RTL
RTL



RTL
DAC WA1
JMS COBV
6
RTR
RTR
RTR
AND (7700)
DAC WA2
C02, LAC WA1
RTL
RTL
RTL
JMS COBV
6
RAL
RTL
RTL
RTL
AND (77)
XOR WA2
XOR (MMBAR)
DAC MMBAR
WRB, JMP . /WAIT FOR DATA TO FINISH
LAM -3
DAC C2
LAC (B)
WRB2, MMDF
JMP .-1
MMWR
ISZ C2
JMP WRB2
WRG, LAC (G)
MMDF
JMP .-1
MMWR
WRMBAR, LAC MMBAR
MMDF
JMP .-1
MMWR
WRMB2, NOP /OR JMP WRMB2+4
LAC (JMP WRMB2+4)
DAC WRMB2
JMP WRM
ISZ MM
ISZ C1
JMP WRM
WREBAR, XCT EK
DAC C1
LAC (E)

```
WRE2,      MMDF
          JMP .-1
          MMWR
          ISZ C1
          JMP WRE2

          CLA
          MMLC
          JMP SUMF1A      /DONE
MM,        @
WA1,       @
WA2,       @
MMBAR,     @
WA3,       @
/WRITE DATA MARKS, 128US, TOTAL=18144US LEFT
WRDATA,    DAC WA3
          LAC (D)
          MMWR
          LAC WA3
          ISZ C2
          SKP
          JMP WRB
          ION
          JMP I @
START

/DBVERSE LOOPS
/VARIABLE COMPLEMENT DBVERSE LOOP, 148US+104US PER CHAR
/FORMAT   JMS OBV
/           6           /NO OF CHARACTERS FROM LEFT TO DBVERSE
COBV,     @
          CMA
          DAC OBV2 /ORIGINAL WORD
COBV1,    LAC I CORV
          CMA
          DAC OBV3 /COUNTER
          ISZ OBV3
          DZM OBV4 /NEW WORD
OBV1,     LAC OBV2
          RAL
          DAC OBV2
          LAC OBV4
          RAR
          DAC OBV4
          ISZ OBV3
          JMP OBV1
          ISZ COBV
          JMP I CORV
```



```
OBV2,    0
OBV3,    0
OBV4,    0
/VARIABLE NON COMPLEMENT OBVERSE LOOP, 176US+104US PER CHARACTER
OBV,
      0
      DAC OBV2
      LAC OBV
      DAC CORV
      JMP CORV1
```

START

TELETYPE ROUTINES WITH OCTAL PRINT LMH 8-8-63 (DLF 12-5-64)
/TURNS INTERRUPT OFF

```
/OCTAL PRINT, WITH ZERO SUPPRESSION
/FORMAT   LAC WD
/        TWORDZ
/        N           /N=NUMBER OF DIGITS TO PRINT FROM LEFT END OF WORD
```

OCTAL

TWORDZ=JMS .

```
      0
      DAC DCPN#UM
      LAC (SZA)
      DAC TWORDZ+17-JMS
      LAC I TWORDZ-JMS
      CMA
      DAC DCPC#NT
      ISZ DCPCNT
      ISZ TWORDZ-JMS
      LAC DCPNUM
      RTL
      RAL
      DAC DCPNUM
      RAL
      AND (7)
      SZA          /MODIFIED
      JMP TWORDZ+25-JMS
      ISZ DCPCNT
      JMP TWORDZ+11-JMS
      TDIGIT
      JMP I TWORDZ-JMS
      DAC DCPD#IG
      LAC (JMP TWORDZ+31-JMS)
      DAC TWORDZ+17-JMS
      LAC DCPDIG
      TDIGIT
      ISZ DCPCNT
      JMP TWORDZ+11-JMS
      JMP I TWORDZ-JMS
```

/OCTAL PRINT, NO ZERO SUPPRESSION
/FORMAT SAME AS TWORDZ

TWORD=JMS .
 @
 DAC DCPNUM
 LAC TWORD-JMS
 DAC TWORDZ-JMS
 LAC (JMP TWORDZ+31-JMS)
 JMP TWORDZ+3-JMS

/TABLE FOR OCTAL TO DECIMAL CONVERSION

DECIMAL

DCPTAB,	100000	10000	1000	100	10	1
OCTAL						

/TELETYPE OUTPUT PACKAGE MODIFIED FOR DECTOG LMH (DLF) 12-5-64

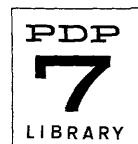
EXT=JMP I-JMS TTAB=10

/TYPE 1 CHARACTER FROM AC BITS 12-17

TY1=JMS .
 @
 RAR
 JMS TY1A
 EXT TY1

/TYPE 1 CHARACTER (5 BIT), LINK INDICATES CASE

TY1A, @
 DAC T#EMY
 AND (37
 SNA
 JMP TY2
 703301
 SKP
 JMP TY1BBB
 LAC OCL
 SPL
 LAC OCU
 SAD OCS
 JMP . 3
 JMS OTY
 DAC OCS



LAC TEMY
JMS OTY
ISZ T#BC
TY2, LAC TEMY
JMP I TY1A

/TYPE 3 CHARACTERS FROM AC 0-5, 6-11,12-17 RESPECTIVELY

TY3=JMS .

Q
JMS RL6
JMS TY1A
JMS RL6
JMS TY1A
JMS RL6
JMS TY1A
EXT TY3

/TYPE A CARRIAGE RETURN, AND LINE FEED

TCR=JMS .

Q
703301
JMP TCRSSS
LAW 215
JMS OTY
LAW 212
JMP TCRRRR
TCRSSS, LAW 2
JMS OTY
LAW 10
TCRRRR, JMS OTY
DZM TBC
EXT TCR

/TELETYPE OUTPUT PACKAGE - PAGE 2

/TYPE A SPACE

TSP=JMS .

Q
LAW 4
703301
SKP



LAW 240
JMS OTY
ISZ TBC
EXT TSP

/TYPE A TABULATION

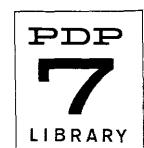
TYT=JMS .
TAB=TYT
 0
 LAC TBC
 ADD (-TTAB-1
 SMA
 JMP .-2
 ADD (1
 SMA
 LAC (-TTAB-1
 ADD (-1
 DAC T#EM
 TSP
 ISZ TEM
 JMP .-2
 EXT TYT

/TYPEWRITER INITIALIZE

TIN=JMS .
 0
 LAC OCL
 DAC OCS
 703301
 JMS OTY
 TCR
 EXT TIN

/TYPE THE DIGIT IN THE AC

TDIGIT=JMS .
 0
 AND (17
 ADD (LAC NCT
 DAC . 1
 XX
 703301
 JMP TDIGT1
 RCL
 JMS RL6
 JMS OTY
 EXT TDIGIT
TDIGT1, TY1
 EXT TDIGIT



/TELETYPE OUTPUT PACKAGE - PAGE 3

/TYPE A STRING OF CHARACTERS

TSR=JMS .

 @
 DAC T#EMY1
 LAC (JMP TSR1
 DAC TY1A 4
 LAC I TEMY1
 TY3
 ISZ TEMY1
 JMP .-3
 TSR1, LAC (JMP TY2
 DAC TY1A 4
 LAC TEMY1
 EXT TSR
/OUTPUT ONE FIVE BIT CHARACTER
OTY, @
 IOF
 DAC TWORD-JMS /SAVE
 LAC NIOT
 703341 /SKIP ON PDP 7
 LAW /COUNTER
 DAC RL6
 LAC TWORD-JMS
 TSF
 SKP
 JMP .+3
 ISZ RL6
 JMP .-4
 TLS
 JMP I 'OTY

/ROTATE LEFT 6

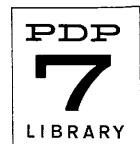
RL6, @
 RTL
 RTL
 RTL
 JMP I RL6

/TABLE OF DIGITS

NCT,	600033	610073	620063	630041
	640025	650003	660053	670071
NIOT,	700031	710007		

/CASE STORAGE

OCU, 33
OCL, 37
OCS, @



/4-7 ADDENDUM DECTOG
TY1BBB, RCR
ADD (LAC MTATAB
DAC . 1
XX
SZL
TY1DDD, JMP TY2-2
JMS RL6
RTL
RTL
JMP TY2-2
/BAUDOT TO ASCII, ASSUMES LETTERS CASE
/ODD RIGHT HALF, EVEN LEFT HALF
MTATAB, 200324
215317
240310
316315
212314
322307
311320
303326
305332
304302
323331
306330
301327
312377
325321
313377

START

REVISED MARK TRACK PROGRAM
RM=040440 /26
RB=444044 /73
RG=404004 /51
RD=444000 /70
RGBAR=144040 /32
RBBAR=004000 /10
RMBAR=400404 /45
REBAR=404404 /55
RE=040040 /22
R25=040404 /25

WFLAGS, 0
MMEF
SKP
JMS WRNVT8 /ERW



MMDF
JMP WFLAGS+1
MMWR
JMP I WFLAGS

RWMKTR, JMS CLRFLG
JMS MMWAIT
LAS /UNIT AND HIGHEST BLOCK NUMBER
MMSE
AND (7777)
CMA
DAC MMWA1 /BLOCK COUNTER
HLT
LAS /NUMBER OF DATA WDS PER BLOCK
AND (7777)
CMA
ADD (5)
DAC MMWA2 /WDS PER BLOCK

LAM DECIMAL -7143 OCTAL
DAC MMWA4
LAW 47 /WRITE TIMING AND MARK TRACK FORWARD
MMLC
LAC (REBAR) /REVERSE END
JMS WFL

RWRM, LAC (R25)
JMS WFLAGS
LAC (RM) /MARK
JMS WFLAGS
LAC (RGBAR) /-G
JMS WFLAGS
LAM -3
DAC MMWA4
LAC (RBBAR) /-B
JMS WFL

LAC MMWA2
DAC MMWA4
LAC (RD) /DATA
JMS WFL
LAM -3
DAC MMWA4
LAC (RB) /FINAL MARKS
JMS WFL
LAC (RG) /G
JMS WFLAGS
LAC (RMBAR) /REVERSE MARK
JMS WFLAGS
LAC (R25)
JMS WFLAGS
ISZ MMWA1

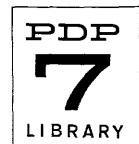
```
JMP RWRM
LAC (DECIMAL -14286 OCTAL)
DAC MMWA4
LAC (RE)
JMS WFL
JMP MM-3           /DONE

WFL,      0
JMS WFLAGS
ISZ MMWA4
JMP .-2
JMP I WFL

/WRITE VIRGIN TAPE FOR MARK TRACK ONLY
WRNVT,      JMS CLRFLG
LAC (1001)
JMS GENBLA
0
DZM MM           /MARK
DZM MMWA1
JMS MMWAIT
LAS             /UNIT
MMSE
DAC SUMF1+4
DAC RVZB+4
JMS REWIND

LAW 41           /SEARCH FWD
MMLC
CLC
DAC MMSUM
LAC (SAD MMWRA-1)
DAC MMAUTO
MMEF
SKP
JMS WRNVT8      /ERW
MMDF
JMP .-4

WRNVT3,      LAW 43           /WRITE FWD
MMLC
/CALCULATE REVERSE MARK
LAC MM
RTR
RTR
AND (700000)
```



DAC MMBAR /BITS 15-17
LAC MM
RTL RTL RTL RTL
RAL
AND (70000)
ADD MMBAR
DAC MMBAR /BITS 12-14
LAC MM
RTL
RAL
AND (7000)
ADD MMBAR
DAC MMBAR /BITS 9-11
LAC MM
RTR
RAR
AND (700)
ADD MMBAR
CMA
DAC MMBAR /BITS 6-8

WRNVT4, LAC I MMAUTO
MMEF
SKP
JMS WRNVT8 /ERW
MMDF
JMP WRNVT5
MMWR
MMRD
XCT MMAUTO
JMP .+5
MMRS
DAC MMRSA
LAW 1400 /BUF
JMP WRM+10
ADD MMSUM
DAC MMSUM
JMP WRNVT4

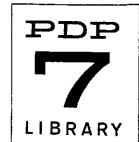
WRNVT5, MMBF
JMP WRNVT4+1
LAC MMSUM
CMA
DAC MMWC
MMWR
LAW 46 /WRITE THRU ENDS

MMLC
CLC
DAC MMSUM
LAC (SAD MMWRA-1)
DAC MMAUTO
LAM -1
DAC MMWA4
CLA
JMS WFL /WRITE -L AND G
LAC MMBAR /REVERSE MARK
JMS WFLAGS
CLC /REVERSE 25
JMS WFLAGS /FORWARD 25
CLA
JMS WFLAGS
ISZ MM /UPDATE MARK
LAC MM
DAC MMWA1
JMS WFLAGS /MARK
CLA
JMS WFLAGS /-G
WRNVT6, MMEF
JMP .+6
MMRS
AND (40000)
SZA
JMP WRNVT7 /END OF TAPE
JMS WRNVT8
MMDF
JMP WRNVT6
JMP WRNVT3

/AUTOMATIC CHECK SUM
WRNVT7, CLA
MMLC
LAM -1
TAD MM
DAC K1 /STARTING BLOCK
CMA
ADD (1)
DAC SUMF1+2
LAC (JMS MMSCR)
DAC SUMF1
LAC (JMP RVZB)
DAC SUMF1A
JMP SUMF1

WRNVT8, 0
JMP WRM+2

START



```

/STOP ADJUSTMENT PROGRAM
/IF ACSO=0, GO FOR 1 SECOND, OTHERWISE GO FOR .12 SECONDS
STAP,      JMS CLRFLG
           JMS STAP3
           DZM C1           /CLEAR DIRECTION
           JMS MMWAIT
           LAS
           MMSE
           LAW 40           /MOVE FWD
           XOR C1           /GET DIRECTION
           MMLC
           LAS
           RAL
           LAC STAPK
           SPL
STAP2,     LAM DECIMAL -3750 OCTAL      /0.12 SECONDS
           JMS TIMER
           CLA
           MMLC
           LAC C1
           XOR (20)
           DAC C1           /CHANGE DIRECTION
           LAC STAPK /STOP FOR 1 SECOND
           JMS TIMER
           JMP STAP+3
STAPK,    DECIMAL -31250 OCTAL      /1 SECOND
STAPK2,   DECIMAL -31250 OCTAL      /1 SECOND, PDP-4
STAPK3,   DECIMAL 119287 OCTAL     /1 SECOND, PDP-7
STAPK4,   DECIMAL -17143 OCTAL     /.12 SECONDS, PDP-7

/TIMING LOOP, 32 MICROSECONDS PER COUNT FOR PDP-4, 7 MS FOR PDP-7
/FORMAT   LAC (-N) /WHERE 'N=MICROSECONDS OVER 32
/
TIMER,    JMS TIMER
          0
          DAC C2
          ISZ I .-1
          JMP .-1
          JMP I TIMER

STAP3,    0
          SKP7
          JMP STAP4
          LAC (LAC STAPK4) /FOR PDP-7
          DAC STAP2
          LAC STAPK3
          DAC STAPK
          JMP I STAP3
STAP4,    LAM DECIMAL -3750 OCTAL      /FOR PDP-4
          DAC STAP2
          LAC STAPK2
          JMP STAP4-2

```

/RECORD BLOCK MARK NUMBERS

RBMN, JMS CLRFLG
JMS MMWAIT
LAS
MMSE
LAW MMWRA-1
DAC MMAUTO
LAM -3777
DAC WA1
JMS REWIND
LAW 41 /SEARCH FWD

RBMN1, MMLC
MMEF
JMP .+6
MMRS
AND (40000) /END OF TAPE
SAD (40000)
JMP RBMN2
JMP RBMN1-2
MMDF
JMP RBMN1
MMRD
DAC I MMAUTO
ISZ WA1
JMP RBMN1
CLA
MMLC
JMP SUMF1A /TYPE END

RBMN2, LAM
DAC I MMAUTO
ISZ WA1
JMP .-2
JMP RBMN2-3

/RETURN FROM SUM CHECK TO WRITE FIRST BLOCK ZERO NUMBER

RVZB, JMS MMSCHR
LAW 1 /BLOCK NUMBER 1
JMP RVZB /ERROR RETURN, SEARCH AGAIN
JMP .+3 /DONE RETURN
Ø /UNIT
JMP . /WAIT FOR SEARCH TO BE COMPLETED
MMDF
JMP .-1
MMRD
SZA
JMP RVZB5 /NON-ZERO BLOCK FOUND
LAW 62 /READ REVERSE

RVZB2,
 MMLC
 MMDF
 JMP .+3
 MMRD
 JMP RVZB2
 MMBF
 JMP RVZB2
 MMRD
 LAW 66 //WRITE ALL REVERSE
 MMLC
RVZB3,
 CLA
 MMEF
 SKP
 JMP RVZB4
 MMDF
 JMP RVZB3+1
 MMWR
 JMP RVZB3
RVZB4,
 MMRS
 AND (40000)
 SAD (40000)
 JMP .+3
 DZM MM
 JMS WRNVT8
 LAC (JMS TYPEND)
 DAC SUMF1A
 JMP SUMF1A
//NON-ZERO BLOCK FOUND
RVZB5,
 CLA
 MMLC
 TIN
 LAC (FLEX NZR)
 TY3
 JMP X1

//REWIND DECTAPE UNIT
REWIND,
 0
 LAW 60 //REWIND
 MMLC
 MMEF
 JMP .-1
 MMRS
 AND (40000) //END OF TAPE
 SNA
 JMP REWIND+1 //IGNORE ERROR
 JMP I REWIND
START