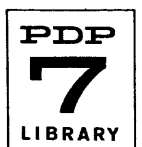


PDP-7 PROGRAM LIBRARY

- 1. IDENTIFICATION
- 1.1 Digital-7-21-IO-Sym, FB DECTRIEVE, PDP-7
- 1.2 Leonard M. Hantman - DEC
- 1.3 12-22-64



2. ABSTRACT

2.1 Purpose

To allow the programmer to save areas of memory on DECtape, and allow quick retrieval of such information, using the toggle switches on the PDP-7.

3. REQUIREMENTS

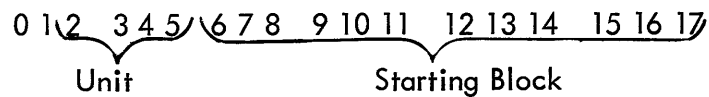
3.3 Equipment

Paper Tape Reader, Teleprinter, DECtapes

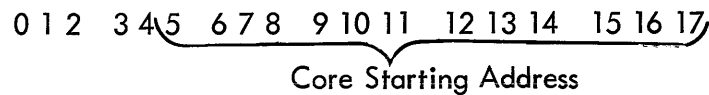
4. USAGE

A) To Store Information

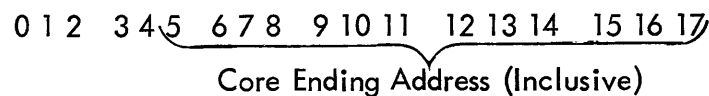
- 1) Set ACS as follows:



- 2) Start at 6001 (or 16001).
3) When HLT occurs set ACS as follows:*



- 4) Press CONTINUE.
5) When HLT occurs set ACS as follows:*



Make sure area requested goes from the lower part of the information to the highest part even if there is an unused portion in between. If too much tape is being wasted, the data can be broken up into smaller groups by storing each small area separately.

If the number of words to be stored does not constitute an integral number of blocks, the last block will be filled with +0.

- 6) Press CONTINUE.

*Be careful that bit 5 is not left set inadvertently from the unit selection in step 1.




```
WRITE=JMS .  
0  
lac i .-1  
dac wrt1  
dac tpsb1  
dac rd2  
isz write-jms  
lac i write-jms  
dac wrt1+1  
isz write-jms  
lac i write-jms
```

- 4) Change the instruction at rd1+6 to lac i write-jms.
- 5) Change the two instructions beginning at wrt1a+3 to:

```
isz write-jms  
jmp i write-jms
```

- 6) Note the following limitations on using the changes as described:

- a) The DECTRIEVE area itself cannot be stored as the program return saved on the tape will be the one desired for writing but not necessarily for reading.

- b) In case of an error, the program will type an error message and the return to the program at the same place as if the transfer was completed correctly. This situation can be handled by inserting instructions before errwa+9 which set a program flag, inserting instructions at READ and WRITE which clear the flag, and checking for the setting of the flag upon returning from DECTRIEVE. To simply HALT if an error occurs, change the instruction at errwa+9 to a HLT.

- c) To eliminate the normal completion messages (but not the error messages), change the instruction at wrt1b+2 to a jmp wrt1a+1.

- 7) The reasons for using DECTRIEVE as a subroutine should be quite clear and specific to the user. If it is being used simply to transfer data, a lot of effort and memory room can be saved by simply using the normal DECTape Subroutines (Digital-7-22-IO).

6. DESCRIPTION

To store data, the user indicates the DECTape block number and core starting and ending addresses of the area to be saved. The routine will store the data on the indicated blocks together with four words of control information used for retrieval. When completed, appropriate messages are typed which can be used to verify the data upon retrieval. All information written is checked by re-reading the data and accumulating the checksums. The information is not stored in memory when sum checking.

To retrieve the information, the user need only indicate the starting block where the information was stored. The control information on the tape will supply enough data to store the information in the correct memory registers. Upon completion, a message will be typed whose pertinent data should be an exact duplicate of the data typed when the information was stored.

The program occupies approximately 1260₈ words of storage and versions are available beginning at either 6000 or 16000 in memory and for either the first or second DECTape controls. Either version can be used with a 4K or 8K machine.

If any errors occur, they will be typed as in DECTOG (q.v.). With the exception of Register 0, any portion of memory including the DECTRIEVE area may be saved. Register 0 and 10 are destroyed by the program. Register 1 is saved and restored after the program is run and is written with the control information. Therefore, when the information is retrieved, Register 1 will appear as it did when the information was stored. In calculating the number of blocks a given area will occupy, be sure to include the space automatically occupied by the four control words.

9. PROGRAM

9.4 Listing

```

          DECTRIEVE LOWER MEMORY
/READ AND WRITE PROGRAMS WITH DECTAPE
MSUMS=0
6000/
READ,      JMP RD1
WRITE,     LAS                      /UNIT AND BLOCK NUMBER
          DAC WRT1                   /UNIT
          DAC T#PSRL
          DAC RD2
          HLT
          LAS                      /CORE START
          DAC WRT1+1
          HLT
          LAS                      /CORE END
          DAC WRT1+2
          LAC (LAC RD2+1)
          DAC MMWRS+3
          LAC (FLEX RD )
          DAC RD#WRL
          LAC (DAC I MMAUTO)
          DAC MMRD3
          DAC M#MDAC
          LAC 1
          DAC MM#SV1
          JMS RDSET
          JMS CLRFLG

```

```

WRT1,      JMS MMWRS
            LAC .+2                /BLOCK NUMBER
            JMP ERR                /ERROR RETURN
            Ø                       /UNIT
            Ø                       /CORE START
            Ø                       /CORE END

            LAC MMWA1
            ISZ MMDONE
            JMP .-2
            LAC (FLEX WR )
            DAC RDWRL
            LAC (CLL)
            DAC MMDAC
            JMP RD1A

WRT1B,     CLA
            MMLC
            TIN
            LAC RDWRL
            TY3
            LAC TPSRL
            AND (7777)
            JMS TWZ6
            LAW CHAR R-
            TY1
            LAM
            TAD MMWA1
            JMS TWZ6
            TSP

WRT1A,     LAC MSUMS
            JMS TWZ6
            LAC MMSV1
            DAC 1
            HLT
            JMP .-1

WRT2,     LAC (FLEX KIE)          /IDENTIFICATION
            MMWR
            JMS WRSUM
            LAC (LAC I MMAUTO)
            DAC MMWR3
            JMS WAIT
            LAC MMAUTO          /CORE START-1
            MMWR
            JMS WRSUM
            JMS WAIT

```

```

LAC MMWDC /WORD COUNTER
MMWR
JMS WRSUM
JMS WAIT
LAC MMSV1
MMWR
JMS WRSUM
JMP MMWR2

WRSUM,  0
ADD MMSUM
DAC MMSUM
JMP I WRSUM

WAIT,   0
MMEF
SKP
JMP MMWR2+2      /ERROR
MMDF
JMP .-4
JMP I WAIT

RD1,    LAC (DAC I MMAUTO)
DAC MMDAC
LAC (FLEX RD )
DAC RDWRL
LAC 1
DAC MMSV1
LAS      /UNIT AND BLOCK NUMBER
DAC RD2
DAC TPSBL

RD1A,   LAC (LAC RD2+2)
DAC MMRDS+5
JMS RDSET
JMS CLRFLG

JMS MMRDS
LAC .+2      /BLOCK NUMBER
JMP ERR      /ERROR

RD2,    0      /UNIT
JMP WRT2     /CORE START, NOT ACTUALLY USED
JMP RD3      /CORE END, NOT ACTUALLY USED

LAC MMWA1
JMP .-1

```

```
RD3,      SAD (FLEX KIE)
           JMP RD4
           TIN
           LAC (FLEX NG )
           TY3
           LAC MMWA1
           JMP WRT1A

RD4,      JMS WRSUM
           LAC MMDAC
           DAC MMRD3
           JMS WAITR
           MMRD
           DAC MMAUTO
           JMS WRSUM
           JMS WAITR
           MMRD
           DAC MMWDC
           JMS WRSUM
           JMS WAITR
           MMRD
           DAC MMSV1
           JMS WRSUM
           DZM MSUMS
           JMP MMRD2

RD5,      DAC WAITR
           ADD MSUMS
           DAC MSUMS
           LAC WAITR
           ADD MMSUM
           JMP MMRD4+4

WAITR,    Ø
           MMEF
           SKP
           JMP MMRD1+2
           MMDF
           JMP .-4
           JMP I WAITR

ERROR,    Ø
           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)

ERRWA,      DAC .+1
            LAC ERRTAB          /MODIFIED
            TY3
            TSP
            LAC MMWA1
            JMS TWZ6
            TSP
            LAC MMRSA
            TWORD
            3
            JMP I ERROR

ERRTAB,    FLEX CMP  FLEX FMT  FLEX NTF
            FLEX ERS  FLEX ERR  FLEX SUM
            FLEX ERW  FLEX BMW  FLEX BMC
            FLEX INT  FLEX FLC  FLEX NFL
            FLEX BUF  FLEX NWR

ERR,       JMS ERROR
            JMP WRT1A+1

TWZ6,     0
            TWORDZ
            6
            JMP I TWZ6

CLRFLG,   0
            IOF          DCF          CRRB
            CPCF        LPCF        LSCF
            700102     PCF          KRB
            TCF         MSI         CLOF
            CLA         707604     701604    /CLEAR BOTH DECTAPE CONTROLS
            LAC (JMP INTERR)
            DAC 1
            JMP I CLRFLG

```

```
INTERR,   DAC A#CSAVE
          MMEF
          SKP
          JMP MMERR
          MMDF
          SKP
          JMP MMDATA
          IORS
          HLT
          JMP .-1
```

```
DISMIS=JMP .
          LAC Ø
          RAL
          LAC ACSAVE
          ION
          JMP I Ø
```

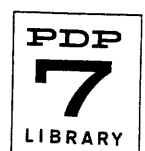
MMAUTO=1Ø

```
RDSET,   Ø
          LAC RD4+1
          DAC MMRD4+11
          LAC (JMP RD5)
          DAC MMRD4+3
          LAC (JMP WRT1B)
          DAC MMRD4+14
          JMP I RDSET
```

/PDP-7 DECTAPE SUBROUTINES, CONTROL 1, LMH 12-22-64
/PDP-7 DECTAPE SEARCH SUBROUTINE
/DISMIS MUST BE DEFINED AS JMP TO DISMISS INTERRUPT ROUTINE

```
MMWR=7Ø75Ø4  
MMLC=7Ø76Ø4  
MMSE=7Ø7644  
MMRS=7Ø7612  
MMDF=7Ø75Ø1  
MMBF=7Ø76Ø1  
MMEF=7Ø7541  
MMRD=7Ø7512  
SKP7=7Ø3341
```

```
/FORMAT  JMS MMSCH /OR MMSCH1 OR MMSCHR  
/        LAW B      /OR LAC (B), BLOCK NUMBER  
/        JMP X      /ERROR RETURN  
/        JMP Y      /SEARCH COMPLETED RETURN  
/        ZZØØØØ    /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#MBLKM /BLOCK TO SEARCH FOR
SNA
JMP MMSCH4
ADD MMEK
SMA
JMP MMSCH4 /FORMAT ERROR
LAM -7
DAC M#MSUM /CHG OF DIRECTION COUNTER
LAC I MMSCH1 /ERROR RETURN
DAC MMERRX
ISZ MMSCH1
LAC I MMSCH1 /COMPLETION RETURN
DAC MMSCH7
ISZ MMSCH1
JMS MMWAIT /CHECK IF DELAY IS NECESSARY
LAC J 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#MWA2      /BLOCK TO LOOK FOR IN THIS DIRECTION
        DZM M#MDONE
MMSAVE, NOP          /OR DISMIS
        LAC (DISMIS)
        DAC MMSAVE
        JMP I MMSCH1      /CONTINUE MULTI-PROGRAMMING
MMREV,  LAW 61
        DAC MMWA3
        MMLC
        LAC (SPA)
        DAC MMSCH2
        LAC MMRLK
        TAD MMSRK
        JMP MMSAVE-2
MMERR,  MMRS
        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, Ø

```

      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#MWA4
      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
      JMP I MMAUTO         /MULTIPROCESS WITH MAIN PROGRAM
MMCHK2, LAC MMWA4
      DAC MMAUTO
      LAC (DISMIS)
      DAC MMSCH7
      JMP I MMCHK

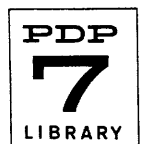
```

/DECTAPE READ SUBROUTINE

```

/FORMAT JMS MMRDS
/      LAW B                 /OR LAC (B), BLOCK NUMBER
/      JMP X                 /ERROR RETURN
/      ZZØØØØ             /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              /ERROR FLAG DURING READING
             JMP MMERRX-1
             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
MMRD4,      MMBF
             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 MMWRS
/         LAW B       /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,    0
          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
MMWR3,    JMP MMWR4
          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
          MMEF
          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
```

```
/TELETYPE ROUTINES WITH OCTAL PRINT, LMH 8-8-63
/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 .
Ø
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 9/29/64 LMH (DF)

EXT=JMP I-JMS TTAB=10

/TYPE 1 CHARACTER FROM AC BITS 12-17

TY1=JMS .
Ø
PAR
JMS TY1A
EXT TY1

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

TY1A, Ø
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 .  
  0  
    JMS RL6  
    JMS TY1A  
    JMS RL6  
    JMS TY1A  
    JMS RL6  
    JMS TY1A  
    EXT TY3
```

/TYPE A CARRIAGE RETURN, AND LINE FEED

```
TCR=JMS .  
  0  
    703301  
    SKP  
    JMP TCRRRR  
    LAW 2  
    JMS OTY  
TCRSSS,     LAW 10  
            JMS OTY  
            DZM TBC  
            EXT TCR  
TCRRRR,     LAW 215  
            JMS OTY  
            LAW 212
```

/TELETYPE OUTPUT PACKAGE - PAGE 2

/TYPE A SPACE

```
TSP=JMS .  
  0  
    LAW 4  
    703301  
    SKP  
    LAW 240  
    JMS OTY  
    ISZ TBC  
    EXT TSP
```

/TYPE A TABULATION

TYT=JMS .
TAB=TYT

```
Ø
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 .

```
Ø
LAC OCL
DAC OCS
703301
JMS OTY
TCR
EXT TIN
```

/TYPE THE DIGIT IN THE AC

TDIGIT=JMS .

```
Ø
AND (17
ADD (LAC NCT
DAC . 1
XX
TY1
EXT TDIGIT
```

/TELETYPE OUTPUT PACKAGE - PAGE 3

/TYPE A STRING OF CHARACTERS

TSR=JMS .

```
Ø
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, 0
IOF
DAC TWORD-JMS /SAVE
CLA
703341
LAW /COUNTER
DAC RL6
LAC TWORD-JMS
TSF
SKP
JMP .+3
ISZ RL6
JMP .-4
TLS
JMP I OTY

/ROTATE LEFT 6

RL6, 0
RTL
RTL
RTL
JMP I RL6

/TABLE OF DIGITS

NCT, 33 73 63 41
25 3 53 71
31 7

/CASE STORAGE

OCU, 33
OCL, 37
OCS, 0
```

```
/PDP-4/7 ADDENDUM
TY1BBB,  ADD (LAC BTATAB-1
          DAC .+1
          XX
          SZL
          JMP TY1CCC
TY1DDD,  JMS 0TY
          JMP TY2-1
TY1CCC,  JMS RL6
          RTL
          RTL
          JMP TY1DDD
BTATAB,  265324
          215215
          271317
          240240
          243310
          254316
          256315
          212212
          251314
          264322
          246307
          270311
          260320
          272303
          273326
          263305
          242332
          244304
          277302
          211323
          266331
          241306
          257330
          255301
          262327
          247312
          377377
          267325
          261321
          250313
          377377

START
```

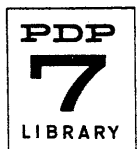
```

R←L
DECTRIEVE, UPPER MEMORY
/READ AND WRITE PROGRAMS WITH DECTAPE
MSUMS=0
16000/
READ,      JMP RD1
WRITE,     LAS           /UNIT AND BLOCK NUMBER
           DAC WRT1      /UNIT
           DAC T#PSRL
           DAC RD2
           HLT
           LAS           /CORE START
           DAC WRT1+1
           HLT
           LAS           /CORE END
           DAC WRT1+2
           LAC (LAC RD2+1)
           DAC MMWRS+3
           LAC (FLEX RD )
           DAC RD#WRL
           LAC (DAC I MMAUTO)
           DAC MMRD3
           DAC M#MDAC
           LAC 1
           DAC MM#SV1
           JMS RDSET
           JMS CLRFLG

           JMS MMWRS
           LAC .+2       /BLOCK NUMBER
           JMP ERR      /ERROR RETURN
WRT1,     0           /UNIT
           0           /CORE START
           0           /CORE END

           LAC MMWA1
           ISZ MMDONE
           JMP .-2
           LAC (FLEX WR )
           DAC RDWRL
           LAC (CLL)
           DAC MMDAC
           JMP RD1A

```



```
WRT1B,   CLA
          MMLC
          TIN
          LAC RDWRL
          TY3
          LAC TPSBL
          AND (7777)
          JMS TWZ6
          LAW CHAR R-
          TY1
          LAM
          TAD MMWA1
          JMS TWZ6
          TSP
          LAC MSUMS
WRT1A,   JMS TWZ6
          LAC MMSV1
          DAC 1
          HLT
          JMP .-1

WRT2,   LAC (FLEX KIE)      /IDENTIFICATION
          MMWR
          JMS WRSUM
          LAC (LAC I MMAUTO)
          DAC MMWR3
          JMS WAIT
          LAC MMAUTO        /CORE START-1
          MMWR
          JMS WRSUM
          JMS WAIT
          LAC MMWDC /WORD COUNTER
          MMWR
          JMS WRSUM
          JMS WAIT
          LAC MMSV1
          MMWR
          JMS WRSUM
          JMP MMWR2

WRSUM,   @
          ADD MMSUM
          DAC MMSUM
          JMP I WRSUM

WAIT,    @
          MMEF
          SKP
          JMP MMWR2+2      /ERROR
          MMDF
          JMP .-4
          JMP I WAIT
```



```

RD1,      LAC (DAC I MMAUTO)
          DAC MMDAC
          LAC (FLEX RD )
          DAC RDWRL
          LAC 1
          DAC MMSV1
          LAS /UNIT AND BLOCK NUMBER
          DAC RD2
          DAC TPSBL

RD1A,     LAC (LAC RD2+2)
          DAC MMRDS+5
          JMS RDSET
          JMS CLRFLG

          JMS MMRDS
          LAC .+2 /BLOCK NUMBER
          JMP ERR /ERROR
RD2,     0 /UNIT
          JMP WRT2 /CORE START, NOT ACTUALLY USED
          JMP RD3 /CORE END, NOT ACTUALLY USED

          LAC MMWA1
          JMP .-1

RD3,     SAD (FLEX KIE)
          JMP RD4
          TIN
          LAC (FLEX NG )
          TY3
          LAC MMWA1
          JMP WRT1A

RD4,     JMS WRSUM
          LAC MMDAC
          DAC MMRD3
          JMS WAITR
          MMRD
          DAC MMAUTO
          JMS WRSUM
          JMS WAITR
          MMRD
          DAC MMWDC
          JMS WRSUM
          JMS WAITR
          MMRD
          DAC MMSV1
          JMS WRSUM
          DZM MSUMS
          JMP MMRD

```

```

RD5,      DAC WAITR
          ADD MSUMS
          DAC MSUMS
          LAC WAITR
          ADD MMSUM
          JMP MMRD4+4

WAITR,    Ø
          MMEF
          SKP
          JMP MMRD1+2
          MMDF
          JMP .-4
          JMP I WAITR

ERROR,    Ø
          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)

ERRWA,    DAC .+1
          LAC ERRTAB          /MODIFIED
          TY3
          TSP
          LAC MMWA1
          JMS TWZ6
          TSP
          LAC MMRSA
          TWORD
          3
          JMP I ERROR

ERRTAB,   FLEX CMP   FLEX FMT   FLEX NTF
          FLEX ERS   FLEX ERR   FLEX SUM
          FLEX ERW   FLEX BMW   FLEX BMC
          FLEX INT   FLEX FLC   FLEX NFL
          FLEX BUF   FLEX NWR

```

```
ERR,      JMS ERROR
          JMP WRT1A+1

TWZ6,     0
          TWORDZ
          6
          JMP I TWZ6

CLRFLG,   0
          IDF          DCF          CRRB
          CPCF         LPCF         LSCF
          700102      PCF          KRB
          TCF          MSI          CLOF
          CLA          707604      701604      /CLEAR BOTH DECTAPE CONTROLS
          LAC (JMP INTERR)
          DAC 1
          JMP I CLRFLG

INTERR,   DAC A#CSAVE
          MMEF
          SKP
          JMP MMERR
          MMDF
          SKP
          JMP MMDATA
          IORS
          HLT
          JMP .-1

DISMIS=JMP .
          LAC 0
          RAL
          LAC ACSAVE
          ION
          JMP I 0

MMAUTO=10

RDSET,    0
          LAC RD4+1
          DAC MMRD4+11
          LAC (JMP RD5)
          DAC MMRD4+3
          LAC (JMP WRT1B)
          DAC MMRD4+14
          JMP I RDSET
```