3 4456 0555905 3
OAK RIDGE NATIONAL LABORATORY LIBRARIES

CENTRAL RESEARCH LIBRARY

ORNL-TM-4582

Cy 45

DECSYSTEM10-PDP-8 DATA TRANSMISSION VIA DECTAPE

L. W. Gilley

OAK RIDGE NATIONAL LABORATORY
CENTRAL RESEARCH LIBRARY
DOCUMENT COLLECTION
LIBRARY LOAN COPY
DO NOT TRANSFER TO ANOTHER PERSON
If you wish someone else to see this document, send in name with document and the library will arrange a loan.
UCN-7969
(3 3-67)

OAK RIDGE NATIONAL LABORATORY
OPERATED BY UNION CARBIDE CORPORATION • FOR THE U.S. ATOMIC ENERGY COMMISSION

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

ORNL-TM-4582

Contract No. W-7405-eng-26

COMPUTER SCIENCES DIVISION

DECSYSTEM10-PDP-8 DATA TRANSMISSION VIA DECTAPE

L. W. Gilley

JUNE 1974

NOTICE This document contains information of a preliminary nature and was prepared primarily for internal use at the Oak Ridge National Laboratory. It is subject to revision or correction and therefore does not represent a final report.

OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37830
operated by
UNION CARBIDE CORPORATION
for the
U.S. ATOMIC ENERGY COMMISSION

3 4456 0555905 3

OAK RIDGE NATIONAL LABORATORY LIBRARIES

CONTENTS

		Page
Abstract		1
I.	Program Description	1
II.	Program Operation	3
III.	Program Listing	7

DECSYSTEM10-PDP-8 DATA TRANSMISSION VIA DECTAPE

L. W. Gilley

ABSTRACT

DT8RW is a Decsystem10 Assembly Language program which will transmit binary data between a standard formatted PDP-8 Dectape and the user's disk area on the Decsystem10, with certain restrictions. Options permit choice of read or write, choice of the number of Dectape blocks to be read or read to a user-specified EOF mark, and choice of starting block number other than 0, which is not permitted.

I. PROGRAM DESCRIPTION

A Decsystem10 Assembly Language routine, DT8RW, has been developed which will transmit data between a user's disk area and a standard formatted PDP-8 Dectape (in or out), with certain restrictions.

The standard format for a PDP-8 Dectape has 129 12-bit bytes (words) per block, a block being 43 Decsystem10 words. Most standard PDP-8 handlers utilize only the first 128 bytes of the block, corresponding to one "page" of core, and read and write sequentially forward starting at a designated block number, which includes block 0. DT8RW transmits data using the dump mode of the IN and OUT UUO's which permits control of the number of bytes (words) transmitted. The program is designed so that on input DT8RW reads a maximum of 43 Decsystem10 words (129 12-bit bytes) per block but transfers only the first 128 12-bit bytes to a disk output buffer. If possible DT8RW assembles three Dectape blocks of data before outputting to disk because this makes one disk

block (3 X 128 = 384 12-bit bytes = 128 36-bit bytes). On output DT8RW inputs blocks of data from disk and stores them in an output buffer such that data is not stored in the 129th and 258th byte positions. Hence, the information in one disk block is stored in a buffer that can be output on three PDP-8 Dectape blocks with the data occupying the first 128 12-bit bytes of each block.

The main (optional) features of the program are

(a) It will write a file of data from Decsystem10 disk to standard formatted PDP-8 Dectape starting at block number 1 or greater and insert as the last 12-bit byte a 12-bit user-designated EOF mark. Block 0 cannot be used since the Decsystem10 will not read or write on block 0 going forward.

(b) It will read a file of data from PDP-8 Dectape starting with a user-designated block number and reading to a user-designated EOF mark.

(c) It will read a user-designated number of PDP-8 blocks (rather than read to an EOF mark) and store the data as a file on the user's disk. If the number of PDP-8 blocks is not a multiple of 3, all 12-bit bytes beyond actual data in the last Decsystem10 block will be zeroed.

Part II describes operation of the program.

II. PROGRAM OPERATION

Utilization of DT8RW requires inputting a number of parameters to select the optional features; therefore, the input sequence is in conversational mode for convenience. In the following text three examples of using the routine are given: first, writing a user's file of data from disk to PDP-8 Dectape; second, reading from Dectape to an EOF mark; and third, reading a designated number of blocks from Dectape. To utilize the program the user must first mount his PDP-8 Dectape using the MOUNT command. It is assumed in what follows that the user's project programmer number is 1234,56.

Execution is begun with the command

```
.RUN DSKC:DT8RW[5,1115]
```

and the routine begins execution and responds by typing

```
NAME.EXT[PPN] =
```

and waits for user input. The file name and PPN that should be input in response will be either the file name and user's disk area from which or into which data is to be transmitted. Given the above is completed, the following optional sequences can be chosen, designated examples 1, 2, and 3. User input is everything to the right of the equal sign or right parenthesis.

Example 1.

```
NAME.EXT[PPN] = FILE.BIN[1234,56]↓
BLOCK #1 = 1↓
READ? (Y OR N) ↓
EOF ON TAPE = 7777↓
```

In the first command string the file name is the name of the disk file to be written on Dectape (the name itself is not written on tape - only data). The comma can be replaced with a slash (/). The brackets are necessary. When the user types RETURN at the end of the first command, that command is interpreted; and, if there are no errors, the octal number of the first block is called for. When the block number and RETURN are input, the read or write option is called for. Everything through the right parenthesis is printed, and in this example the user types N↓ . Then the routine calls for a 12 bit EOF byte. This must be a 4-digit or smaller octal number. When the RETURN following the EOF has been input, if there is no command error, program execution begins; and when the file has been completely output, the routine types DONE and exits (the Monitor then types EXIT). If a command error is found in any command, the routine is restarted and types NAME.EXT[PPN]=. Also, one can restart the program during the input of any command by inputting $ (not ESC) in the command string followed by RETURN.

Example 2.

```
NAME.EXT[PPN] = FILE.BIN[1234,56] ↓
BLOCK #1 = 1 ↓
READ (Y or N) Y ↓
# BLOCKS ON TAPE = ↓
EOF ON TAPE = 7777 ↓
```

The first two commands are described in the first example. Here the user wants to read an entire file from tape, so in the third command string he types Y and RETURN after the right parenthesis. After processing this command, the routine types # BLOCKS ON TAPE =. Here the user has the choice of reading a specified number of blocks; or, by inputting

RETURN only as shown, he can input an EOF byte as shown in the last command string and the routine will read data until a 12-bit byte is found with a bit pattern identical to the four-digit octal number input. Any unique bit pattern is suitable.

Example 3.

```
NAME.EXT[PPN] = FILE.BIN[1234,56]↓
BLOCK #1 = 1↓
READ? (Y or N) Y↓
# BLOCKS ON TAPE = 12↓
```

In this example it is desired to read 12 PDP-8 blocks of data into a disk file named FILE.BIN in the user's area. The routine proceeds after inputting the number of blocks because no search is to be made for an EOF mark.

The following are error messages that may be output:

```
COMMAND ERROR
TOO MANY DIGITS IN FIELD
OPEN UUO ERR
DATA MISSED OR PARITY ERR ON DT INPUT
ERR ON OUTPUT TOO DSK
DT OUTPUT ERR
DT INPUT ERR
INPUT ERR ON DSK
```

If command errors are made, the routine will generally output COMMAND ERROR and restart with the first command. If a numeric field, such as EOF mark, contains too many digits, the routine prints TOO MANY DIGITS IN FIELD and restarts at the first command. The third message above will not occur unless the routine is "clobbered" in some manner. The remaining messages concern some kind of hardware error.

III. PROGRAM LISTING

```
        TITLE DT8RW

;THIS ROUTINE READS FROM AND WRITES ON A STANDARD FORMATTED PDP-8
;DECTAPE, I.E. 43 36-BIT WORDS PER BLOCK. ON INPUT, ONLY 128 12-BIT
;BYTES PER DECTAPE BLOCK ARE STORED ON DSK. ON OUTPUT, DATA FROM A DSK
;BLOCK ARE OUTPUT TO THE 128 12-BIT BYTES PER PDP-8 BLOCK.

START:  CALLI   0       ;STOP I/O
        OPEN    4,SPEC4 ;TTY INPUT
NPUT:   JUMPA   ERR8
        INBUF   4,1     ;1 BUF FOR INPUT
        OPEN    3,SPEC3 ;TTY OUTPUT
        JUMPA   ERR8
        OUTBUF  3,1

RSNPPN: OUTSTR  MSG1
INNPPN: SETZM   RWFLG   ;ZERO READ WRITE FLAG
        SETZM   FILEN   ;NAME
        SETZM   PPNFLG
        SETZM   CONFLG
        SETZ    12,     ;PERIOD COUNTER
        SETZ    7,      ;CHAR COUNTER
        SETZ    3,      ;WILL CONTAIN CONVERTED BIN #
        MOVEI   6,6     ;SET CHAR LIMIT
        MOVE    2,FNP   ;GET FILE NAME POINTER
        MOVEM   2,FNBP

        CLRBFI
INCHR:  INCHWL  11      ;GET A CHAR
        CAIN    11,44   ;$? START OVER.
        JUMPA   RSNPPN
        CAIN    11,40   ;SPACE?
        JUMPA   INCHR   ;IGNOR SPACE
        CAIN    11,15   ;CR?
        JUMPA   CROUT
        CAIN    11,56   ;PERIOD?
        JUMPA   SPER    ;SERVICE PERIOD
        CAIN    11,133  ;LEFT BRACKET?
        JUMPA   SLBRAC
        CAIN    11,135
        JUMPA   SRBRAC  ;RIGHT BRACKET?
        CAIN    11,54
        JUMPA   SCOMA   ;SERVICE COMMA
        CAIN    11,57   ;/?
        JUMPA   SCOMA
        AOJ     7,
        CAMLE   7,6
        JUMPA   ERR10
        MOVE    1,CONFLG
        CAIL    1,1     ;IF FLAG IS 0 SKIP
        JUMPA   OCTCON  ;CONVERT TO OCTAL
SBTCON: CAIG    11,37   ;LIMIT CHARS TO UPPER CASE
        JUMPA   INCHR
        CAIL    11,140
        JUMPA   INCHR
        ADDI    11,40   ;CON TO 6-BIT
        ANDI    11,77   ;TRUNCATE TO 6-BIT
        IDPB    11,FNBP ;DEPOSIT IN NEXT BYTE
        JUMPA   INCHR
```

```
SPER:   AOJ     12,       ;PERIOD COUNTER
        CAIG    12,1
        JUMPA   .+2
        JUMPA   RSNPPN    ;TOO MANY PER. RESTART.
        CAIG    7,1       ;MUST HAVE AT LEAST 1 CHAR IN NAME
        JUMPA   RSNPPN    ;CHAR IN
        SETZ    7,
        MOVEI   6,3       ;EXT LIMIT
        MOVE    2,EXTP    ;EXT POINTER
        MOVEM   2,PNBP    ;SET UP BYTE POINTER FOR EXT
        JUMPA   INCHR
SLBRAC: MOVE    1,CONFLG
        CAIL    1,1       ;SKP     IF FLG =0
        JUMPA   ERR9
        AOS     CONFLG    ;DON'T SKP
        SETZ    7,
        SETZ    3,
        MOVEI   6,5       ;NUMBER COUNTER
        JUMPA   INCHR

CCTCON: CAIGE   11,60     ;NUMBER?
        JUMPA   ERR9
        CAILE   11,67
        JUMPA   ERR9
        CAILE   7,4
        JUMPA   ERR10
        MOVEI   11,-60(11)
        ROT     3,3
        ADD     3,11
        JUMPA   INCHR
SRBRAC: HRRM    3,LXWD    ;STORE # IN XWD
        AOS     PPNFLG
        JUMPA   INCHR


SCOMA:  SKIPN   CONFLG    ;COMMA IS ILLEGAL IF CONFLG NOT SET
        JUMPA   ERR9
        HRLM    3,LXWD
        SETZ    3,
        SETZ    7,
        MOVEI   6,3
        JUMPA INCHR


CROUT:  MOVE    2,PPNFLG
        CAIE    2,1
        JUMPA   RSNPPN    ;START OVER
        MOVE    5,FILEN           ;SETUP FILE NAME, EXT, AND XWD
        MOVEM   5,EI
        MOVEM   5,EO      ;SETUP FOR INPUT OR OUTPUT
        MOVE    5,EXT
        MOVEM   5,EI+1
        MOVEM   5,EO+1
        MOVE    5,LXWD
        MOVEM   5,EI+3
        MOVEM   5,EO+3
        OUTSTP  MSG2
        CLRBPI
        SETZ    7,
        SETZ    3,
IN2:    INCHWL  11        ;BLOCK # 1 = ?
        CAIN    11,44     ;$? START OVER.
        JUMPA   RSNPPN
```

```
        CAIN    11,15
        JUMPA   SCR     ;SERVICE CR
        CAIGE   11,60   ;NUMBER?
        JUMPA   ERR9
        CAILE   11,71
        JUMPA   ERR9
        AOJ     7,      ;ADD 1 TO CHAR COUNTER
        CAILE   7,5
        JUMPA   ERR10
        MOVEI   11,-60(11)      ;CONVERT TO BIN #
        IMULI   3,|D10
        ADD     3,11
        JUMPA   IN2
SCR:    MOVEM   3,SBLK  ;STARTING BLOCK
        OUTSTR  MSG3    ;INPUT READ/WRITE OPTION
        SETZM   RWFLG
        CLRBFI
IN3:    INCHWL  11      ;READ OR WRITE ?
        CAIN    11,44   ;$? START OVER.
        JUMPA   RSNPPN
        CAIE    11,131  ;Y?
        JUMPA   SCR3    ;WRITE
        AOS     RWFLG
RWPARM: OUTSTR  MSG4    ;# BLOCKS ON TAPE
        SETZM   RDFLG1
        SETZM   3
        CLRBFI
IN4:    INCHWL  11
        CAIN    11,44
        JUMPA   RSNPPN
        CAIN    11,15   ;CR?
        JUMPA   SCR3
        AOS     RDFLG1
        CAIGE   11,60   ;NUMBER?
        JUMPA   ERR9
        CAILE   11,71
        JUMPA   ERR9
        MOVEI   11,-60(11)
        IMULI   3,|D10
        ADD     3,11
        JUMPA   IN4

STNOB:  MOVEM   3,NOBLKS
        JUMPA   SCR4
SCR3:   MOVE    2,RDFLG1
        CAILE   2,0
        JUMPA   STNOB   ;STORE # BLOCKS
        OUTSTR  MSG5    ;EOF ON TAPE =
        SETZM   EOF
        SETZ    4,
        SETOM   EOFFLG
        CLRBFI
IN5:    INCHWL  11      ;INPUT  EOF
        CAIN    11,44
        JUMPA   RSNPPN
        CAIN    11,15   ;CR?
        JUMPA   SCR4
        AOJ     4,      ;ADD 1 TO CHAR COUNTER
        CAILE   4,4     ;IF 4 OR LESS CONT.
        JUMPA   ERR10
        CAIGE   11,60
        JUMPA   ERR9
```

```
        CAILE   11,67   ;MUST BE OCTAL
        JUMPA   ERR9
        MOVEI   11,-60(11)      ;MAKE DIGIT
        MOVE    3,EOF
        ROT     3,3     ;ROT FOR ANOTHER OCTAL #
        ADD     3,11
        MOVEM   3,EOF
        JUMPA   IN5

SCR4:   SKIPE   RWFLG   ;TEST FOR READ OR WRITE
        JUMPA   INPUT
OUTPUT: OPEN    1,SPEC1 ;OPEN FOR READING DISK
        JUMPA   ERR8
        INBUF   1,3     ;SET UP 3 BUFS FOR DSK
        LOOKUP  1,FILEN ;LOCATE FILE ON DSK
        JUMPA   ERR1    ;NO FILE
        USETI   1,1     ;INITIALIZE FOR DSK INPUT
        OPEN    2,SPEC2 ;OPEN FOR DECTAPE OUTPUT
        JUMPA   ERR8
        MOVE    2,SBLK
        HRRM    2,USETO1        ;SET STARTING BLOCK
USETO1: USETO   2,1
RESET1: IN      1,      ;INPUT A BUF FROM DSK
        SKIPA
        JUMPA   ERRCHK
        SETZM   EFLG    ;EFLG=1 SAYS EOF HAS BEEN INPUT
        MOVE    3,BPC01 ;SET POINTER TO RETRIEVE BYTES
        MOVEM   3,BPO1
        MOVE    3,IBUF
        AOJ     3,
        AOJ     3,
        HRRM    3,BPO1  ;BPO1 GETS DATA
        MOVE    3,BPC02
        MOVEM   3,BPO2  ;BPO2 STORES IN TBUF1
        MOVE    5,IBUF+2
        IMULI   5,3
        MOVEM   5,BYTCNT        ;SAVE BYTE COUNT
LDOBUF: MOVEI   4,|D128
        ILDB    3,BPO1
        IDPB    3,BPO2
        SOJE    5,DTDMP ;IF BYTE COUNT IS 0 GO TO DTDMP
        SOJE    4,.+2
        JUMPA   LDOBUF+1
        SETZ    3,
        IDPB    3,BPO2
        JUMPA   LDOBUF
DTDMP:  MOVE    3,BYTCNT
        CAIL    3,600   ;SEE IF WE HAVE FULL BUFFER
        JUMPA   OUT     ;YES
        CAIN    4,1     ;NO
        IBP     3,BPO2  ;WE ARE AT A VALID BYTE SPACE
        AOS     EFLG    ;SET EFLG
        MOVE    2,EOF   ;GET EOF BYTE
        IDPB    2,BPO2
OUT:    OUTPUT  2,DOLST1
        JSR     OCHK
        JUMPA   ERR3
        OUTPUT  2,DOLST2
        JSR     OCHK
        JUMPA   ERR3
        OUTPUT  2,DOLST3
        SKIPA
```

```
        JUMPA   ERR3
        JUMPA   RESET1  ;GO GET ANOTHER BLOCK

OCHK:   0               ;SUBROUTINE TO CHECK HOW MANY BLOCKS
        AOS     OCHK    ;  TO OUTPUT.
        SKIPN   EFLG
        JRST    @OCHK
        SUBI    5,|D128
        SKIPL   5
        JRST    @OCHK   ;OUTPUT NEXT BLOCK
        MOVEI   3,RESET1
        HRRM    3,OCHK  ;SET RETURN TO OUTPUT NO MORE BLOCKS
        JRST    @OCHK

ERRCHK: STATZ   1,740000        ;CHECK EACH READ FOR ERROR
        JUMPA   ERR2    ;ERR ON INPUT FROM DSK
        SKIPE   EFLG    ;IF EFLG SET, EOF ALREADY OUTPUT
        JUMPA   EX
        MOVE    3,EOF   ;MUST INSERT EOF
        ROT     3,-|D12
        MOVEM   3,TBUF1
        OUTPUT  2,DOLST1
        SKIPA
        JUMPA   ERR3
        JUMPA   EX


INPUT:  OPEN    1,SPEC5 ;INPUT FROM DECTAPE
        JUMPA   ERR8
        OUTBUF  1,1     ;SETUP 1 BUF TO WRITE ON DSK
        ENTER   1,EO
        JUMPA   ERR6
        OUT     1,      ;INITIALIZE BUFFER
        SKIPA
        JUMPA   ERR6
        OPEN    2,SPEC6
        JUMPA   ERR8
        MOVE    2,SBLK  ;GET STARTING BLOCK #
        HRRM    2,USETI1
USETI1: USETI   2,1
        MOVE    10,NOBLKS
        SETZ    7,
LOOP2:  MOVE    3,BPCO1
        MOVEM   3,BPO1
        MOVE    3,BPCO2
        MOVEM   3,BPO2
        MOVE    3,OBUF
        AOJ     3,
        MOVEM   3,WCADR ;SAVE WORD COUNT ADR
        AOJ     3,
        HRRM    3,BPO1
        MOVEI   4,3
        SETZM   WCOUNT
        INPUT   2,DOLST1        ;READ FROM DECTAPE
        JSR     INCHK
        JUMPA   ERR7
        INPUT   2,DOLST2        ;AGAIN
        JSR     INCHK
        JUMPA   ERR7
        INPUT   2,DOLST3        ;AGAIN - NOW HAVE 1 DSK BLOCK
        SKIPA
        JUMPA   ERR7
LOOP1:  MOVEI   5,|D128
```

```
        ILDB    3,BPO2
        IDPB    3,BPO1
        AOS     WCOUNT  ;COUNT THE # OF 12 BIT BYTES
        SKIPE   NOBLKS
        JUMPA   BYTCHK
        CAME    3,EOF   ;CHECK FOR END OF FILE
        JUMPA   BYTCHK
        JUMPA   OPEXIT  ;IS EOF
BYTCHK: SOJE    5,.+2
        JUMPA   LOOP1+1
        SKIPN   NOBLKS          ;SKP IF NO BLOCKS NOT C
        JUMPA   CONT
        SOJE    10,.+2  ;SKP IF NO BLOCKS LEFT TO READ
        SKIPA
        JUMPA   BKEXIT
CONT:   IBP     BPO2
        SOJE    4,.+2
        JUMPA   LOOP1
        MOVEI   3,200
        HRRM    3,@OBUF+1       ;STORE WORD COUNT
OP:     OUT     1,
        JUMPA   LOOP2
        JUMPA   ERR3

INCHK:  0
        AOS     INCHK   ;SET RETURN PAST ERR MSG
        AOJ     7,
        CAME    7,10
        JRST    @INCHK
        MOVEI   7,LOOP1
        HRRM    7,INCHK
        JRST    @INCHK

OPEXIT: SOS     WCOUNT  ;COUNTED ECF SO SUB 1
BKEXIT: MOVE    3,WCOUNT        ;CAL # PDP-10 WORDS TO BE OUTPUT
        IDIVI   3,3
        SKIPE   4
        AOJ     3,
        HRRM    3,@WCADR
        OUT     1,      ;OUTPUT FINAL BUF
        JUMPA   EX
        JUMPA   ERR3
EX:     RELEAS  1,
        RELEAS  2,
        RELEAS  4,
        OUTSTR  MSGDN
        RELEAS  3,
        EXIT


ERR1:   OUTSTR  MSG10
        EXIT
ERR2:   OUTSTR  MSG11
        EXIT
ERR3:   OUTSTR MSG12
        EXIT
ERR4:   OUTSTR  MSG13
        EXIT
ERR5:   OUTSTR  MSG14
        EXIT
ERR6:   OUTSTR  MSG15
        EXIT
```

```
ERR7:   OUTSTR  MSG16
        JUMPA   RSNPPN
ERR8:   OUTSTR  MSG17
        EXIT
ERR9:   OUTSTR  MSG18
        JUMPA   RSNPPN
ERR10:  OUTSTR  MSG19
        JUMPA   RSNPPN

MSG1:   ASCIZ /NAME.EXT[PPN] = /
MSG2:   ASCIZ /BLOCK # 1 = /
MSG3:   ASCIZ /READ? (Y OR N)/
MSG4:   ASCIZ /# BLOCKS ON TAPE = /
MSG5:   ASCIZ /EOF ON TAPE = /
MSG6:   ASCIZ /# PDP-10 W PER BLK = /
MSG10:  ASCIZ /FILE NOT ON DSK/
MSG11:  ASCIZ /INPUT ERR ON DSK/
MSG12:  ASCIZ /OUTPUT ERR ON DT/
MSG13:  ASCIZ /DT INPUT ERR/
MSG14:  ASCIZ /DT OUTPUT ERR/
MSG15:  ASCIZ /ERR ON OUTPUT TO DSK/
MSG16:  ASCIZ /DATA MISSED OR PARITY ERR ON DT INPUT/
MSG17:  ASCIZ /OPEN UUO ERR/
MSG18:  ASCIZ /COMMAND ERROR /
MSG19:  ASCIZ /TOO MANY DIGITS IN FIELD /
MSGDN:  ASCIZ /DONE/

DOLST1: IOWD    |D43,TBUF1
        0
DOLST2: IOWD    |D43,   TBUF2
        0
DOLST3: IOWD    |D43,TBUF3
        0
BPC01:  POINT   12,0
BP01:   0
BPC02:  POINT   12,TBUF1
BP02:   0

TBUF1:  BLOCK   |D43
TBUF2:  BLOCK   |D43
TBUF3:  BLOCK   |D43
SPEC1:  EXP     10
        SIXBIT /DSK/
        XWD     0,IBUF
SPEC2:  EXP     116
        SIXBIT /DTA/
        0
SPEC3:  EXP     1
        SIXBIT /TTY/
        XWD     TTOUT,0
SPEC4:  EXP     1
        SIXBIT /TTY/
        XWD     0,TTIN

SPEC5:  EXP     30
        SIXBIT /DSK/
        XWD     OBUF,0
SPEC6:  EXP     116
        SIXBIT /DTA/
        0

EI:     0
        0
```

```
        0
        0
EO:     0
        0
        0
        0
IBUF:   BLOCK   3
OBUF:   BLOCK   3
TTIN:   BLOCK   3
TTOUT:  BLOCK   3
FILEN:  0
EXT:    0
LXWD:   0
FNP:    POINT   6,FILEN
EXTP:   POINT   6,EXT
FNBP:   0
EFLG:   0
CONFLG: 0
WCOUNT: 0
BYTCNT: 0
WCADR:  0
EOF:    0
EOFFLG: 0
NAMEF:  0
NOBLKS: 0
PERIOD: 0
PPNFLG: 0
RDFLG1: 0
RWFLG:  0
SBLK:   0
        END     START
```

INTERNAL DISTRIBUTION

1. R. K. Adams
2. L. L. Anthony
3. N. A. Betz
4. P. D. Brooke
5. A. A. Brooks
6-15. J. A. Carpenter
16. H. P. Carter
17. J. G. Craven
18. J. S. Crowell
19. F. L. Culler
20. R. A. Dory
21. R. J. Easterday
22. J. E. Francis
23-28. L. W. Gilley
29. R. K. Gryder
30. R. W. Henderson
31. R. F. Hibbs
32. R. P. Leinius
33. W. J. McClain
34. Erlie McDaniels
35. R. W. Peelle
36. C. E. Price
37. Herman Postma
38. G. B. Raine
39. C. R. Stewart
40. J. G. Sullivan
41. C. L. Thompson
42. P. R. Vanstrum
43. G. W. Westley
44. G. E. Whitesides
45-47. Central Research Library
48. Document Reference Section
49-68. Laboratory Records
69. Laboratory Records - Record Copy
70. ORNL Patent Office

EXTERNAL DISTRIBUTION

71. Dr. V. A. Kamath, Scientific Advisor, Attention: P. K. Patwardhan, Bhabha Atomic Research Centre, Trombay, Bombay, India
72. J. N. Rogers, Division 8321, Sandia Laboratories, P. O. Box 969, Livermore, CA 94550
73. Dr. Milton E. Rose, Mathematical and Computer Sciences Program, Molecular Sciences and Energy Research, Division of Physical Research, U. S. Atomic Energy Commission, Washington, DC 20545
74-75. Technical Information Center, Oak Ridge, Tennessee 37830
76. Research and Technical Support Division, ORO
77. Patent Office, AEC, ORO