NAM    PDS-SYS2N
OPT    O. NOG

PDS SYSTEM 2N CASETTE DRIVERS (SYS2NF)
PROGRAMMED BY ERIC JAMESON

COPYRIGHT 1976 SPHERE CORPORATION
940 N. 400 E.; NORTH SALT LAKE, UTAH  84054
P. O. BOX 213; BOUNTIFUL, UTAH  84010

SPHERE RESERVES ALL RIGHTS FOR THE REPRODUCTION,
DISTRIBUTION AND USE OF THE PDS SOFTWARE.
NO COPIES MAY BE MADE OR DISTRIBUTED WITHOUT THE
WRITTEN PERMISSION OF SPHERE CORP.

THE PROGRAM DEVELOPMENT SYSTEM (PDS SYS2N) IS A SET OF
PROGRAMS RESIDING ON ERASABLE PROGRAMMABLE READ ONLY
MEMORY WHICH ALLOW EVEN THE SMALLEST USER TO USE HIS
SPHERE SYSTEM AS A COMPLETE COMPUTER SYSTEM FOR THE
DEVELOPMENT OF COMPUTER PROGRAMS.
TOWARD THIS END, THE 5 PDS EPROMS CONTAIN A CURSOR
BASED EDITOR, A MINI-ASSEMBLER, AND THE SPHERE DEBUGGING
AID (SDA), AS WELL AS A SET OF UTILITY ROUTINES TO DO 16
BIT MULTIPLY AND DIVIDE, ASCII-TO-BINARY, AND
BINARY-TO-ASCII ROUTINES, AND ROUTINES TO DO
INPUT AND OUTPUT TO THE AUDIO CASSETTE.

THE SYS2N SOFTWARE IS AN UPGRADE OF THE PDS V3A
(V3N & V3D) SOFTWARE DESIGNED TO RUN WITH THE CASSETTE
SYSTEM. THERE ARE TWO VERSIONS OF THE SYS2 SOFTWARE:
SYS2N WHICH RUNS WITH THE NEW KEYBOARD AND THE SYS2A
WHICH RUNS WITH THE ORIGINAL (KBD/1A) KEYBOARD. THE
MAIN DIFFERENCE BETWEEN THE V3A AND SYS2 VERSIONS ARE
THAT A FIFTH EPROM HAS BEEN ADDED AND THAT THE MINI-
ASSEMBLER HAS BEEN DELETED AND REPLACED WITH A SET OF
COMMANDS TO DO LOADING AND DUMPING OF CASSETTE TAPES.

The SYS2NF prom is a version of the SYS2N cassette prom with a
software bug fixed. This bug would cause the next block of a
multiblock read to be skipped if the checksum on the preceding
block was a 16 and the tape had not been previously used. The
change is on page 10 where the ESC test and branch now branches
to RDHDR1 instead of RDHDR.
<table>
<thead>
<tr>
<th>Address</th>
<th>Symbol</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00051</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00052</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00053</td>
<td>TMP EQU</td>
<td>$00</td>
<td>16 BIT ACC. PSEUDO REG B.</td>
</tr>
<tr>
<td>00054</td>
<td>Tmp1 EQU</td>
<td>$02</td>
<td></td>
</tr>
<tr>
<td>00055</td>
<td>ARB EQU</td>
<td>$04</td>
<td>_HI BYTE OF ARB.</td>
</tr>
<tr>
<td>00056</td>
<td>AR3 EQU</td>
<td>$04</td>
<td>_LO BYTE OF ARB.</td>
</tr>
<tr>
<td>00057</td>
<td>AR2 EQU</td>
<td>$05</td>
<td></td>
</tr>
<tr>
<td>00058</td>
<td>ARA EQU</td>
<td>$06</td>
<td>16 BIT ARITH PSEUDO REG A.</td>
</tr>
<tr>
<td>00059</td>
<td>AR1 EQU</td>
<td>$06</td>
<td>_HI BYTE OF ARA.</td>
</tr>
<tr>
<td>00060</td>
<td>AR0 EQU</td>
<td>$07</td>
<td>_LO BYTE OF ARA.</td>
</tr>
<tr>
<td>00061</td>
<td>DIGIT EQU</td>
<td>$08</td>
<td>BYTE USED BY ASCBIN FOR TMP.</td>
</tr>
<tr>
<td>00062</td>
<td>CStatus EQU</td>
<td>$09</td>
<td>CASSETTE I-O STATUS BYTE.</td>
</tr>
<tr>
<td>00063</td>
<td>OUTEN EQU</td>
<td>$0A</td>
<td>END OF OUTPUT BUFFER TEXT.</td>
</tr>
<tr>
<td>00064</td>
<td>BUFADR EQU</td>
<td>$0C</td>
<td>START OF I/O BUFFER (PTR)</td>
</tr>
<tr>
<td>00065</td>
<td>BUFEND EQU</td>
<td>$0E</td>
<td>PTR TO END OF I/O BUFFR.</td>
</tr>
<tr>
<td>00066</td>
<td>OUTBUF EQU</td>
<td>$11</td>
<td>START OF OUTPUT BUFFER.</td>
</tr>
<tr>
<td>00067</td>
<td>ENDMEM EQU</td>
<td>$1A</td>
<td>DEST. ADDR. FOR TEXT MOVE.</td>
</tr>
<tr>
<td>00068</td>
<td>CSRPTR EQU</td>
<td>$1C</td>
<td>LAST ADDRES OF REAL MEMORY.</td>
</tr>
<tr>
<td>00069</td>
<td>BUFPTR EQU</td>
<td>$1E</td>
<td>PTR TO CURSOR ON SCREEN.</td>
</tr>
<tr>
<td>00070</td>
<td>BUFFLO EQU</td>
<td>$20</td>
<td>TEMP PTR USED BY OUTSTR.</td>
</tr>
<tr>
<td>00071</td>
<td>BUFFHI EQU</td>
<td>$22</td>
<td>PTR TO END OF LOW EDIT TXT.</td>
</tr>
<tr>
<td>00072</td>
<td>SCNPTST EQU</td>
<td>$24</td>
<td>PTR TO START OF HI TEXT.</td>
</tr>
<tr>
<td>00073</td>
<td>SRCASM EQU</td>
<td>$26</td>
<td>PTR TO BUFFER TXT START.</td>
</tr>
<tr>
<td>00074</td>
<td>ONDYAL EQU</td>
<td>$2A</td>
<td>PTR TO ASSMLR SOURCE CODE.</td>
</tr>
<tr>
<td>00075</td>
<td>SYMLAL EQU</td>
<td>$2C</td>
<td>HAS ASSMLR OPERND VALUE.</td>
</tr>
<tr>
<td>00076</td>
<td>BRKPTR EQU</td>
<td>$2E</td>
<td>VALUE PUT IN ASM. SYMVL.</td>
</tr>
<tr>
<td>00077</td>
<td>BRKADR EQU</td>
<td>$30</td>
<td>TEMP SAVE FOR BRKPT DATA.</td>
</tr>
<tr>
<td>00078</td>
<td>EDIT EQU</td>
<td>$32</td>
<td>ADDRESS OF BREAKPOINT.</td>
</tr>
<tr>
<td>00079</td>
<td>BLKNAME EQU</td>
<td>$33</td>
<td>0 IF EDITOR IS NOT RUNNING.</td>
</tr>
<tr>
<td>00080</td>
<td>IOBUFF EQU</td>
<td>$35</td>
<td>CASSETTE BLOCK NAME.</td>
</tr>
<tr>
<td>00081</td>
<td>ACIANO EQU</td>
<td>$38</td>
<td>I-O BUFFER FOR DEBUGGER.</td>
</tr>
<tr>
<td>00082</td>
<td>NOPINT EQU</td>
<td>$3A</td>
<td>SYS2N CASSETTE ACIA ADDR.</td>
</tr>
<tr>
<td>00083</td>
<td>BLKTP EQU</td>
<td>$3B</td>
<td>CASSETTE NAME PRINT FLAG.</td>
</tr>
<tr>
<td>00084</td>
<td>BFRPTR EQU</td>
<td>$3C</td>
<td>CASSETTE BLOCK TYPE CODE.</td>
</tr>
<tr>
<td>00085</td>
<td>BFRSZE EQU</td>
<td>$3E</td>
<td>ADDR. OF I-O BUFF. FOR CASS.</td>
</tr>
<tr>
<td>00086</td>
<td>PVICAL EQU</td>
<td>$40</td>
<td>LENGTH OF CASS. BUFFER.</td>
</tr>
</tbody>
</table>

**FOLLOWING ARE VARIABLE VALUES.**

<table>
<thead>
<tr>
<th>Address</th>
<th>Symbol</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00087</td>
<td>TIMEA EQU</td>
<td>$26F0</td>
<td>TIMER COUNTER.</td>
</tr>
<tr>
<td>00088</td>
<td>ON EQU</td>
<td>$01</td>
<td>ACIA VALUE TO TURN ON CASS.</td>
</tr>
<tr>
<td>00089</td>
<td>OFF EQU</td>
<td>$05</td>
<td>ACIA VALUE TO STOP CASS.</td>
</tr>
<tr>
<td>00090</td>
<td>ETX EQU</td>
<td>$03</td>
<td>END-OF-TEXT.</td>
</tr>
<tr>
<td>00091</td>
<td>SYN EQU</td>
<td>$16</td>
<td>SYNCHRONISE.</td>
</tr>
<tr>
<td>00092</td>
<td>ETB EQU</td>
<td>$17</td>
<td>END-OF-TRANSMISSION-BLOCK.</td>
</tr>
<tr>
<td>00093</td>
<td>ESC EQU</td>
<td>$18</td>
<td>ESCAPE TO NONSTANDARD HDR.</td>
</tr>
<tr>
<td>00094</td>
<td>ERR4 EQU</td>
<td>$'T</td>
<td>SET FOR TRAILER ERROR.</td>
</tr>
<tr>
<td>00095</td>
<td>ERR5 EQU</td>
<td>$'C</td>
<td>SET FOR CHECKSUM ERR.</td>
</tr>
<tr>
<td>00096</td>
<td>TIME EQU</td>
<td>20833</td>
<td>TIME CNTR FOR 1/4 SEC.</td>
</tr>
<tr>
<td>00097</td>
<td>TINCNT EQU</td>
<td>9</td>
<td>TIMES FOR 2 &amp; 1/4 SECONDS.</td>
</tr>
</tbody>
</table>
PDS SYSTEM 2N CASSETTE DRIVERS

ORG $FB00

CASSETTE I-O DRIVERS

THE CASSETTE DRIVERS LOAD AND DUMP A BLOCK OF DATA
TO AND FROM THE CASSETTE. THEY HANDLE BOTH THE HEADER
AND TRAILER FORMATTING.
THE DRIVERS ARE SET UP AS A SET OF SUBROUTINES
CALLED BY THE EXECUTIVE OR THE USER'S PROGRAMS.

LOW MEMORY ADDRESSES USED BY THE DRIVERS ARE:

CSTATS AT 09 USED TO STORE INPUT ERR CODE
BLKNAM AT 33 2 CHAR. NAME OF BLOCK.
ACIANO AT 38 PTR. TO THE ACIA CURRENTLY USED.
Noprnt AT 3A PRINT FLAG FOR BLOCK NAME.
BFPRTR AT 3C START OF CASSETTE I-O BUFFER.
BFRSZE AT 3E END OF CASSETTE DATA BUFFER.

THE DRIVERS CAN BE RUN FROM THE PDS-SYS2N SOFTWARE
SYSTEM, WHICH CONTAIN THE CASSETTE LOAD AND DUMP
COMMANDS IN THE EXEC, OR FROM THE PDS-V3A, WHERE THE
DRIVERS ARE CALLED THROUGH THE DEBUGGER.

THE DRIVERS DISPLAY PERTINENT DATA ON THE SCREEN
WHEN WRITING OR READING FROM THE CASSETTES. ON A WRITE,
THE CHARACTERS BEING WRITTEN ONTO THE CASSETTE ARE
DISPLAYED IN THE SECOND CHARACTER POSITION FROM THE
UPPER RIGHT HAND CORNER OF THE SCREEN. ON A READ, THE
CHARACTERS BEING READ IN ARE DISPLAYED IN THE UPPER
RIGHT HAND CORNER OF THE SCREEN. THE NAME OF THE BLOCK
CURRENTLY BEING READ OR SEARCHED OVER IS DISPLAYED ON THE
RIGHT HAND SIDE OF THE SECOND LINE. THE ERROR CODE FOR
A READ IS DISPLAYED ON THE RIGHT SIDE OF THE THIRD LINE.
NO CHANGE IN CHARACTER MEANS THAT THE READ WAS O.K.
A "C" MEANS THERE WAS A CHECKSUM ERROR ON THE BLOCK.
A "T" MEANS THAT THE WRONG NUMBER OF BYTES HERE READ
INTO THE BUFFER (TRAILER ERROR). THIS WOULD OF COURSE
IMPY A CHECKSUM ERROR ALONG WITH THE TRAILER ERROR.

THE CSTATS (CASSETTE STATUS) BYTE WILL CONTAIN A 0
IF THE BLOCK READ IN WAS O.K. IF IT WAS A BAD READ, IT
WILL CONTAIN A 54 FOR TRAILER ERROR OR A 43 FOR A
CHECKSUM ERROR CODE UPON EXIT FROM THE ROUTINE.

SETTING THE NOPRNT (NO PRINTING) FLAG TO A 0 WILL
STOP THE DISPLAY OF CHARACTERS ON THE SCREEN DURING
CASSETTE READ AND WRITE, EXCEPT FOR THE T & C ERROR
CODES, WHICH ARE ALWAYS DISPLAYED WHEN THEY OCCUR.
* IF THE FIRST BYTE OF BLKNAM (BLOCK NAME) IS A 0
* WHEN THE READ BLOCK ROUTINE IS ENTERED, THE NEXT BLOCK
* WILL BE READ FROM TAPE NO MATTER WHAT THE NAME OF THE
* TAPE BLOCK IS. ON THE SYS2N EXEC, A CONTROL SPACE
* CHARACTER CAN BE TYPED IN AS THE FIRST CHARACTER OF THE
* NAME IN THE LOAD BLOCK COMMAND. THUS, A (CNTL L)(CNTL
* SPACE) (X) WOULD READ IN THE NEXT BLOCK ON THE TAPE.

* THE TAPE FORMAT IS:
* SYN 16
* SYN 16
* SYN 16
* SYN 16
* ESC 1B
* HI BYTE OF 16 BIT BLOCK LENGTH
* LOW BYTE OF BLOCK LENGTH
* FIRST CHAR OF BLOCK NAME
* SECOND CHARACTER OF NAME
* DATA
* ...
* ...
* DATA
* ETX
* ETB 1F
* CHECKSUM
* CHECKSUM
* CHECKSUM
* CHECKSUM

* THE CHECKSUM IS CALCULATED FROM THE DATA, WHICH
* IS READ IN FROM THE CASSETTE BUFFER RESIDING IN MEMORY.

* THE FORMAT FOR THE CASSETTE BUFFER IS:

BFRPTR

C AssettE  buffer

BFRSZ

* WHEN DATA IS READ IN, THE READ ROUTINE SETS 'BFRSZ'
* TO POINT TO THE LAST CHARACTER READ INTO THE BUFFER
* NOTE THAT THERE IS NO OVERFLOW CHECK WHEN DATA IS READ
* INTO MEMORY. ON OUTPUT TO THE CASSETTE, THE BLOCK
* LENGTH IS CALCULATED FROM THE BFRPTR AND BFRSZ POINTERS.
* *
* THE READ BLOCK AND WRITE BLOCK ROUTINES AUTOMATICALLY
* TURN ON AND OFF THE CASSETTE.

3
THE ADDRESSES OF THE Routines FOR CONTROLLING THE CASSETTE ARE AS FOLLOWS:

INTLZ - FB00 : Initializes the ACIA FOR USE.
WRITEL - FB2D : Writes a formatted block to cassette.
READL - FB31 : Reads in a formatted block of tape.
WRITEF - FB2F : Writes a block to a running cassette.
READF - FB33 : Reads a block from a running tape.
CASOUT - FB62 : Writes out onto tape the byte in A.
CASIN - FB7E : Reads in a byte into accum. A.
TURNON - FB77 : Turns on the cassette drive.
TRNOFF - FBB0 : Turns off the cassette drive when the pointer to ACIA is passed in the X reg.

NOTE THAT ALL THE ABOVE ROUTINES ARE SUBROUTINES THAT ARE ENTERED BY A JSR OR BSR CALL.

USING THE CASSETTE DRIVERS.

IF THE EPROMS ON THE CPU BOARD ARE THE SYS2N OR THE SYS2D PROMS, THE DRIVERS ARE USED BY THE I (INITIALIZE), L (LOAD FROM CASSETTE) AND S (STORE ONTO CASSETTE) COMMANDS THAT ARE A PART OF THE SYS-2 EXECUTIVE.

IF THE CPU PROMS BEING USED ARE THE PDS-V3A OR THE PDS-V3N OR V3D PROM SETS, THEN THE USER MUST SET UP THE POINTERS AND CALL THE ROUTINES HIMSELF, USING THE DEBUGGER ON THE CPU PROMS. (NOTE THAT THE V3A AND V3N DEBUGGERS VARY, FOR INSTANCE THE V3A GO COMMAND IS A 'G' WHILE THE V3N GO COMMAND IS A 'CNTL G'. REFER TO THE USERS MANUAL FOR DETAILS.)

THE FIRST THING TO DO IS OPEN THE LOW MEMORY LOCATIONS USED BY THE DRIVERS AS FLAGS AND POINTERS AND INITIALIZE THE LOCATIONS. THE ACIANO, BLKNAM, BFRPTR AND BFRSZ SHOULD NOW BE GIVEN VALUES.

ACIANO WOULD POINT TO F050 FOR THE FIRST CASSETTE AND TO F060 FOR THE SECOND CASSETTE DRIVE.

A SIMPLE PROGRAM TO CALL THE ROUTINES SHOULD NOW BE WRITTEN. OPEN LOCATION 900. TYPE IN THROUGH THE DEBUGGER THE INSTRUCTIONS JSR BDXX, JMP FE4F. XX IS THE SECOND BYTE OF THE DESIRED DRIVER ROUTINE, I.E. 00 FOR INITIALIZATION OR 91 FOR READING A BLOCK OR 2D FOR WRITING A BLOCK. THUS, TO INITIALIZE AN ACIA, ACIANO WOULD BE LOADED WITH THE ADDRESS OF THE ACIA AND THE USER WOULD THEN JUMP TO THE ROUTINE AT 900 BY OPENING 900 AND JUMPING TO IT WITH THE 'G' COMMAND. THE ROUTINE WOULD BE AS FOLLOWS:

CALL 900 BD JSR
* 901 FB HI BYTE OF ROUTINE ADDRESS
* 902 00 LOW BYTE OF DRIVER ADDRESS.
* 903 7E JMP
* 904 FE ADDRESS OF THE DEBUGGER ON
* 905 4F THE PDS-V3A PROM SET
THE ACIA WOULD NOW BE INITIALIZED AND CONTROL WOULD
HAVE RETURNED TO THE DEBUGGER. TO READ IN A BLOCK,
THE NAME WOULD BE PUT IN BLKNAM AND LOCATION 902 WOULD
BE CHANGED TO 91. THE USER WOULD THEN JUMP TO 900.

TO TEST THE CASSETTE, SET BFRPTR TO 6060 AND BFRSZE
TO 8E00. THIS WILL ALLOW THE USER TO WRITE OUT DATA
FROM THE FOURTH, FIFTH, SIXTH AND SEVENTH LINE OF THE
CRT DISPLAY AND THEN READ IT BACK ONTO THE DISPLAY. DATA
CAN BE TYPED ONTO THE SCREEN BY OPENING A LOCATION WITH
THE DEBUGGER AND THEN MOVING THE CURSOR AROUND THE SCREEN
TO CHANGE THE CHARACTERS. THIS IS POSSIBLE BECAUSE THE
ROUTINE TO INPUT AN ADDRESS CALLS THE EDITOR FOR INPUT.

THE DRIVER ROUTINES CAN ALSO BE USED TO PERFORM
I-O WITH A MODEM OR TELETYPE. THE MAIN HARDWARE
DIFFERENCE BETWEEN THE CASSETTE AND MODEM/TTY IS THAT THE
CASSETTE HAS A DIVIDE BY 16 CLOCK AND IS UNIDIRECTIONAL
WHILE THE MODEM/TTY HAVE A DIVIDE BY 64 CLOCK AND CAN BE
BIDIRECTIONAL. BECAUSE OF THE CLOCK CHANGE THE ACIA MUST
BE TURNED ON WITH A DIFFERENT VALUE BEFORE THE READ BLOCK
OR WRITE BLOCK ROUTINES ARE ENTERED. TO TURN ON THE ACIA
STORE THE VALUE '82' INTO LOCATION F050 OR F060. ONCE
IT IS TURNED ON, EITHER CASIN OR CASOUT MAY BE CALLED
REPEATEDLY OR WRITE MODEM BLOCK (WRMOD) OR READ MODEM
(RDOMOD) CAN BE CALLED ONCE. WRMOD & RDOMOD ARE THE SAME
AS WRTBLK & RDBLK EXCEPT THEY DO NOT TURN ON THE ACIA. A
PROGRAM TO READ IN A BLOCK OF DATA FROM A MODEM OR TTY
WOULD THUS BE (PLACED BEFORE THE 'CALL' ROUTINE):
8FA DE 38 LDX ACIANO LOADS ACIA POINTER.
8FC 86 B2 LDA A #$B2 LOADS STARTUP VALUE.
8FE A7 00 STA A 0,X PUTS START CODE INTO ACIA.
LOCATION 902 WOULD NOW BE 2F FOR WRITING AND 93 TO READ.
TO READ IN A CHARACTER FROM THE TELETYPE TURN ON THE
ACIA AND GO TO THE FOLLOWING ROUTINE:
TTYIN JSR CASIN READS IN A CHAR FROM KEYBOARD.
JMP CASOUT TYPES OUT CHAR ON PRINTER.
THE RS232 SHOULD HAVE BEEN STRAPPED TO HALF DUPLEX.

IT IS TO BE STRESSED THAT THE RELIABILITY OF THE
CASSETTE CONTROLLER DEPENDS ON THE ADJUSTMENT OF THE
TRIMMER ON THE SIM BOARD. IF THE TRIMMER IS OUT OF
ADJUSTMENT THE DATA WILL NOT READ IN PROPERLY. BECAUSE
OF THE OSCILLOSCOPE TO ADJUST THE TRIMMER. IT CAN ALSO
BE ADJUSTED BY READING IN A STRING OF SINGLE CHARACTERS
FROM THE CASSETTE AND ADJUSTING IT UNTIL THE CHARACTERS
SYNC IN PROPERLY. THE BEST CHARACTER TO USE IS A STRING
OF 'U'S. TO READ IN THE STRING FOR TESTING USE A
LOOP TO GET A CHARACTER FROM THE CASSETTE AND THEN
DISPLAY THAT CHARACTER. A SAMPLE ROUTINE TO DO THIS IS:
A JSR $FB7E LOADS A WITH CASSETTE CHAR.
JSR $FCAD POS-V3A PUTCHR ROUTINE.
TST $F001 TESTS KEYBOARD FOR A KEY.
BPL A SKIPS BACK IF NO INPUT.
THE ABOVE ROUTINE WOULD INPUT CHARACTERS UNTIL A KEY
ON THE KEYBOARD WAS DEPRESSED.
INTLZ INITIALIZES THE ACIA CONTROLLER FOR A SPECIFIC TAPE UNIT AT MOUNT TIME.

INTLZ LDX ACIANO \x GETS ACIA ADDRESS.
LDA #13 \x RESETS THE ACIA.
STA 0,X
LDA #OFF \x SETS ACIA TO \x 16 2 STOPS, BIT FORMAT.
STA 0,X
RTS

WRTHDR FORMATS THE HEADER ON THE TAPE.

LDA A \#SYN \x PUTS SYNC CHARs ONTO TAPE.
LDA A BSR \x CASOUT
BSR CASOUT
LDA A \#ESC
BSR CASOUT
LDA B BFRSZE \x FOLLOWING OUTPUTS LENGTH.
LDA B BFRSZE+1 \x LOADS LO BYTE OF END PTR.
SUB B BFRPTR+1 \x SUBS LO BYTE OF BEGIN PTR.
SBC A BFRPTR \x SUBS HI BYTE OF START PTR.
BSR CASOUT \x OUTPUTS HI LENGTH BYTE.
TBA \x LOADS LO BYTE OF LENGTH.
BSR CASOUT \x OUTPUTS LOW LENGTH BYTE.
LDA A BLKNAM \x PUTS OUT NAME OF BLOCK.
BSR CASOUT
LDA A BLKNAM+1 \x PUTS OUT LAST OF NAME.
BSR CASOUT
RTS \x RETURNS BACK TO WRBLK.

WRBLK WRITES OUT A BLOCK OF DATA TO THE CASSETTE.

BSR \x TURNON \x TURNS ON THE CASSETTE.
FOLLOWING WRITES FOR CASSETTE TO GET UP TO SPEED.
BSR WRTMOD \x TIMCNT \x LOADS TIME LOOP COUNTER.
LDA B \#TIME1 \x MASTER TIME LOOP (1/4 SEC).
LDA \#TIME \x COUNTS CYCLES OF LOOP.
DEK \x TIME1 \x TESTS FOR FIRST TIME OUT.
BNE \x TIME1 \x COUNTS TIMES IN LOOP.
DEC B \x TIME1 \x SKIPS BACK UNTIL DONE.
THE TIME LOOP IS NOW FINISHED.
BSR WRTHDR \x WRITES HEADER ON THE TAPE.
BSR WRTBFR \x WRITES OUT BUFFER DATA.
FOLLOWING WRITES THE TRAILER OUT ONTO THE TAPE.
WRTTLR LDA A \#ETB \x OUTPUTS END-OF-BLOCK CHAR.
BSR CASOUT \x ETB IS DISPLAYED A "N".
TBA \x GETS CHECKSUM FROM B.
BSR CASOUT \x OUTPUTS THE CHECKSUM.
BSR CASOUT \x OUTPUTS TRAILER FILLER BYTES.
BSR CASOUT \x END OF TRAILER WRITING ROUTINE.
BSR TRNOFF \x HALTS CASSETTE DRIVE
RTS

-
WRTBFR writes out the contents of the buffer onto the cassette tape.

WRTBFR CLR B - init checksum count.
LDX BFRPTR ACIA control mask.
WBFR1 LDA A 0.X loads char. from buffer.
STX TMP saves buffer ptr.
BSR CASOUT puts char onto cassette.
ABA A gets A+B.
TAB B gets A+B.
LDX TMP restores ptr. into buffer.
CPX BFRSZE tests if buffer empty.
BEQ CSOEXT exits when empty.
INX INC pointer.
BNE WBFR1 skips back if chars. left.

CASOUT takes the char in A and puts it out onto the cassette tape.

CASOUT PSH A saves char to read out.
LDX ACIANO X gets physical ACIA addr.
LDA A #2 loads control test bits.
CASO1 BIT A 0.X tests if ACIA buffer empty.
BEQ CASO1 loops back until ready.
PUL A gets orig char.
STA 1.X stores char into ACIA buffer.
TST NOPRNT tests if printout allowed.
BEQ CSOEXT skips printing if A 0.
STA A $E01E displays char on screen.
CSOEXT RTS

Turnon turns on the cassette drive.

Turnon LDX ACIANO loads cassette ACIA address.
LDA A #ON
STA A 0,X turns ACIA on.
CASIN reads in a character from the cassette tape into the A accumulator.

RDBLK reads in a block from the cassette tape into buffer memory.

END OF TRAILER READ IN.

Following turns off the cassette drive.
RDHDR FINDS THE START OF THE BLOCK ON THE TAPE. FIGURES THE BUFFER END AND CHECKS THE NAME.

RDHDR F1ND5S THE START OF THE BLOCK ON THE TAPE. FIGURES THE BUFFER END AND CHECKS THE NAME.

RDHDR1

BSR CASIN

CMP A #SYN

BNE RDHDR

BSR CASIN

CMP A #ESC

BNE RDHDR1

GOES BACK IF NOT GOOD HDR.

BSR CASIN

READS IN HI BYTE OF LEN.

TAB

SAVES HI LENGTH BYTE.

BSR CASIN

INPUTS LO SIZE (LEN) BYTE.

ADD A BFPRTR+1

FORMS POINTER TO THE

ADC B BFPRTR

TOP BYTE OF THE BUFFER.

STA A BFRRSZE+1

SAVES THE HI BFF. PTR.

STA B BFRRSZE

TO THE CASSETTE BUFFER.

BSR CASIN

READS IN BLOCK NAME.

FSH A -

SAVES FIRST CHAR OF NAME.

BSR CASIN

READ IN SECOND CHAR INTO A.

LDA B NOPRT

TESTS IF PRINT IS OFF.

PUL B -

RESTORES FIRST NAME CHAR.

BEQ RHDR1

SKIPS IF PRINT FLAG IS 0.

STA B $E03E

DISPLAYS BLOCK NAME ON

STA A $E03F

THE CRT SCREEN.

TST, BLKNAM

TESTS IF NAME IS CHECKED.

BEQ RHDR2

SKIPS IF NO NAME CHECK.

CMP B BLKNAM

TESTS FIRST CHAR OF NAME.

BNE RDHDR

SKIPS BACK IF BAD NAME

CMP A BLKNAM+1

TESTS SECOND NAME CHAR.

BNE RDHDR

SKIPS BACK IF BAD NAME.

RD8FR READS DATA INTO THE MEMORY BUFFER FROM THE CASSETTE.

RD8FR

CLR B -

LDX BFRPRTR

STA TMP

BSR CASIN

LDX TMP

STA A 0,X

ABA

TAB

CPX BFRRSZE

BEQ RHDR2

SKIPS TO EXIT IF ALL IS IN.

INX

BNE JLT RD8FR

END

INIT B FOR CHECKSUM.

LOADS START OF BUFFER.

A GETS CHAR READ IN.

X GETS BUFFER PTR.

STORS CHAR INTO BUFFER.

H GETS A+B.

B GETS A.

TESTS IF BUFFER FULL.

SKIPS TO EXIT IF ALL IS IN.

INC TO NEXT CHAR POSITION.

GOES BACK IF ANY LEFT.