DRV11B

DIAGNOSTIC TEST

MD-11-DVDRA-A

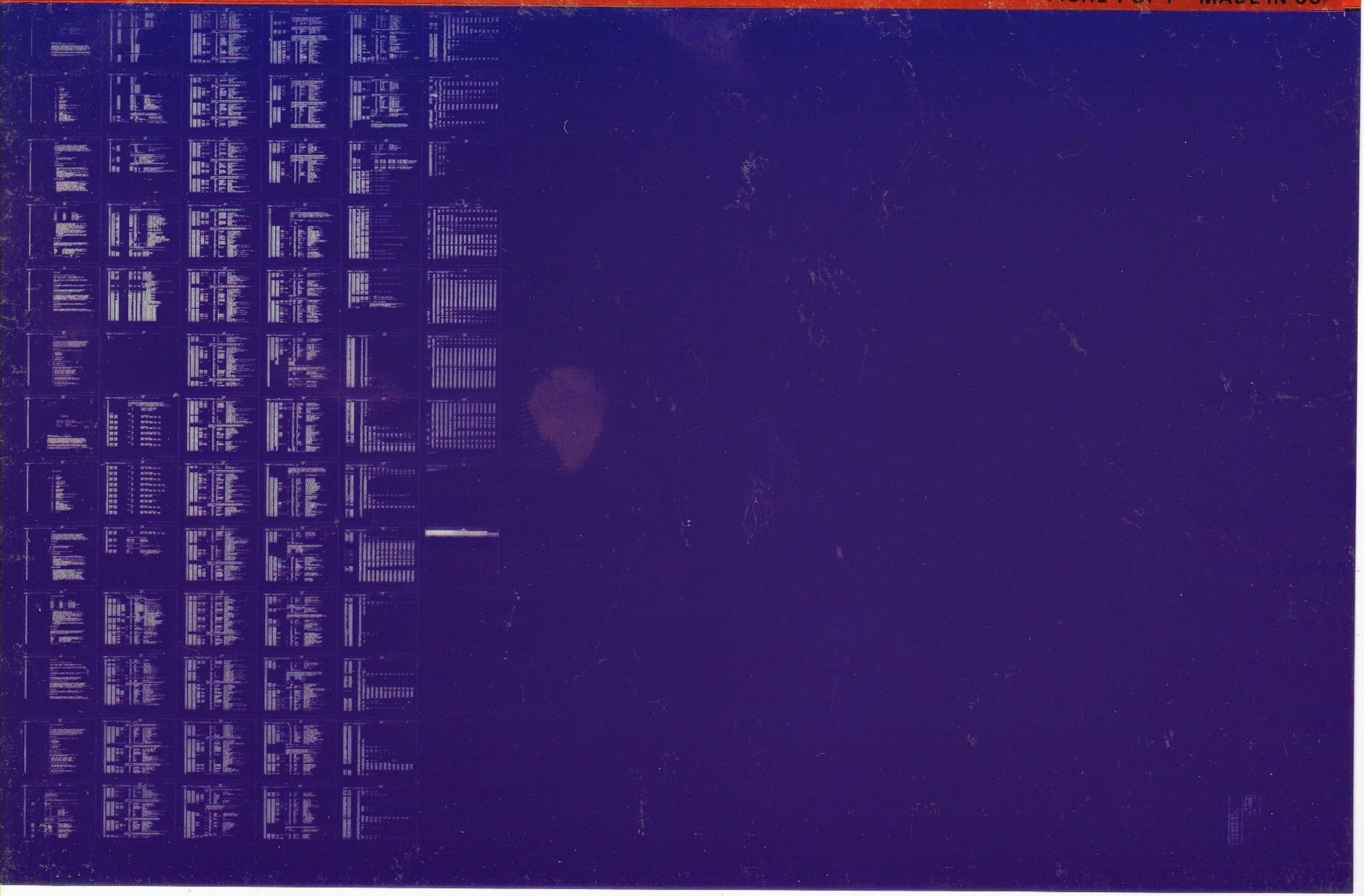
EP-DVDRA-A-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA



.REM!

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DVDRA-A
PRODUCT NAME: DRV11B DMA INTERFACE DIAGNOSTIC
DATE: OCTOBER 1976
MAINTAINER: DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

1.0 ABSTRACT

THE DRV11B DIAGNOSTIC PROGRAM IS A SERIES OF TESTS DESIGNED
TO TEST ALL LOGIC FUNCTIONS AND DATA PATHS ACCESSIBLE WITH
THE LOOP BACK CABLE INSERTED IN THE USER I/O CONNECTORS.
TOTAL PROGRAM CONTROL IS ACCOMPLISHED THRU THE CONSOLE TERMINAL
VIA THE ODT/CONSOLE MICROCODE AND THE PROVISIONS OF SECTION 5
OF THIS DOCUMENT. IF THE SYSTEM ALSO INCLUDES AN "REV11" (DMA REFRESH),
THE DMA REFRESH MUST BE DISABLED AND CPU REFRESH MUST BE ENABLED.

2.0 REQUIREMENTS

2.1 EQUIPMENT

1. PDP11/03 COMPUTER OR LSI-11 PROCESSOR
2. DLV11 WITH I/O TYPE TERMINAL
3. DRV11B WITH LOOP BACK CABLE

2.2 STORAGE

THE PROGRAM USES THE LOWER 4K OF MEMORY.

3.0 LOADING PROCEDURE

1. ASSURE THAT THE LSI-11 IS IN THE ODT MICROCODE STATE.
2. LOAD THE LOW OR HIGH SPEED READER WITH THE ABSOLUTE LOADER TAPE.
3. TYPE THE READER'S CSR ADDRESS (177550-LOW OR 177550-HIGH) AND CHARACTER 'L'.
4. AFTER TAPE IS LOADED, LOAD THE DRV11B BINARY TAPE INTO THE READER AND TYPE THE CHARACTER 'P'.
5. IF THE ABSOLUTE LOADER HAS ALREADY BEEN LOADED (STEPS 2 & 3), THEN ONLY THE STARTING ADDRESS OF THE ABSOLUTE LOADER AND THE CHARACTER 'G' NEED BE TYPED (WITH THE DRV11B BINARY TAPE IN THE APPROPRIATE READER).

4.0 STARTING PROCEDURE

1. MAKE SURE THE MAINTENANCE LOOP BACK CABLE IS INSERTED IN THE I/O CONNECTORS ON THE M7950 MODULE.
2. MAKE SURE THE DEVICE BUS & VECTOR ADDRESSES AGREE WITH THE DEFAULT VALUES DEFINED IN SECTION 7.1. IF NOT, CHANGE LOCATION(S) AS DESIRED VIA THE 'ADDRESS/' ODT COMMAND.
3. INSURE THAT THE HALT SWITCH IS DISABLED (IF ANY).
4. TYPE THE STARTING ADDRESS OF 200 AND THE CHARACTER G.
5. THE PROGRAM WILL RESPOND BY TYPING THE SOFTWARE SWITCH REGISTER CONTENTS AND ALLOWING THE USER TO CHANGE ITS CONTENTS BY ENTERING OCTAL SWITCH REGISTER DATA TERMINATED BY A CARRIAGE RETURN - SEE SECTION 5.0 FOR SWITCH REGISTER OPTIONS.

5.0 SOFTWARE SWITCH REGISTER

5.1 OPTIONS

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON TEST
SW13=1	020000	INHIBIT ERROR TYPEOUTS
SW11=1	004000	INHIBIT ITERATIONS
SW10=1	002000	BELL ON ERROR
SW09=1	001000	LOOP ON ERROR
SW08=1	0004XX	LOOP ON TEST IN SWR (7-0)

5.2 CONTROL

1. THE SOFTWARE SWITCH REGISTER 'SWREG' (LOC. 176) CAN BE CHANGED BY USING THE ODT FACILITIES.
2. THE SOFTWARE SWITCH REGISTER CAN BE CHANGED UNDER PROGRAM CONTROL BY TYPING THE 'CONTROL & G' KEYS. THIS KEYBOARD OPERATION WILL PRINT OUT THE CURRENT CONTENTS AND ACCEPT NEW OCTAL SWITCH REGISTER DATA TERMINATED WITH A CARRIAGE RETURN.
3. ONCE THE ODT MODE HAS BEEN ENTERED BECAUSE OF AN ERROR CONDITION WITH BIT15 SET (HALT ON ERROR), STEP #2 ABOVE IS OF NO VALUE, SO RESORT TO STEP #1 TO ALTER THE SOFTWARE SWITCH REGISTER IF DESIRED BEFORE TYPING 'P' (CONTINUE).
4. IF THE PROGRAM IS PERFORMING RESET INSTRUCTIONS, SEVERAL 'CONTROL & G' COMMANDS MAY BE NECESSARY TO BE ACKNOWLEDGE BY THE PROGRAM.

6.0 ERROR REPORTING

6.1 ERROR COMMENT

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED IF NEEDED FROM THE COMMENT AT THE ERROR PC OR FROM THE TEST ITSELF.

6.2 ERROR DATA

*ERRPC	LISTING ADDRESS WHERE THE ERROR WAS DETECTED
*TSTNUM	TEST NUMBER WHERE THE ERROR OCCURRED
BUSADR	DRV11B BUS REG ADDRESS OF CONCERNED OPERATION
EXPCT	DATA THAT WAS EXPECTED
RCVD	DATA THAT WAS RECEIVED
ADRS	MEMORY ADDRESS OF DATA TRANSFER ON ERROR

*ALWAYS REPORTED

7.0 MISCELLANEOUS

7.1 DRV11B BUS & VECTOR ADDRESS MODIFICATION

MODIFY LOCATION '$BASE' IF BASE BUS ADDRESS IS NOT 172410.
MODIFY LOCATION '$VECT1' IF VECTOR ADDRESS IS NOT 124.

*NOTE: USE THE LSI-11 ODT FACILITIES TO MODIFY THESE LOCATIONS AFTER PROGRAM LOAD. NO VECTOR ASSIGNMENT ABOVE 774 SHOULD BE ALLOWED.

7.2 XXDP/APT NOTES

THIS DIAGNOSTIC IS CHAINABLE UNDER XXDP (REF. 7.5)(REQUIRES 8K OR MORE).
THIS DIAGNOSTIC DOES SUPPORT "FPT" AND HAS RUN UNDER IT.

7.3 POWER FAIL

A POWER FAILURE WILL CAUSE A RESTART MESSAGE ON POWER UP AT WHICH TIME THE PROGRAM IS RESTARTED (ONLY ON SYSTEMS WITH NON-VOLATILE MEMORY AND WITH APPROPRIATE HARDWARE).

7.4 MULTIPLE DRV11B INTERFACE TESTING

THIS PROGRAM DOES NOT "AUTO-SIZE" THE NUMBER OF DRV11B'S CONNECTED.
THIS DIAGNOSTIC WILL TEST SEQUENTIALLY UP TO 8 DRV11B INTERFACES WITH CONTIGUOUS BUS AND VECTOR ADDRESSES. THIS IS ACCOMPLISHED BY THE OPERATOR SETTING UP LOCATION '$DEVM' WITH A BIT MAP INDICATING WHAT INTERFACES ARE TO TESTED. I.E. BIT0=1 SAYS TEST 1ST DRV11B, BIT1=1 SAYS TEST 2ND DRV11B, BIT2=1 SAYS TEST 3RD DRV11B, ETC..

7.5 RESTRICTIONS

IF THE SYSTEM ALSO INCLUDES AN "REV11" (DMA REFRESH), THE DMA REFRESH MUST BE DISABLED AND CPU REFRESH MUST BE ENABLED.

8.0 EXECUTION TIME

EXECUTION TIME RANGES FROM ABOUT 5 SECONDS WITH NO ITERATIONS TO ABOUT 90 SECONDS WITH ITERATIONS ENABLED WITH ONE DRV11B CONNECTED.
AN END PASS MESSAGE INDICATES ALL TESTS HAVE COMPLETED ON ALL SELECTED UNITS.

9.0 PROGRAM TEST DESCRIPTIONS

9.1 GENERAL

THIS DIAGNOSTIC CONTAINS A SERIES OF INDEPENDENT TESTS DESIGNED TO TEST LOGIC FUNCTIONS AND DATA PATHS OF THE DRV11B DMA INTERFACE. A HIGH DEGREE OF TESTING IS ACCOMPLISHED WITH THE AID OF THE MAINTENANCE LOOP BACK CABLE PROVIDED FOR DIAGNOSTIC TESTING. A COMPLETE LIST OF TESTS IS AVAILABLE IN THE TABLE OF CONTENTS AT THE BEGINNING OF THE LISTING. THE COMMENT FIELD WITHIN EACH TEST CAN BE BENEFICIAL IN TEST UNDERSTANDING.

9.2 REGISTER TESTS

THE FOLLOWING REGISTERS ARE READ/WRITE & RESET TESTED:

1. WORD COUNT
2. BUFFER ADDRESS
3. COMMAND/STATUS
4. DATA BUFFER

9.3 BYTE ADDRESSING TESTS

1. COMMAND/STATUS
2. DATA BUFFER

9.4 'FNCT' TO 'STAT' WRAP AROUND TEST

9.5 READY INTERRUPT TEST

9.6 NPR DATA TRANSFER TESTS

THE FOLLOWING NPR XFERS ARE CHECKED FOR CORRECT STATUS, WORD COUNT, BUFFER ADDRESS & DATA:

1. SINGLE 'DATI' XFER - FLOATING I/O PTRN
2. SINGLE 'DATO' XFER - FLOATING I/O PTRN
3. 200 'DATI' XFERS - FLOATING I/O PTRN
4. 200 'DATO' XFERS - FLOATING I/O PTRN
5. SINGLE 'DATI' XFER TO THE TTY PRINTER CSR

9.7 MAINT MODE NPR DATA TRANSFER TESTS

1. THAT MAINT MODE CONTROLS 'FNCT' BITS
2. 200 MAINT MODE XFERS - CHECKING STATUS & DATA
3. 200 MAINT MODE XFERS TO EACH 4K AVAILABLE MEM

9.8 BURST & NON-BURST MODE TESTS

1. THAT CPU IS LOCKED OUT IN BURST MODE
2. THAT CPU IS NOT LOCKED OUT IN NON-BURST MODE

9.9 'NEX' ERROR CONDITION TEST

```
                                        1C.0    LISTING
                                                -------
                                        !
                                        .REM!
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DVDRA-A

PRODUCT NAME: DRV11B DMA INTERFACE DIAGNOSTIC

DATE: OCTOBER 1976

MAINTAINER: DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

1.0 ABSTRACT

THE DRV11B DIAGNOSTIC PROGRAM IS A SERIES OF TESTS DESIGNED
TO TEST ALL LOGIC FUNCTIONS AND DATA PATHS ACCESSIBLE WITH
THE LOOP BACK CABLE INSERTED IN THE USER I/O CONNECTORS.
TOTAL PROGRAM CONTROL IS ACCOMPLISHED THRU THE CONSOLE TERMINAL
VIA THE ODT/CONSOLE MICROCODE AND THE PROVISIONS OF SECTION 5
OF THIS DOCUMENT. IF THE SYSTEM ALSO INCLUDES AN "REV11" (DMA REFRESH),
THE DMA REFRESH MUST BE DISABLED AND CPU REFRESH MUST BE ENABLED.

2.0 REQUIREMENTS

2.1 EQUIPMENT

1. PDP11/03 COMPUTER OR LSI-11 PROCESSOR
2. DLV11 WITH I/O TYPE TERMINAL
3. DRV11B WITH LOOP BACK CABLE

2.2 STORAGE

THE PROGRAM USES THE LOWER 4K OF MEMORY.

3.0 LOADING PROCEDURE

1. ASSURE THAT THE LSI-11 IS IN THE ODT MICROCODE STATE.
2. LOAD THE LOW OR HIGH SPEED READER WITH THE ABSOLUTE LOADER TAPE.
3. TYPE THE READER'S CSR ADDRESS (177560-LOW OR 177550-HIGH) AND CHARACTER 'L'.
4. AFTER TAPE IS LOADED, LOAD THE DRV11B BINARY TAPE INTO THE READER AND TYPE THE CHARACTER 'P'.
5. IF THE ABSOLUTE LOADER HAS ALREADY BEEN LOADED (STEPS 2 & 3), THEN ONLY THE STARTING ADDRESS OF THE ABSOLUTE LOADER AND THE CHARACTER 'G' NEED BE TYPED (WITH THE DRV11B BINARY TAPE IN THE APPROPRIATE READER).

4.0 STARTING PROCEDURE

1. MAKE SURE THE MAINTENANCE LOOP BACK CABLE IS INSERTED IN THE I/O CONNECTORS ON THE M7950 MODULE.
2. MAKE SURE THE DEVICE BUS & VECTOR ADDRESSES AGREE WITH THE DEFAULT VALUES DEFINED IN SECTION 7.1. IF NOT, CHANGE LOCATION(S) AS DESIRED VIA THE 'ADDRESS/' ODT COMMAND.
3. INSURE THAT THE HALT SWITCH IS DISABLED (IF ANY).
4. TYPE THE STARTING ADDRESS OF 200 AND THE CHARACTER G.
5. THE PROGRAM WILL RESPOND BY TYPING THE SOFTWARE SWITCH REGISTER CONTENTS AND ALLOWING THE USER TO CHANGE ITS CONTENTS BY ENTERING OCTAL SWITCH REGISTER DATA TERMINATED BY A CARRIAGE RETURN - SEE SECTION 5.0 FOR SWITCH REGISTER OPTIONS.

5.0 SOFTWARE SWITCH REGISTER

5.1 OPTIONS

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON TEST
SW13=1	020000	INHIBIT ERROR TYPEOUTS
SW11=1	004000	INHIBIT ITERATIONS
SW10=1	002000	BELL ON ERROR
SW09=1	001000	LOOP ON ERROR
SW08=1	0004XX	LOOP ON TEST IN SWR <7-0>

5.2 CONTROL

1. THE SOFTWARE SWITCH REGISTER 'SWREG' (LOC. 176) CAN BE CHANGED BY USING THE ODT FACILITIES.
2. THE SOFTWARE SWITCH REGISTER CAN BE CHANGED UNDER PROGRAM CONTROL BY TYPING THE 'CONTROL & G' KEYS. THIS KEYBOARD OPERATION WILL PRINT OUT THE CURRENT CONTENTS AND ACCEPT NEW OCTAL SWITCH REGISTER DATA TERMINATED WITH A CARRIAGE RETURN.
3. ONCE THE ODT MODE HAS BEEN ENTERED BECAUSE OF AN ERROR CONDITION WITH BIT15 SET (HALT ON ERROR), STEP #2 ABOVE IS OF NO VALUE, SO RESORT TO STEP #1 TO ALTER THE SOFTWARE SWITCH REGISTER IF DESIRED BEFORE TYPING 'P' (CONTINUE).
4. IF THE PROGRAM IS PERFORMING RESET INSTRUCTIONS, SEVERAL 'CONTROL & G' COMMANDS MAY BE NECESSARY TO BE ACKNOWLEDGE BY THE PROGRAM.

6.0 ERROR REPORTING

6.1 ERROR COMMENT

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED IF NEEDED FROM THE COMMENT AT THE ERROR PC OR FROM THE TEST ITSELF.

6.2 ERROR DATA

*ERRPC	LISTING ADDRESS WHERE THE ERROR WAS DETECTED
*TSTNUM	TEST NUMBER WHERE THE ERROR OCCURRED
BUSADR	DRV11B BUS REG ADDRESS OF CONCERNED OPERATION
EXPCT	DATA THAT WAS EXPECTED
RCVD	DATA THAT WAS RECEIVED
ADRS	MEMORY ADDRESS OF DATA TRANSFER ON ERROR

*ALWAYS REPORTED

7.0 MISCELLANEOUS

7.1 DRV11B BUS & VECTOR ADDRESS MODIFICATION

MODIFY LOCATION '$BASE' IF BASE BUS ADDRESS IS NOT 172410.
MODIFY LOCATION '$VECT1' IF VECTOR ADDRESS IS NOT 124.

*NOTE: USE THE LSI-11 ODT FACILITIES TO MODIFY THESE LOCATIONS AFTER PROGRAM LOAD. NO VECTOR ASSIGNMENT ABOVE 774 SHOULD BE ALLOWED.

7.2 XXDP/APT NOTES

THIS DIAGNOSTIC IS CHAINABLE UNDER XXDP (REF. 7.5)(REQUIRES 8K OR MORE).
THIS DIAGNOSTIC DOES SUPPORT "APT" AND HAS RUN UNDER IT.

7.3 POWER FAIL

A POWER FAILURE WILL CAUSE A RESTART MESSAGE ON POWER UP AT WHICH TIME THE PROGRAM IS RESTARTED (ONLY ON SYSTEMS WITH NON-VOLATILE MEMORY AND WITH APPROPRIATE HARDWARE).

7.4 MULTIPLE DRV11B INTERFACE TESTING

THIS PROGRAM DOES NOT "AUTO-SIZE" THE NUMBER OF DRV11B'S CONNECTED.
THIS DIAGNOSTIC WILL TEST SEQUENTIALLY UP TO 8 DRV11B INTERFACES WITH CONTIGUOUS BUS AND VECTOR ADDRESSES. THIS IS ACCOMPLISHED BY THE OPERATOR SETTING UP LOCATION '$DEVM' WITH A BIT MAP INDICATING WHAT INTERFACES ARE TO TESTED. I.E. BIT0=1 SAYS TEST 1ST DRV11B, BIT1=1 SAYS TEST 2ND DRV11B, BIT2=1 SAYS TEST 3RD DRV11B, ETC..

7.5 RESTRICTIONS

IF THE SYSTEM ALSO INCLUDES AN "REV11" (DMA REFRESH), THE DMA REFRESH MUST BE DISABLED AND CPU REFRESH MUST BE ENABLED.

8.0 EXECUTION TIME

EXECUTION TIME RANGES FROM ABOUT 5 SECONDS WITH NO ITERATIONS TO ABOUT 90 SECONDS WITH ITERATIONS ENABLED WITH ONE DRV11B CONNECTED.
AN END PASS MESSAGE INDICATES ALL TESTS HAVE COMPLETED ON ALL SELECTED UNITS.

9.0 PROGRAM TEST DESCRIPTIONS

9.1 GENERAL

THIS DIAGNOSTIC CONTAINS A SERIES OF INDEPENDENT TESTS DESIGNED TO TEST LOGIC FUNCTIONS AND DATA PATHS OF THE DRV11B DMA INTERFACE. A HIGH DEGREE OF TESTING IS ACCOMPLISHED WITH THE AID OF THE MAINTENANCE LOOP BACK CABLE PROVIDED FOR DIAGNOSTIC TESTING. A COMPLETE LIST OF TESTS IS AVAILABLE IN THE TABLE OF CONTENTS AT THE BEGINNING OF THE LISTING. THE COMMENT FIELD WITHIN EACH TEST CAN BE BENEFICIAL IN TEST UNDERSTANDING.

9.2 REGISTER TESTS

THE FOLLOWING REGISTERS ARE READ/WRITE & RESET TESTED:

1. WORD COUNT
2. BUFFER ADDRESS
3. COMMAND/STATUS
4. DATA BUFFER

9.3 BYTE ADDRESSING TESTS

1. COMMAND/STATUS
2. DATA BUFFER

9.4 'FNCT' TO 'STAT' WRAP AROUND TEST

9.5 READY INTERRUPT TEST

9.6 NPR DATA TRANSFER TESTS

THE FOLLOWING NPR XFERS ARE CHECKED FOR CORRECT STATUS, WORD COUNT, BUFFER ADDRESS & DATA:

1. SINGLE 'DATI' XFER - FLOATING I/O PTRN
2. SINGLE 'DATO' XFER - FLOATING I/O PTRN
3. 200 'DATI' XFERS - FLOATING I/O PTRN
4. 200 'DATO' XFERS - FLOATING I/O PTRN
5. SINGLE 'DATI' XFER TO THE TTY PRINTER CSR

9.7 MAINT MODE NPR DATA TRANSFER TESTS

1. THAT MAINT MODE CONTROLS 'FNCT' BITS
2. 200 MAINT MODE XFERS - CHECKING STATUS & DATA
3. 200 MAINT MODE XFERS TO EACH 4K AVAILABLE MEM

9.8 BURST & NON-BURST MODE TESTS

1. THAT CPU IS LOCKED OUT IN BURST MODE
2. THAT CPU IS NOT LOCKED OUT IN NON-BURST MODE

9.9 'NEX' ERROR CONDITION TEST

```
                                10.0    LISTING
                                        -------

                                !
                                .TITLE  MAINDEC-11-DVDRA-A  DRV11B DMA INTERFACE DIAGNOSTIC
                                ;*COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY R. MOORE
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-CO),MAR 21, 1976.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
167400                          $SWR=167400
000001                          $TN=1
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        11            INHIBIT ITERATIONS
                                ;*        10            BELL ON ERROR
                                ;*         9            LOOP ON ERROR
                                ;*         8            LOOP ON TEST IN SWR<7:0>
                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
```

```
                        R4=     %4              ;;GENERAL REGISTER
                        R5=     %5              ;;GENERAL REGISTER
                        R6=     %6              ;;GENERAL REGISTER
                        R7=     %7              ;;GENERAL REGISTER
                        .EQUIV  R6,SP           ;;STACK POINTER
                        .EQUIV  R7,PC           ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
```

```
                        000200          BIT07=  200
                        000100          BIT06=  100
                        000040          BIT05=  40
                        000020          BIT04=  20
                        000010          BIT03=  10
                        000004          BIT02=  4
                        000002          BIT01=  2
                        000001          BIT00=  1
                                        .EQUIV  BIT09,BIT9
                                        .EQUIV  BIT08,BIT8
                                        .EQUIV  BIT07,BIT7
                                        .EQUIV  BIT06,BIT6
                                        .EQUIV  BIT05,BIT5
                                        .EQUIV  BIT04,BIT4
                                        .EQUIV  BIT03,BIT3
                                        .EQUIV  BIT02,BIT2
                                        .EQUIV  BIT01,BIT1
                                        .EQUIV  BIT00,BIT0

                                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                        000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                        000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                        000014          TBITVEC=14              ;;"T" BIT
                        000014          TRTVEC= 14              ;;TRACE TRAP
                        000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                        000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                        000024          PWRVEC= 24              ;;POWER FAIL
                        000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        000034          TRAPVEC=34              ;;"TRAP" TRAP
                        000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        000064          TPVEC=  64              ;;TTY PRINTER VECTOR
                        000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        172410          ABASE=  172410          ;BASE DRV11B BUS ADRS EQUATE
                        000124          AVECT1= 000124          ;BASE DRV11B VECTOR ADRS EQUATE -
                        000001          ADEVM=  1               ;DEFAULT TO ONE DRV11B
                        106427          MTPS=106427             ;INSTR EQUATE THAT MOVES BYTE TO PSW
                                        .SBTTL  TRAP CATCHER

                        000000                  .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                        000174                  .=174
000174  000000         DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000         SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                        .SBTTL  STARTING ADDRESS(ES)
000200  000137  001544                  JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
                        000100          .=100
000100  000104  000200  000002          .WORD   104,200,2       ;IF 'B EVENT' ON Q BUS IS CONNECTED
                                                                ;IGNORE IT'S INTERRUPT - JUST DO A RTI
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
000106                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  007546                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000106                                  .=$SVPC                 ;; RESTORE PC
001000                                  .=1000
                                .SBTTL  APT PARAMETER BLOCK

                                ;;******************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;******************************************************************
001000                                  .$X=.   ;;SAVE CURRENT LOCATION
000024                                  .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
000044                                  .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
001000                                  .=.$X   ;;RESET LOCATION COUNTER
                                ;;******************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

001000                          $APTHD:
001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001174                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000031                  $TSTM:  .WORD   25.     ;;RUN TIM OF LONGEST TEST
001006  000006                  $PASTM: .WORD   6.      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000144                  $UNITM: .WORD   100.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000052                          .WORD   $EAEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

                                ;;**************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.

001100                                  .=1100
001100                          $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                          .WORD   0
001102  000                     $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000                     $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000                     $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001                     $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134  000                     $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000                     $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  $SWR:   .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
                                $TKS:   177560                  ;;TTY KBD STATUS
                                $TKB:   177562                  ;;TTY KBD BUFFER
                                $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
                                $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000                     $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002                     $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012                     $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000                     $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000                  $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001162  000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001164  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001170  077                     $QUES:  .ASCII  /?/             ;;QUESTION MARK
001171  015                     $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001172  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;**************************************************************
                                .EVEN
001174                          $MAIL:                          ;;APT MAILBOX
001174  000000                  $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001176  000000                  $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001200  000000                  $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001202  000000                  $PASS:  .WORD   APASS           ;;PASS COUNT
001204  000000                  $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
```

```
853  001206  000000                $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
854  001210  000000                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
855  001212  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
856  001214                         $ETABLE:                ;;APT ENVIRONMENT TABLE
857  001214     000                 $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
858  001215     000                 $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
859  001216  000000                 $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
860  001220  000000                 $USWR:  .WORD   AUSWR   ;;USER SWITCHES
861  001222  000000                 $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
862                                 ;*                      BITS 15-11=CPU TYPE
863                                 ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
864                                 ;*                              11/70=06,PDQ=07,Q=10
865                                 ;*                      BIT 10=REAL TIME CLOCK
866                                 ;*                      BIT  9=FLOATING POINT PROCESSOR
867                                 ;*                      BIT  8=MEMORY MANAGEMENT
868  001224     000                 $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
869  001225     000                 $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
870                                 ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
871                                 ;*                              900 NSEC CORE=001
872                                 ;*                              300 NSEC BIPOLAR=002
873                                 ;*                              500 NSEC MOS=003
874  001226  000000                 $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
875                                 ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
876  001230     000                 $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
877  001231     000                 $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
878  001232  000000                 $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
879  001234     000                 $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
880  001235     000                 $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
881  001236  000000                 $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
882  001240     000                 $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
883  001241     000                 $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
884  001242  000000                 $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
885  001244  000124                 $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
886  001246  000000                 $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
887  001250  172410                 $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
888  001252  000001                 $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
889  001254  000000                 $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
890  001256  000000                 $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
891  001260  000000                 $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
892  001262  000000                 $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
893  001264  000000                 $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
894  001266  000000                 $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
895  001270  000000                 $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
896  001272  000000                 $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
897  001274  000000                 $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
898  001276  000000                 $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
899  001300  000000                 $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
900  001302  000000                 $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
901  001304  000000                 $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
902  001306  000000                 $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
903  001310  000000                 $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
904  001312  000000                 $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
905  001314  000000                 $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
906  001316  000000                 $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
```

```
                                
                                
001320                          SETEND:
                                
```

```
                                        .SBTT_  ERROR POINTER TABLE

                                        ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                        ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                        ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                        ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                        ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                        ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                        ;*      DH              ;;POINTS TO THE DATA HEADER
                                        ;*      DT              ;;POINTS TO THE DATA
                                        ;*      DF              ;;POINTS TO THE DATA FORMAT

001320                                  $ERRTB:
                                        ;ERROR  1
001320  013372                                  EM1             ;REG TIMEOUT ER
001322  014263                                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001324  014434                                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001326  000000                                  0

                                        ;ERROR  2
001330  013411                                  EM2             ;REG READ/WRITE ER
001332  014263                                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001334  014434                                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001336  000000                                  0

                                        ;ERROR  3
001340  013433                                  EM3             ;BUS RESET ER
001342  014263                                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001344  014434                                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001346  000000                                  0

                                        ;ERROR  4
001350  013450                                  EM4             ;FNCT BITS FAILED TO SET STAT BITS
001352  014263                                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001354  014434                                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001356  000000                                  0

                                        ;ERROR  5
001360  013512                                  EM5             ;READY INTR FAILURE
001362  014263                                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001364  014434                                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001366  000000                                  0

                                        ;ERROR  6
001370  013535                                  EM6             ;READY CLR OR SET ER
001372  014263                                  DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001374  014434                                  DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001376  000000                                  0
```

```
                                        ;ERROR  7
001400  013561                                EM7             ;STATUS ER ON XFER
001402  014263                                DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001404  014434                                DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001406  000000                                0

                                        ;ERROR  10
001410  013603                                EM10            ;WORD COUNT ER ON XFER
001412  014263                                DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001414  014434                                DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001416  000000                                0

                                        ;ERROR  11
001420  013631                                EM11            ;BUFFER ADRS ER ON XFER
001422  014263                                DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001424  014434                                DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001426  000000                                0

                                        ;ERROR  12
001430  013660                                EM12            ;DATA ER FROM MEM
001432  014330                                DH2             ;ERRPC  TSTNUM  BUSADR  ADRS    EXPCT   RCVD
001434  014450                                DT2             ;$ERRPC TSTNUM  $BDADR  $GDADR  $GDDAT  $BDDAT
001436  000000                                0

                                        ;ERROR  13
001440  013701                                EM13            ;DATA ER TO MEM
001442  014330                                DH2             ;ERRPC  TSTNUM  BUSADR  ADRS    EXPCT   RCVD
001444  014450                                DT2             ;$ERRPC TSTNUM  $BDADR  $GDADR  $GDDAT  $BDDAT
001446  000000                                0

                                        ;ERROR  14
001450  013720                                EM14            ;SINGLE CYCLE OFF DID NOT LOCK OUT CPU
001452  014405                                DH3             ;ERRPC  TSTNUM  BUSADR
001454  014466                                DT3             ;$ERRPC TSTNUM  $BDADR
001456  000000                                0

                                        ;ERROR  15
001460  013766                                EM15            ;SINGLE CYCLE ON LOCKED OUT CPU
001462  014405                                DH3             ;ERRPC  TSTNUM  BUSADR
001464  014466                                DT3             ;$ERRPC TSTNUM  $BDADR
001466  000000                                0

                                        ;ERROR  16
001470  014154                                EM16            ;NEX LOGIC ER
001472  014263                                DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001474  014434                                DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001476  000000                                0

                                        ;ERROR  17
001500  014171                                EM17            ;CYCLE FAILED TO CLK DBR (IN)
001502  014263                                DH1             ;ERRPC  TSTNUM  BDADR   GDDAT   BDDAT
001504  014434                                DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001506  000000                                0
```

```
1014                                    ;ERROR  2U
1015  001510  014226                            EM2O            ;DATA ER FROM I/O PAGE (XCSR)
1016  001512  014330                            DH2             ;ERRPC  TSTNUM  BUSADR  ADRS    EXPCT   RCVD
1017  001514  014450                            DT2             ;$ERRPC TSTNUM  $BDADR  $GDADR  $GDDAT  $BDDAT
1018  001516  000000                            0
1019
1020
1021                                    ;DRV11B BUS REGISTER ADDRESS POINTERS
1022
1023  001520  172410                    DRVWCR: 172410          ;WORD COUNT
1024  001522  172412                    DRVBAR: 172412          ;BUFFER ADDRESS
1025  001524  172414                    DRVCSR: 172414          ;COMMAND/STATUS
1026  001526  172416                    DRVDBR: 172416          ;DATA BUFFER
1027
1028                                    ;DRV11B VECTOR ADDRESS POINTERS
1029
1030  001530  000124                    DRVCT0: 124             ;READY, NEX & INCOMPLETE DATIO VECTOR
1031  001532  000126                    DRVCT2: 126             ;NEW PSW ON INTR
1032
1033                                    ;COMMON PROGRAM LOCATION(S)
1034
1035  001534  000000                    TSTNUM: 0               ;CONTAINS TEST NUMBER ON ERROR
1036  001536  000001                    DMAP:   1               ;DEVICE MAP - EA BIT SAYS TEST THAT DRV11B
1037  001540  000000                    CORSZ:  0               ;CONTAINS 1ST NON-EXISTANT MEM ADRS
1038  001542  014476                    DBUFP:  DBUF            ;CONTAINS CURRENT 4K NPR BUFFER ADRS
```

```
1039                                                    .SBTTL  PROGRAM START
1040  001544                                    START:
1041                                            .SBTTL  INITIALIZE THE COMMON TAGS
1042                                            ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1043  001544  012706  001100                           MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1044  001550  005026                                   CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1045  001552  022706  001140                           CMP     #SWR,R6 ;;DONE?
1046  001556  001374                                   BNE     .-6             ;;LOOP BACK IF NO
1047  001560  012706  001100                           MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1048                                            ;;INITIALIZE A FEW VECTORS
1049  001564  012737  012064  000020                   MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1050  001572  012737  000340  000022                   MOV     #340,@#IOTVEC+2 ;;LEVEL 7
1051  001600  012737  011522  000030                   MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1052  001606  012737  000340  000032                   MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1053  001614  012737  013326  000034                   MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1054  001622  012737  000340  000036                   MOV     #340,@#TRAPVEC+2;LEVEL 7
1055  001630  012737  013122  000024                   MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
1056  001636  012737  000340  000026                   MOV     #340,@#PWRVEC+2 ;;LEVEL 7
1057  001644  005037  001160                           CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
1058  001650  005037  001162                           CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1059  001654  112737  000001  001115                   MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1060  001662  012737  001662  001106                   MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1061  001670  012737  001670  001110                   MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
1062                                            ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1063                                            ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1064  001676  013746  000004                           MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1065  001702  012737  001736  000004                   MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1066  001710  012737  177570  001140                   MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1067  001716  012737  177570  001142                   MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1068  001724  022777  177777  177206                   CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1069  001732  001012                                   BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1070                                                                            ;;AND  THE HARDWARE SWR IS NOT = -1
1071  001734  000403                                   BR      65$             ;;BRANCH IF NO TIMEOUT
1072  001736  012716  001744          64$:     MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1073  001742  000002                                   RTI
1074  001744  012737  000176  001140  65$:     MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1075  001752  012737  000174  001142                   MOV     #DISPREG,DISPLAY
1076  001760  012637  000004          66$:     MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1077
1078  001764  005037  001202                           CLR     $PASS           ;;CLEAR PASS COUNT
1079  001770  132737  000200  001215                   BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
1080  001776  001403                                   BEQ     67$             ;;YES,USE NON-APT SWITCH
1081  002000  012737  001216  001140                   MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
1082  002006                                    67$:
1083  002006  012700  001520          START1:  MOV     #DRVWCR,R0      ;SET UP REG ADRS POINTERS
1084  002012  013701  001250                           MOV     $BASE,R1        ;GET BASE ADRS
1085  002016  010120                   SETUP2:  MOV     R1,(R0)+        ;LOAD EM
1086  002020  062701  000002                           ADD     #2,R1           ;
1087  002024  022700  001530                           CMP     #DRVOBR+2,R0    ;ALL DONE?
1088  002030  001372                                   BNE     SETUP2          ;BR IF NOT
1089  002032  012700  001530                           MOV     #DRVCTO,R0      ;SET UP DRV11B VECTOR ADRS POINTER
1090  002036  013701  001244                           MOV     $VECT1,R1       ;GET BASE VECTOR ADRS
1091  002042  042701  170000                           BIC     #170000,R1      ;CLR OUT PRIORITY BITS
1092  002046  010120                   SETUP3:  MOV     R1,(R0)+        ;
```

```
1093  002050  062701  000002                        ADD     #2,R1           ;POINT TO NEXT
1094  002054  022700  001534                        CMP     #DRVCT2+2,R0    ;ALL DONE?
1095  002060  001372                                BNE     SETUP3          ;BR IF NOT
1096                                        .SBTTL  TYPE PROGRAM NAME
1097                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1098  002062  005227  177777                        INC     #-1             ;;FIRST TIME?
1099  002066  001052                                BNE     64$             ;;BRANCH IF NO
1100  002070  104400  002136                        TYPE    ,65$            ;;TYPE ASCIZ STRING
1101                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1102  002074  005737  000042                        TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1103  002100  001012                                BNE     66$             ;;BRANCH IF YES
1104  002102  123727  001214  000001                CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
1105  002110  001406                                BEQ     66$             ;;BRANCH IF YES
1106  002112  023727  001140  000176                CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1107  002120  001005                                BNE     67$             ;;BRANCH IF NO
1108  002122  104405                                GTSWR                   ;;GET SOFT-SWR SETTINGS
1109  002124  000403                                BR      67$
1110  002126  112737  000001  001134  66$:   MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1111  002134                                67$:
1112  002134  000427                                BR      64$             ;;GET OVER THE ASCIZ
1113                                        ;;65$:  .ASCIZ  <CRLF>#MD-11-DVDRA-A  DRV11B DMA INTERFACE DIAG  #<CRLF>
1114  002214                                64$:
1115  002214  005737  001540                        TST     CORSZ           ;TEST IF FIRST PASS
1116  002220  001002                                BNE     CORSZR          ;BR IF NOT
1117  002222  104400  014025                        TYPE,   WARN            ;TELL THE OPERATOR TO TURN OFF DMA REFRESH
1118                                        ;;**************************************************************
1119                                        ;LET'S SEE HOW MUCH MEM WE HAVE
1120                                        ;;**************************************************************
1121  002226  012700  020000                CORSZR: MOV     #20000,R0       ;USE R0 TO LOOK
1122  002232  012737  002244  000004                MOV     #2$,@#ERRVEC    ;SET UP TIME OUT RETURN ADRS
1123  002240  005720                        1$:     TST     (R0)+           ;TAKE A LOOK
1124  002242  000776                                BR      1$              ;UNTIL TIMEOUT
1125  002244  042700  017777                2$:     BIC     #17777,R0       ;POINT TO 1ST NON-EXSISTANT 4K BLK
1126  002250  010037  001540                        MOV     R0,CORSZ        ;SAVE FOR LATER
1127  002254  012737  000006  000004                MOV     #ERRVEC+2,@#ERRVEC  ;RESTORE VECTOR
1128  002262  012737  014476  001542                MOV     #DBUF,DBUFP     ;INITIALIZE TO LOWEST 4K
1129  002270  012737  000000  001206                MOV     #0,$UNIT        ;SET UP UNIT COUNT
1130  002276  013737  001252  001536                MOV     $DEVM,DMAP      ;GET THE # & POSITION OF DRV11B'S
1131  002304  042737  177400  001536                BIC     #177400,DMAP    ;UP TO 8 ONLY
1132  002312  001406                                BEQ     RESTRT          ;GO CONTINUE AS IF SOMETHING WAS SELECTED
1133  002314  032737  000001  001536                BIT     #1,DMAP         ;IS 1ST DRV11B SELECTED?
1134  002322  001002                                BNE     RESTRT          ;BR IF SO
1135  002324  000137  007364                        JMP     NXDEV1          ;NO - GO ADVANCE BASE DRV11B ADDRESSES
1136  002330  106427  000200                RESTRT: MTPS,   200             ;SET PRIORITY TO HIGHEST LEVEL
1137  002334  012706  001100                        MOV     #STACK,SP       ;ALWAYS RESET STACK PTR
1138  002340  013737  001206  001204                MOV     $UNIT,$DEVCT    ;LOAD APT COUNTER
1139  002346  013700  001206                        MOV     $UNIT,R0        ;MAKE AN INDEX
1140  002352  006300                                ASL     R0              ; VALUE
1141  002354  013760  001520  001260                MOV     DRVWCR,$DDWO(R0)        ;SAVE THE BUS ADDRESS
1142  002362  000005                                RESET                   ;INITIALIZE DRV11B BEFORE TESTING
```

```
                                        ;;*****************************************************************
                                        ;*TEST 1         TEST THAT ALL DRV11B REGS ARE ACCESSIBLE
                                        ;;*****************************************************************
002364  000240                  TST1:   <NOP>
002366  012737  002402  001106          MOV     #10$,$LPADR     ;;SET SCOPE LOOP ADDRESS
002374  012737  000001  001200          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
002402  112737  000001  001102  10$:     MOVB    #1,$TSTNM       ;SET TO TEST #1
002410  012737  002444  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
002416  005037  001124                  CLR     $GDDAT          ;NO DATA COMPARE
002422  005037  001126                  CLR     $BDDAT          ;NO DATA COMPARE
002426  012737  002462  000004          MOV     #2$,@#ERRVEC    ;SET UP TIMEOUT RETURN ADDRS
002434  013700  001520                  MOV     DRVWCR,R0       ;SET UP 1ST DRV11 BUS ADRS
002440  012701  000004                  MOV     #4,R1           ;SET UP REG COUNT
002444  010037  001122          1$:     MOV     R0,$BDADR       ;SET UP CURRENT DRV BUS ADRS
002450  005710                          TST     (R0)            ;SEE IF THERE
002452  005720                          TST     (R0)+           ;BUMP TO NEXT
002454  005301                          DEC     R1              ;COUNT 4 OF THEM
002456  001403                          BEQ     3$              ;BR IF ALL DONE
002460  000771                          BR      1$              ;TRY NEXT
002462  022626                  2$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
002464  104001                          ERROR   1               ;BUS ADRS INDICATED DID NOT RESPOND
002466  012737  000006  000004  3$:     MOV     #ERRVEC+2,@#ERRVEC  ;RESTORE LOC 4

                                        ;;*****************************************************************
                                        ;*TEST 2         TEST THAT THE WORD COUNT REG IS WRITE/READABLE (FLOAT 0 COM PTRN)
                                        ;;*****************************************************************
002474  000004                  TST2:   SCOPE
002476  012737  002522  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
002504  013737  001520  001122          MOV     DRVWCR,$BDADR   ;SET UP WC REG ADRS
002512  005000                          CLR     R0              ;R0 SAYS SHIFT PTRN WHEN 0
002514  012737  177776  001124          MOV     #-2,$GDDAT      ;FLOAT 0 RIGHT TO LEFT
002522  013777  001124  176770  1$:     MOV     $GDDAT,@DRVWCR  ;LD WC
002530  017737  176764  001126          MOV     @DRVWCR,$BDDAT  ;READ IT BACK
002536  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
002544  001401                          BEQ     2$              ;BR IF SO
002546  104002                          ERROR   2               ;WORD COUNT WRITE/READ FAILURE
002550  005137  001124          2$:     COM     $GDDAT          ;COMPELEMENT ZERO
002554  005100                          COM     R0              ;R0 SAYS SHIFT LEFT WHEN = 0
002556  001361                          BNE     1$              ;TRY THE COMPLEMENT IF R0 NOT 0
002560  006337  001124                  ASL     $GDDAT          ;COMPLEMENT WAS DONE - NOW SHIFT ZERO LEFT
002564  005237  001124                  INC     $GDDAT          ;KEEP LSB SET
002570  103754                          BCS     1$              ;AGAIN TILL ALL PATRNS DONE

                                        ;;*****************************************************************
                                        ;*TEST 3         TEST THAT THE BUFFER ADDRESS REG IS WRITE/READABLE (FLOAT 0 COM PTRN)
                                        ;;*****************************************************************
002572  000004                  TST3:   SCOPE
002574  012737  002620  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
002602  013737  001522  001122          MOV     DRVBAR,$BDADR   ;SET UP BA REG ADRS
002610  005000                          CLR     R0              ;R0 SAYS SHIFT PTRN WHEN 0
002612  012737  177774  001124          MOV     #-4,$GDDAT      ;FLOAT 0 RIGHT TO LEFT
002620  013777  001124  176674  1$:     MOV     $GDDAT,@DRVBAR  ;LD BA
002626  017737  176670  001126          MOV     @DRVBAR,$BDDAT  ;READ IT BACK
002634  042737  000001  001126          BIC     #BIT00,$BDDAT   ;DON'T WANT BIT00
```

```
1197  002642  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1198  002650  001401                          BEQ     2$              ;BR IF SO
1199  002652  104002                          ERROR   2               ;BUS ADRS WRITE/READ FAILURE
1200  002654  005137  001124          2$:     COM     $GDDAT          ;COMPLEMENT ZERO
1201  002660  042737  000001  001124          BIC     #BIT00,$GDDAT   ;BIT 00 NOT INVOLVED
1202  002666  005100                          COM     R0              ;R0 SAYS SHIFT LEFT WHEN = 0
1203  002670  001353                          BNE     1$              ;TRY THE COMPLEMENT IF R0 NOT 0
1204  002672  006337  001124                  ASL     $GDDAT          ;COMPLEMENT WAS DONE - NOW SHIFT ZERO LEFT
1205  002676  103004                          BCC     TST4            ;;NEXT TEST IF BIT 15 DONE
1206  002700  062737  000002  001124          ADD     #2,$GDDAT       ;KEEP ADDR LSB SET
1207  002706  000744                          BR      1$              ;AGAIN TILL ALL PATTERNS DONE
1208
1209                                  ;;*****************************************************************
1210                                  ;*TEST 4         TEST THAT THE DATA BUFFER REG IS WRITE/READABLE (FLOAT 0 COM PTRN)
1211                                  ;;*****************************************************************
1212  002710  000004                  TST4:   SCOPE
1213  002712  012737  002736  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
1214  002720  013737  001526  001122          MOV     DRVDBR,$BDADR   ;SET UP DB REG ADRS
1215  002726  005000                          CLR     R0              ;R0 SAYS SHIFT PTRN WHEN 0
1216  002730  012737  177776  001124          MOV     #-2,$GDDAT      ;FLOAT 0 RIGHT TO LEFT
1217  002736  013777  001124  176562  1$:     MOV     $GDDAT,@DRVDBR  ;LD DB
1218  002744  017737  176556  001126          MOV     @DRVDBR,$BDDAT  ;READ IT BACK
1219  002752  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1220  002760  001401                          BEQ     2$              ;BR IF SO
1221  002762  104002                          ERROR   2               ;DATA BUFFER WRITE/READ FAILURE (LOOP BACK)
1222  002764  005137  001124          2$:     COM     $GDDAT          ;COMPLEMENT ZERO
1223  002770  005100                          COM     R0              ;R0 SAYS SHIFT LEFT WHEN = 0
1224  002772  001361                          BNE     1$              ;TRY THE COMPLEMENT IF R0 NOT 0
1225  002774  006337  001124                  ASL     $GDDAT          ;COMPLEMENT WAS DONE - NOW SHIFT ZERO LEFT
1226  003000  005237  001124                  INC     $GDDAT          ;KEEP LSB SET
1227  003004  103754                          BCS     1$              ;AGAIN TILL ALL PATRNS DONE
1228
1229                                  ;;*****************************************************************
1230                                  ;*TEST 5         TEST THAT THE DATA BUFFER REG IS BYTE ADDRESSABLE
1231                                  ;;*****************************************************************
1232  003006  000004                  TST5:   SCOPE
1233  003010  013700  001526                  MOV     DRVDBR,R0       ;GET DB REG ADRS
1234  003014  010037  001122                  MOV     R0,$BDADR       ;SET UP DB REG ADRS
1235  003020  005010                          CLR     (R0)            ;ZERO DATA BUFFER REG
1236  003022  012737  177177  001124          MOV     #177177,$GDDAT  ;LD EXPECTED
1237  003030  012737  077776  001126          MOV     #77776,$BDDAT   ;SEND DATA FROM "BDDAT"
1238  003036  153760  001126  000001          BISB    $BDDAT,1(R0)    ;LOAD HI BYTE DB
1239  003044  153710  001127                  BISB    $BDDAT+1,(R0)   ;LOAD LO BYTE DB
1240  003050  011037  001126                  MOV     (R0),$BDDAT     ;READ IT BACK
1241  003054  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1242  003062  001401                          BEQ     TST6            ;;NEXT TEST IF SO
1243  003064  104002                          ERROR   2               ;DATA ERROR ON BYTE ADDRESSING THE DATA BUFFER REG
1244
1245                                  ;;*****************************************************************
1246                                  ;*TEST 6         TEST THAT RESET CLEARS WORD COUNT, BUS ADDRESS & DATA REGS
1247                                  ;;*****************************************************************
1248  003066  000004                  TST6:   SCOPE
1249  003070  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1250  003076  005037  001124                  CLR     $GDDAT          ;LD EXPECTED
```

```
1251  003102  012777  177777  176410          MOV     #-1,@DRVWCR     ;SET ALL BITS - WC REG
1252  003110  012777  177776  176404          MOV     #-2,@DRVBAR     ;SET ALL BITS - BUS ADRS REG
1253  003116  012777  177777  176402          MOV     #-1,@DRVDBR     ;SET ALL BITS - DB OUT REG
1254  003124  000005                          RESET                   ;DO A BUS RESET
1255  003126  017737  176366  001126          MOV     @DRVWCR,$BDDAT  ;READ WC REG
1256  003134  001404                          BEQ     1$              ;BR IF CLRED
1257  003136  013737  001520  001122          MOV     DRVWCR,$BDADR   ;SET UP WC REG ADRS
1258  003144  104003                          ERROR   3               ;RESET FAILED TO CLR WC REG
1259  003146  017737  176350  001126  1$:     MOV     @DRVBAR,$BDDAT  ;READ BUS ADRS REG
1260  003154  042737  000001  001126          BIC     #BIT00,$BDDAT   ;DON'T WANT BIT00
1261  003162  001404                          BEQ     2$              ;BR IF CLRED
1262  003164  013737  001522  001122          MOV     DRVBAR,$BDADR   ;SET UP BA REG ADRS
1263  003172  104003                          ERROR   3               ;RESET FAILED TO CLR BUS ADRS REG
1264  003174  017737  176326  001126  2$:     MOV     @DRVDBR,$BDDAT  ;READ DATA BUFFER REG
1265  003202  001404                          BEQ     TST7            ;;NEXT TEST IF CLRED
1266  003204  013737  001526  001122          MOV     DRVDBR,$BDADR   ;SET UP DB REG ADRS
1267  003212  104003                          ERROR   3               ;RESET FAILED TO CLR DATA BUFFER OUT REG
1268
1269                                  ;;**********************************************************************
1270                                  ;*TEST 7         TEST THAT THE CONTROL/STATUS REG IS WRITE/READABLE (COUNT PTRN)
1271                                  ;;**********************************************************************
1272  003214  000004                  TST7:   SCOPE
1273  003216  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1274  003224  106427  000200                  MTPS,   200             ;DON'T WANT ANY INTRS
1275  003230  004537  007602                  JSR     R5,SETVEC       ;SET UP INTR RETURN ADRS IN CASE
1276  003234  003332                          3$                      ;RETURN TO 3$ ON ILLEGAL INTR
1277  003236  013737  003256  001110          MOV     1$,$LPERR       ;SET UP SCOPE LOOP ADRS
1278  003244  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1279  003252  012700  160000                  MOV     #160000,R0      ;START AT 0 - HI BITS FOR NOISE
1280  003256  010037  001124          1$:     MOV     R0,$GDDAT       ;LD EXPECTED
1281  003262  042737  167201  001124          BIC     #167201,$GDDAT  ;MASK TO WRITEABLE BITS
1282  003270  010077  176230                  MOV     R0,@DRVCSR      ;LD CSR
1283  003274  017737  176224  001126          MOV     @DRVCSR,$BDDAT  ;READ IT BACK
1284  003302  042737  007200  001126          BIC     #7200,$BDDAT    ;DON'T LOOK AT STAT & RDY BITS
1285  003310  023737  001 24  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1286  003316  001401                          BEQ     2$              ;BR IF SO
1287  003320  104002                          ERROR   2               ;CONTROL/STATUS REG WRITE/READ FAILURE
1288  003322  062700  000002          2$:     ADD     #2,R0           ;ADVANCE COUNT PATTERN
1289  003326  001353                          BNE     1$              ;WRITE NEXT PATTERN IF NOT ALL TESTED
1290  003330  000413                          BR      4$              ;GO RESTORE VECTOR
1291  003332  022626                  3$:     CMP     (SP)+,(SP)+     ;FIX STACK - SHOULD NOT HAVE INTR'ED
1292  003334  052737  000200  001124          BIS     #200,$GDDAT     ;CORRECT EXPECTED
1293  003342  017737  176156  001126          MOV     @DRVCSR,$BDDAT  ;READ CSR
1294  003350  042737  007000  001126          BIC     #7000,$BDDAT    ;DON'T WANT 'STAT' BITS
1295  003356  104005                          ERROR   5               ;CPU FAILED TO LOCK OUT DRV11B INTR REQ
1296  003360  004737  007622          4$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1297
1298                                  ;;**********************************************************************
1299                                  ;*TEST 10        TEST  HAT RESET CLEARS ALL WRITEABLE BITS & SET READY IN CSR
1300                                  ;;**********************************************************************
1301  003364  000004                  TST10:  SCOPE
1302  003366  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1303  003374  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1304  003402  012737  000200  001124          MOV     #200,$GDDAT     ;LD EXPECTED
```

```
1305  003410  012777  177776  176106          MOV     #-2,@DRVCSR     ;LD ALL CSR BITS
1306  003416  000005                          RESET                   ;DO A BUS RESET
1307  003420  017737  176100  001126          MOV     @DRVCSR,$BDDAT  ;READ CSR
1308  003426  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1309  003434  001401                          BEQ     .ST11           ;;NEXT TEST IF CSR CLRED & READY SET
1310  003436  104003                          ERROR   3               ;RESET FAILED TO SET UP THE CSR
1311
1312                                  ;;*****************************************************************
1313                                  ;*TEST 11        TEST THAT THE CSR IS BYTE ADDRESSABLE
1314                                  ;;*****************************************************************
1315  003440  000004                  .ST11:  SCOPE
1316  003442  013700  001524                  MOV     DRVCSR,R0       ;GET CSR ADRS
1317  003446  010037  001122                  MOV     R0,$BDADR       ;SET UP CSR ADRS
1318  003452  005010                          CLR     (R0)            ;ZERO CSR
1319  003454  012737  010300  001124          MOV     #10300,$GDDAT   ;LD EXPECTED
1320  003462  012737  040020  001126          MOV     #40020,$BDDAT   ;SEND DATA FROM "BDDAT" - USE "MAIN + IE
1321  003470  153760  001126  000001          BISB    $BDDAT,1(R0)    ;LOAD HI BYTE CSR
1322  003476  153710  001127                  BISB    $BDDAT+1,(R0)   ;LOAD LO BYTE CSR
1323  003502  011037  001126                  MOV     (R0),$BDDAT     ;READ IT BACK
1324  003506  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1325  003514  001401                          BEQ     1$              ;BR IF SO
1326  003516  104002                          ERROR   2               ;DATA ERROR ON BYTE ADDRESSING THE CSR
1327  003520  005010                  1$:     CLR     (R0)            ;ZERO CSR BEFORE ADVANCING
1328
1329                                  ;;*****************************************************************
1330                                  ;*TEST 12        TEST THAT THE 3 "FNCT" BITS CONTROL THE 3 "STAT" BITS (COUNT PTRN)
1331                                  ;;*****************************************************************
1332  003522  000004                  .ST12:  SCOPE
1333  003524  012737  003552  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
1334  003532  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1335  003540  012737  007000  001124          MOV     #7000,$GDDAT    ;LD EXPECTED
1336  003546  012700  000016                  MOV     #16,R0          ;R0 CONTAINS "FNCT" BITS WRITTEN
1337  003552  010077  175746          1$:     MOV     R0,@DRVCSR      ;WRITE INTO "FNCT" BITS
1338  003556  017737  175742  001126          MOV     @DRVCSR,$BDDAT  ;READ BACK THRU "STAT" BITS
1339  003564  042737  170777  001126          BIC     #170777,$BDDAT  ;MASK TO "STAT" BITS ONLY
1340  003572  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1341  003600  001401                          BEQ     2$              ;BR IF SO
1342  003602  104004                          ERROR   4               ;"FNCT" BITS FAILED TO SET "STAT" BITS (LOOP BACK)
1343  003604  162737  001000  001124  2$:     SUB     #1000,$GDDAT    ;CHANGE TO NEXT EXPECTED
1344  003612  162700  000002                  SUB     #2,R0           ;DECREASE COUNT PATTERN
1345  003616  100355                          BPL     1$              ;DO AGAIN UNTIL 0 TESTED
1346
1347                                  ;;*****************************************************************
1348                                  ;*TEST 13        TEST THAT READY SET WILL CAUSE AN INTERRUPT AT LEVEL 0
1349                                  ;;*****************************************************************
1350  003620  000004                  .ST13:  SCOPE
1351  003622  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1352  003630  106427  000200                  MTPS    200             ;DONT WANT INTR YET
1353  003634  000005                          RESET                   ;SET THE READY FLAG BY INIT
1354  003636  012777  003716  175664          MOV     #1$,@DRVCTO     ;SET UP PREMATURE INTR RETURN ADRS
1355  003644  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1356  003652  012737  000300  001124          MOV     #300,$GDDAT     ;LD EXPECTED (READY + IE)
1357  003660  106427  000000                  MTPS    0               ;ALLOW AN INTR
1358  003664  021616                          CMP     (SP),(SP)       ;STALL
```

```
1359  003666  012777  003732  175631          MOV     #2$,@DRVCTO     ;SET UP EXPECTED INTR RETURN ADRS
1360  003674  052777  000100  175622          BIS     #BIT6,@DRVCSR   ;ENABLE THE EXPECTED INTERRUPT
1361  003702  021616                          CMP     (SP),(SP)       ;STALL
1362  003704  017737  175614  001126          MOV     @DRVCSR,$BDDAT  ;GET THE CSR
1363  003712  104005                          ERROR   5               ;READY FAILED TO CAUSE AN INTERRUPT
1364  003714  000417                          BR      3$              ;GO RESTORE VECTOR
1365  003716  022626                  1$:     CMP     (SP)+,(SP)+     ;SHOULD NEVER GET HERE - IE NOT WORKING?
1366  003720  017737  175600  001126          MOV     @DRVCSR,$BDDAT  ;GET THE CSR
1367  003726  104005                          ERROR   5               ;READY INTERRUPTED WITHOUT THE IE BIT
1368  003730  000411                          BR      3$              ;GO RESTORE VECTOR
1369  003732  022626                  2$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RETURN
1370  003734  017737  175564  001126          MOV     @DRVCSR,$BDDAT  ;READ STATUS
1371  003742  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1372  003750  001401                          BEQ     3$              ;BR IF SO
1373  003752  104005                          ERROR   5               ;INCORRECT STATUS ON READY INTR
1374  003754  004737  007622          3$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1375
1376                                  ;;**********************************************************************
1377                                  ;*TEST 14        TEST THAT GO CLRS READY & FNCT 2 WILL SET IT
1378                                  ;;**********************************************************************
1379  003760  000004                  TST14:  SCOPE
1380  003762  106427  000200                  MTPS,   200             ;DONT WANT ANY INTRS
1381  003766  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1382  003774  005037  001124                  CLR     $GDDAT          ;EXPECT 0
1383  004000  012777  000001  175516          MOV     #1,@DRVCSR      ;SET GO WHICH SHOULD CLR READY
1384  004006  017737  175512  001126          MOV     @DRVCSR,$BDDAT  ;READ THE CSR
1385  004014  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1386  004022  001401                          BEQ     1$              ;BR IF SO
1387  004024  104006                          ERROR   6               ;THE GO BIT FAILED TO CLR READY
1388  004026  052777  000004  175470  1$:     BIS     #4,@DRVCSR      ;FNCT 2 SHOULD SET READY
1389  004034  012737  002204  001124          MOV     #2204,$GDDAT    ;LD EXPECTED
1390  004042  017737  175456  001126          MOV     @DRVCSR,$BDDAT  ;GET CSR
1391  004050  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1392  004056  001401                          BEQ     TST15           ;;NEXT TEST IF SET
1393  004060  104006                          ERROR   6               ;FNCT 2 (VIA ATTN) FAILED TO SET READY
1394
1395                                  ;;**********************************************************************
1396                                  ;*TEST 15        TEST THAT READY CONTROLS 'BAR' BIT00
1397                                  ;;**********************************************************************
1398  004062  000004                  TST15:  SCOPE
1399  004064  012777  000004  175432          MOV     #4,@DRVCSR      ;SET READY
1400  004072  013737  001522  001122          MOV     DRVBAR,$BDADR   ;SET UP BAR ADRS
1401  004100  012737  000001  001124          MOV     #1,$GDDAT       ;EXPECT LSB OF BAR
1402  004106  005077  175410                  CLR     @DRVBAR         ;CLR BAR
1403  004112  017737  175404  001126          MOV     @DRVBAR,$BDDAT  ;READ BAR
1404  004120  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1405  004126  001401                          BEQ     1$              ;BR IF SO
1406  004130  104002                          ERROR   2               ;ADD FAILED TO READ A ONE(SB TIED TO RDY)
1407  004132  012777  000001  175364  1$:     MOV     #1,@DRVCSR      ;SET GO(CLRS BAR BIT00)
1408  004140  017737  175356  001126          MOV     @DRVBAR,$BDDAT  ;READ BAR
1409  004146  001403                          BEQ     2$              ;BR IF ZERO
1410  004150  005037  001124                  CLR     $GDDAT          ;EXPECTED ZERO
1411  004154  104002                          ERROR   2               ;WHEN RDY CLRED-ADD FAILED TO READ A ZERO
1412  004156  012777  000004  175340  2$:     MOV     #4,@DRVCSR      ;INSURE RDY SET BEFORE ADVANCING
```

```
1413
1414                                    ;;*****************************************************************
1415                                    ;*TEST 16       TEST THAT 'CYCLE' WILL CLOCK THE DBR (IN)
1416                                    ;;*****************************************************************
1417  004164  000004                    $ST16:  SCOPE
1418  004166  013737  001526  001122            MOV     DRVDBR,$BDADR   ;SET UP DBR ADRS
1419  004174  012737  125252  001124            MOV     #125252,$GDDAT  ;LD EXPECTED
1420  004202  013777  001124  175316            MOV     $GDDAT,@DRVDBR  ;LD DBR WITH #125252
1421  004210  012777  000400  175306            MOV     #400,@DRVCSR    ;SET CYCLE - SHOULD CLK DBR (IN)
1422  004216  012777  052525  175302            MOV     #52525,@DRVDBR  ;CHANGE DBR (OUT) DATA - SHOULD NOT AFFECT (IN)
1423  004224  017737  175276  001126            MOV     @DRVDBR,$BDDAT  ;READ DBR
1424  004232  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
1425  004240  001401                            BEQ     1$              ;BR IF SO
1426  004242  104017                            ERROR   17              ;CYCLE DID NOT LATCH DBR (IN) DATA
1427  004244  042777  000400  175252  1$:       BIC     #400,@DRVCSR    ;REMOVE CYCLE
1428  004252  012737  052525  001124            MOV     #52525,$GDDAT   ;NOW EXPECT #52525
1429  004260  017737  175242  001126            MOV     @DRVDBR,$BDDAT  ;READ DBR
1430  004266  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
1431  004274  001401                            BEQ     TST17           ;;NEXT TEST IF SO
1432  004276  104002                            ERROR   2               ;DBR FAILED TO READ WHEN CYCLE CLRED (NORMAL)
1433
1434                                    ;;*****************************************************************
1435                                    ;*TEST 17       TEST SINGLE "DATI" NPR TRANSFERS (FLOATING 0 COMPLEMENT PATRN)
1436                                    ;;*****************************************************************
1437  004300  000004                    $ST17:  SCOPE
1438  004302  012737  004324  001110            MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
1439  004310  004537  007602                    JSR     R5,SETVEC       ;GO SET UP INTERRUPT RETURN
1440  004314  004420                            2$                      ;RETURN TO 2$ ON INTR
1441  004316  012700  177776                    MOV     #-2,R0          ;FLOAT ZERO RIGHT TO LEFT
1442  004322  005001                            CLR     R1              ;R1 CONTROLS DATA SHIFTING
1443  004324  012777  177777  175166  1$:       MOV     #-1,@DRVWCR     ;DO ONE XFER
1444  004332  012777  014476  175162            MOV     #DBUF,@DRVBAR   ;GET DATA WORD FROM "DBUF"
1445  004340  010037  014476                    MOV     R0,DBUF         ;SET UP MEM DATA
1446  004344  012777  000101  175152            MOV     #101,@DRVCSR    ;SET IE & GO
1447  004352  052777  000400  175144            BIS     #400,@DRVCSR    ;SET CYCLE
1448  004360  106427  000000                    MTPS,   0               ;ENABLE THE INTR
1449  004364  013737  001524  001122            MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1450  004372  012737  000700  001124            MOV     #700,$GDDAT     ;LD EXPECTED
1451  004400  017737  175120  001126            MOV     @DRVCSR,$BDDAT  ;READ THE CSR
1452  004406  042777  000100  175110            BIC     #100,@DRVCSR    ;CLR IE
1453  004414  104005                            ERROR   5               ;WCO FAILED TO INTERRUPT (CHECK FOR WCO)
1454  004416  000442                            BR      4$              ;GO RESTORE VECTOR
1455  004420  022626                    2$:     CMP     (SP)+,(SP)+     ;INTR RETURNS HERE - FIX STACK SINCE NO RTI
1456  004422  004537  007646                    JSR     R5,CKSTAT       ;GO CHECK STATUS
1457  004426  000700                            700                     ;CSR STATUS EXPECTED
1458  004430  000001                            1                       ;# OF XFERS
1459  004432  104007                            ERROR   7               ;RETURN HERE IF STATUS ER - EXPECTED CYCLE, READY & IE
1460  004434  000433                            BR      4$              ;GO RESTORE VECTOR
1461  004436  104010                            ERROR   10              ;RETURN HERE IF WC ER - EXPECTED 0
1462  004440  000431                            BR      4$              ;GO RESTORE VECTOR
1463  004442  104011                            ERROR   11              ;RETURN HERE IF BAR ER - SHOULD = DBUF+2
1464  004444  000427                            BR      4$              ;GO RESTORE VECTOR
1465  004446  012737  014476  001120            MOV     #DBUF,$GDADR    ;RETURN HERE IF OK - SET UP XFER ADRS
1466  004454  010037  001124                    MOV     R0,$GDDAT       ;LD EXPECTED
```

```
1467  004460  017737  175042  001126          MOV     @DRVDBR,$BDDAT  ;READ DATA XFERED
1468  004466  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1469  004474  001405                          BEQ     3$              ;BR IF SO
1470  004476  013737  001526  001122          MOV     DRVDBR,$BDADR   ;SET UP DBR ADRS
1471  004504  104012                          ERROR   12              ;DATA ER - DBR CONTAINS WRONG DATA
1472  004506  000406                          BR      4$              ;GO RESTORE VECTOR
1473  004510  005100                  3$:     COM     R0              ;RETURN HERE ON GOOD DATA - NOW COM PATRN
1474  004512  005101                          COM     R1              ;KEEP TRACK OF COMPLEMENT
1475  004514  001303                          BNE     1$              ;DO COMPLEMENT OF THIS FLOATING ZERO IF C
1476  004516  006300                          ASL     R0              ;WAS DONE - NOW SHIFT ZERO LEFT
1477  004520  005200                          INC     R0              ;KEEP LSB SET
1478  004522  103700                          BCS     1$              ;AGAIN TILL ZERO BIT IN CARRY
1479  004524  004737  007622          4$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1480
1481                                  ;;*****************************************************************
1482                                  ;*TEST 20       TEST SINGLE "DATO" NPR TRANSFERS (FLOATING 0 COMPLEMENT PATRN)
1483                                  ;;*****************************************************************
1484  004530  000004                  TST20:  SCOPE
1485  004532  012737  004554  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
1486  004540  004537  007602                  JSR     R5,SETVEC       ;GO SET UP INTERRUPT RETURN
1487  004544  004642                          2$                      ;RETURN TO 2$ ON INTR
1488  004546  012700  177776                  MOV     #-2,R0          ;FLOAT ZERO RIGHT TO LEFT
1489  004552  005001                          CLR     R1              ;R1 CONTROLS DATA SHIFTING
1490  004554  012777  177777  174736  1$:     MOV     #-1,@DRVWCR     ;DO ONE XFER
1491  004562  012777  014476  174732          MOV     #DBUF,@DRVBAR   ;WRITE DATA WORD TO "DBUF"
1492  004570  010077  174732                  MOV     R0,@DRVDBR      ;SET UP DATA IN DBR
1493  004574  012777  000103  174722          MOV     #103,@DRVCSR    ;SET IE, GO & FNCT1 (C1 CONTROL)
1494  004602  052777  000400  174714          BIS     #400,@DRVCSR    ;SET CYCLE
1495  004610  106427  000000                  MTPS,   0               ;ENABLE THE INTR
1496  004614  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1497  004622  012737  001702  001124          MOV     #1702,$GDDAT    ;LD EXPECTED
1498  004630  017737  174670  001126          MOV     @DRVCSR,$BDDAT  ;READ THE CSR
1499  004636  104005                          ERROR   5               ;WCO FAILED TO INTERRUPT (CHECK FOR WCO)
1500  004640  000442                          BR      4$              ;GO RESTORE VECTOR
1501  004642  022626                  2$:     CMP     (SP)+,(SP)+     ;INTR RETURNS HERE - FIX STACK SINCE NO RTI
1502  004644  004537  007646                  JSR     P5,CKSTAT       ;GO CHECK STATUS
1503  004650  001702                          1702                    ;CSR STATUS EXPECTED
1504  004652  000001                          1                       ;# OF XFERS
1505  004654  104007                          ERROR   7               ;RETURN HERE IF STATUS ER - EXPECTED STAT C,
1506                                                                  ;CYCLE, READY, IE & FNCT 1
1507  004656  000433                          BR      4$              ;GO RESTORE VECTOR
1508  004660  104010                          ERROR   10              ;RETURN HERE IF WC ER - EXPECTED 0
1509  004662  000431                          BR      4$              ;GO RESTORE VECTOR
1510  004664  104011                          ERROR   11              ;RETURN HERE IF BAR ER - SHOULD = DBUF+2
1511  004666  000427                          BR      4$              ;GO RESTORE VECTOR
1512  004670  012737  014476  001120          MOV     #DBUF,$GDADR    ;RETURN HERE IF OK - SET UP XFER ADRS
1513  004676  010037  001124                  MOV     R0,$GDDAT       ;LD EXPECTED
1514  004702  013737  014476  001126          MOV     DBUF,$BDDAT     ;GET DATA XFERED
1515  004710  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1516  004716  001405                          BEQ     3$              ;BR IF SO
1517  004720  013737  001526  001122          MOV     DRVDBR,$BDADR   ;SET UP DBR ADRS
1518  004726  104013                          ERROR   13              ;DATA ER - MEM CONTAINS WRONG DATA
1519  004730  000406                          BR      4$              ;GO RESTORE VECTOR
1520  004732  005100                  3$:     COM     R0              ;RETURN HERE ON GOOD DATA - NOW COM PATRN
```

```
1521  004734  005101                          COM     R1              ;KEEP TRACK OF COMPLEMENT
1522  004736  001306                          BNE     1$              ;DO COMPLEMENT OF THIS FLOATING ZERO IF 0
1523  004740  006300                          ASL     R0              ;COMPLEMENT WAS DONE - NOW SHIFT ZERO LEFT
1524  004742  005200                          INC     R0              ;KEEP LSB SET
1525  004744  103703                          BCS     1$              ;AGAIN TILL ZERO BIT IN CARRY
1526  004746  004737  007622          4$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1527
1528                                  ;;**********************************************************************
1529                                  ;*TEST 21        TEST 200 "DATI" NPR TRANSFERS (BURST MODE)
1530                                  ;;**********************************************************************
1531  004752  000004                  TST21:  SCOPE
1532  004754  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1533  004762  004537  007602                  JSR     R5,SETVEC       ;GO SET UP INTERRUPT RETURN
1534  004766  005064                          1$                      ;RETURN TO 1$ ON INTR
1535  004770  004737  010026                  JSR     PC,LDBUF        ;GO LOAD BUFFER WITH COMPLEMENTING PATRN
1536  004774  012777  177470  174516          MOV     #-200.,@DRVWCR  ;LOAD WC REG - WILL DO 200 XFERS
1537  005002  012777  014476  174512          MOV     #DBUF,@DRVBAR   ;SET UP CURRENT ADRS
1538  005010  106427  000000                  MTPS,   0               ;ENABLE THE INTR
1539  005014  012777  000101  174502          MOV     #101,@DRVCSR    ;SET IE & GO
1540  005022  052777  000400  174474          BIS     #400,@DRVCSR    ;SET CYCLE
1541  005030  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1542  005036  012737  000700  001124          MOV     #700,$GDDAT     ;LD EXPECTED
1543  005044  017737  174454  001126          MOV     @DRVCSR,$BDDAT  ;READ THE CSR
1544  005052  042777  000100  174444          BIC     #100,@DRVCSR    ;CLR INTR ENABLE
1545  005060  104005                          ERROR   5               ;WCO FAILED TO INTERRUPT (SNGL CYCL ON COULD CAUSE THIS)
1546  005062  000434                          BR      2$              ;GO RESTORE VECTOR
1547  005064  022626                  1$:     CMP     (SP)+,(SP)+     ;INTR RETURNS HERE - FIX STACK SINCE NO RTI
1548  005066  004537  007646                  JSR     R5,CKSTAT       ;GO CHECK STATUS
1549  005072  000700                          700                     ;CSR STATUS EXPECTED
1550  005074  000310                          200.                    ;# OF XFERS
1551  005076  104007                          ERROR   7               ;RETURN HERE IF STATUS ER - EXPECTED CYCLE, READY & IE
1552  005100  000425                          BR      2$              ;GO RESTORE VECTOR
1553  005102  104010                          ERROR   10              ;RETURN HERE IF WC ER - EXPECTED 0
1554  005104  000423                          BR      2$              ;GO RESTORE VECTOR
1555  005106  104011                          ERROR   11              ;RETURN HERE IF BAR ER - SHOULD = DBUF+620
1556  005110  000421                          BR      2$              ;GO RESTORE VECTOR
1557  005112  012737  015314  001120          MOV     #DBUF+616,$GDADR  ;OK - SET UP LAST XFER ADRS WHERE #70707 SHOULD BE
1558  005120  012737  070707  001124          MOV     #70707,$GDDAT   ;LD EXPECTED
1559  005126  017737  174374  001126          MOV     @DRVDBR,$BDDAT  ;DBR SHOULD HAVE LAST DATUM
1560  005134  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1561  005142  001404                          BEQ     2$              ;BR IF SO
1562  005144  013737  001526  001122          MOV     DRVDBR,$BDADR   ;SET UP DBR ADRS
1563  005152  104012                          ERROR   12              ;DATA ER - DBR DID NOT CONTAIN EXPECTED LAST XFER
1564  005154  004737  007622          2$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1565
1566                                  ;;**********************************************************************
1567                                  ;*TEST 22        TEST 200 "DATO" NPR TRANSFERS (BURST MODE)
1568                                  ;;**********************************************************************
1569  005160  000004                  TST22:  SCOPE
1570  005162  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1571  005170  004537  007602                  JSR     R5,SETVEC       ;GO SET UP INTERRUPT RETURN
1572  005174  005274                          1$                      ;RETURN TO 1$ ON INTR
1573  005176  012777  177470  174314          MOV     #-200.,@DRVWCR  ;WORD WC REG - WILL DO 200 XFER'S
1574  005204  012777  014476  174310          MOV     #DBUF,@DRVBAR   ;SET UP CURRENT ADRS
```

```
1575  005212  012777  177377  174306          MOV     #177377,@DRVDBR ;THIS WILL BE WRITTEN TO MEM
1576  005220  106427  000000                  MTPS,   0               ;ENABLE THE INTR
1577  005224  012777  000103  174272          MOV     #103,@DRVCSR    ;SET IE, FNCT 1 & GO
1578  005232  052777  000400  174264          BIS     #400,@DRVCSR    ;SET CYCLE
1579  005240  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1580  005246  012737  001702  001124          MOV     #1702,$GDDAT    ;LD EXPECTED
1581  005254  017737  174244  001126          MOV     @DRVCSR,$BDDAT  ;READ THE CSR
1582  005262  042777  000100  174234          BIC     #100,@DRVCSR    ;CLR INTR ENABLE
1583  005270  104005                          ERROR   5               ;WCO FAILED TO INTERRUPT (SNGL CYCL ON COULD CAUSE THIS.
1584  005272  000416                          BR      2$              ;GO RESTORE VECTOR
1585  005274  022626                  1$:     CMP     (SP)+,(SP)+     ;INTR RETURNS HERE - FIX STACK SINCE NO RTI
1586  005276  004537  007646                  JSR     R5,CKSTA        ;GO CHECK STATUS
1587  005302  001702                          1702                    ;CSR STATUS EXPECTED
1588  005304  000310                          200.                    ;# OF XFERS
1589  005306  104007                          ERROR   7               ;RETURN HERE IF STATUS ER - EXPECTED STAT C.
1590                                                                  ;CYCLE, READY, IE & FNCT 1
1591  005310  000407                          BR      2$              ;GO RESTORE VECTOR
1592  005312  104010                          ERROR   10              ;RETURN HERE IF WC ER - EXPECTED 0
1593  005314  000405                          BR      2$              ;GO RESTORE VECTOR
1594  005316  104011                          ERROR   11              ;RETURN HERE IF BAR ER - SHOULD = DBUF+620
1595  005320  000403                          BR      2$              ;GO RESTORE VECTOR
1596  005322  004737  010150                  JSR     PC,CKDAT        ;RETURN HERE IF OK - NOW GO CHECK DATA
1597  005326  104013                          ERROR   13              ;RETURN HERE IF DATA ER - DBR CONTAINS WRONG DATA
1598  005330  004737  007622          2$:     JSR     PC,RSTVEC       ;RETURN HERE IF DATA CHECK OK - GO RESTORE VECTOR
1599
1600                                  ;;**************************************************************
1601                                  ;*TEST 23       TEST THAT THE CPU IS LOCKED OUT WITH SINGLE CYCLE OFF
1602                                  ;;**************************************************************
1603  005334  000004                  TST23:  SCOPE
1604  005336  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1605  005344  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1606  005352  004537  007602                  JSR     R5,SETVEC       ;GO SET UP INTR RETURN
1607  005356  005466                          3$                      ;RETURN TO 3$ ON INTR
1608  005360  012700  000010                  MOV     #10,R0          ;DO EIGHT 200 WORD XFER'S
1609  005364  005037  001126                  CLR     $BDDAT          ;USR $BDDAT AS A COUNTER
1610  005370  012777  177470  174122  1$:     MOV     #-200.,@DRVWCR  ;DO 200 XFERS (DATI'S)
1611  005376  012777  014476  174116          MOV     #DBUF,@DRVBAR   ;FROM DBUF
1612  005404  106427  000000                  MTPS,   0               ;ALLOW AN INTR
1613  005410  012777  000101  174106          MOV     #101,@DRVCSR    ;SET IE & GO
1614  005416  052777  000400  174100          BIS     #400,@DRVCSR    ;SET CYCLE
1615  005424  000240                          NOP                     ;FREEBEE
1616  005426  000240                          NOP
1617  005430  000240                          NOP
1618  005432  005237  001126          2$:     INC     $BDDAT          ;START COUNTING - SHOULD NEVER GET HERE
1619  005436  001375                          BNE     2$              ;UNTIL 64K
1620  005440  012737  000700  001124          MOV     #700,$GDDAT     ;LD EXPECTED
1621  005446  017737  174052  001126          MOV     @DRVCSR,$BDDAT  ;READ STATUS
1622  005454  042777  000100  174042          BIC     #100,@DRVCSR    ;CLR IE
1623  005462  104005                          ERROR   5               ;NO INTERRUPT ON 200 DATI'S
1624  005464  000407                          BR      4$              ;GO RESTORE VECTOR
1625  005466  022626                  3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
1626  005470  005300                          DEC     R0              ;DONE 8 TIMES?
1627  005472  001336                          BNE     1$              ;BR IF NOT
1628  005474  005737  001126                  TST     $BDDAT          ;SHOULD STILL BE ZERO
```

```
1629  005500  001401                          BEQ     4$              ;BR IF SO
1630  005502  104014                          ERROR   14              ;BURST MD (SINGLE CYCLE=0) FAILS TO LOCK OUT CPU
1631  005504  004737  007622          4$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1632
1633                                  ;;*****************************************************************
1634                                  ;*TEST 24        TEST THAT THE CPU IS NOT LOCKED OUT WITH SINGLE CYCLE ON
1635                                  ;;*****************************************************************
1636  005510  000004                  TST24:  SCOPE
1637  005512  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1638  005520  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1639  005526  004537  007602                  JSR     R5,SETVEC       ;GO SET UP INTR RETURN
1640  005532  005640                          3$                      ;RETURN TO 3$ ON INTR
1641  005534  012700  000010                  MOV     #10,R0          ;DO EIGHT 200 WORD XFER'S
1642  005540  012737  000000  001126          MOV     #0,$BDDAT       ;USE $BDDAT AS A COUNTER
1643  005546  012777  177470  173744  1$:     MOV     #-200.,@DRVWCR  ;DO 200 XFERS (DATI'S)
1644  005554  012777  014476  173740          MOV     #DBUF,@DRVBAR   ;FROM DBUF
1645  005562  106427  000000                  MTPS,   0               ;ALLOW AN INTR
1646  005566  012777  000111  173730          MOV     #111,@DRVCSR    ;SET IE, FNCT3 & GO
1647  005574  052777  000400  173722          BIS     #400,@DRVCSR    ;SET CYCLE
1648  005602  000240                          NOP                     ;FREEBEE
1649  005604  005237  001126          2$:     INC     $BDDAT          ;START COUNTING
1650  005610  001375                          BNE     2$              ;UNTIL 64K - SHOULD INTR BEFORE OVERFLOW
1651  005612  012737  004710  001124          MOV     #4710,$GDDAT    ;LD EXPECTED
1652  005620  017737  173700  001126          MOV     @DRVCSR,$BDDAT  ;READ STATUS
1653  005626  042777  000100  173670          BIC     #100,@DRVCSR    ;CLR IE
1654  005634  104005                          ERROR   5               ;NO INTERRUPT ON 200 DATI'S (WITH SINGLE CYCLE)
1655  005636  000423                          BR      5$              ;GO RESTORE VECTOR
1656  005640  022626                  3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
1657  005642  005300                          DEC     R0              ;DONE 8 TIMES?
1658  005644  001340                          BNE     1$              ;BR IF NOT
1659  005646  022737  000000  001126          CMP     #0,$BDDAT       ;$BDDAT SHOULD HAVE BEEN COUNTED
1660  005654  103401                          BCS     4$              ;BR IF SO
1661  005656  104015                          ERROR   15              ;CPU APPEARED LOCKED OUT WITH SINGLE CYCLE SET
1662  005660  017737  173640  001126  4$:     MOV     @DRVCSR,$BDDAT  ;READ STATUS
1663  005666  012737  004710  001124          MOV     #4710,$GDDAT    ;LD EXPECTED
1664  005674  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1665  005702  001401                          BEQ     5$              ;BR IF SO
1666  005704  104007                          ERROR   7               ;STATUS INCORRECT ON XFER WITH SINGLE CYCLE SET
1667  005706  004737  007622          5$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1668
1669                                  ;;*****************************************************************
1670                                  ;*TEST 25        TEST THAT MAINT MODE CONTROLS FNCT BITS, XFER DIR & SINGLE CYCLE
1671                                  ;;*****************************************************************
1672  005712  000004                  TST25:  SCOPE
1673  005714  012737  000200  001160          MOV     #200,$TIMES     ;;DO 200 ITERATIONS
1674  005722  004537  007602                  JSR     R5,SETVEC       ;GO SET UP INTR RETURN
1675  005726  006044                          2$                      ;RETURN TO 2$ ON INTR
1676  005730  004737  010100                  JSR     PC,LDBUF1       ;GO SET UP DBUF (SPECIAL COM PATTERN)
1677  005734  012737  011702  006052          MOV     #11702,3$       ;3$ CONTAINS EXPECTED STATUS
1678  005742  012737  000001  006054          MOV     #1,4$           ;4$ CONTAINS THE CURRENT XFER NO # (MAX 8)
1679  005750  106427  000000          1$:     MTPS,   0               ;ALLOW INTR
1680  005754  012777  014476  173540          MOV     #DBUF,@DRVBAR   ;SET UP CURRENT ADRS
1681  005762  013777  006054  173530          MOV     4$,@DRVWCR      ;GET XFER #
1682  005770  005477  173524                  NEG     @DRVWCR         ;NEGATE FOR WC
```

```
1683  005774  012777  010101  173522          MOV     #10101,@DRVCSR  ;SET UP MAINT, IE & CO
1684  006002  052777  000400  173514          BIS     #400,@DRVCSR    ;SET CYCLE
1685  006010  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1686  006016  013737  006052  001124          MOV     3$,$GDDAT       ;LD EXPECTED
1687  006024  017737  173474  001126          MOV     @DRVCSR,$BDDAT  ;READ STATUS
1688  006032  042777  000100  173464          BIC     #100,@DRVCSR    ;DISABLE IE
1689  006040  104005                          ERROR   5               ;NO INTR ON XFER (IN MAINT MD)
1690  006042  000440                          BR      6$              ;GO RESTORE VECTOR
1691  006044  022626                  2$:     CMP     (SP)+,(SP)+     ;RETURN HERE ON INTR - FIX STACK SINCE NO RTI
1692  006046  004537  007646                  JSR     R5,CKSTAT       ;GO CHECK STATUS
1693  006052  011702                  3$:     11702                   ;THIS LOCATION WILL CONTAIN EXPECTED STATUS
1694  006054  000001                  4$:     1                       ;THIS LOCATION WILL CONTAIN CURRENT XFER # (MAX 8)
1695  006056  104007                          ERROR   7               ;RETURN HERE IF STATUS ER - WILL EXPECT
1696                                                                  ;MAINT, COUNT INCREASE OF FNCT & STAT BITS,
1697                                                                  ;CYCLE, READY & IE
1698  006060  000431                          BR      6$              ;GO RESTORE VECTOR
1699  006062  104010                          ERROR   10              ;RETURN HERE IF WC ER - SHOULD BE 0
1700  006064  000427                          BR      6$              ;GO RESTORE VECTOR
1701  006066  104011                          ERROR   11              ;RETURN HERE IF BAR ER-
1702  006070  000425                          BR      6$              ;GO RESTORE VECTOR
1703  006072  062737  001002  006052          ADD     #1002,3$        ;RETURN HERE IF OK - ADVANCE EXPECTED STATUS LOC
1704  006100  032737  020000  006052          BIT     #BIT13,3$       ;LOOK FOR OVERFLOW
1705  006106  001403                          BEQ     5$              ;BR IF NOT
1706  006110  012737  010700  006052          MOV     #10700,3$       ;FNCT & STAT BITS SHOULD BE ZERO THIS TIME
1707  006116  005237  006054          5$:     INC     4$              ;ADVANCE CURRENT XFER #
1708  006122  022737  000011  006054          CMP     #11,4$          ;HAVE 10 XFERS BEEN DONE
1709  006130  001307                          BNE     1$              ;BR IF NOT
1710  006132  004537  010216                  JSR     R5,CKDAT1       ;NOW GO CHECK DATA
1711  006136  000010                          10                      ;# OF XFER'S TO CHECK
1712  006140  104013                          ERROR   13              ;RETURN HERE IF DATA ER - (WITH MAINT SET)
1713  006142  000240                          NOP                     ;RESTORE VECTOR NEXT
1714  006144  004737  007622          6$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1715
1716                                          ;;*****************************************************************
1717                                          ;*TEST 26       TEST THAT A DATI FROM A NON-EXISTANT BUS ADRS SETS 'NEX'
1718                                          ;;*****************************************************************
1719  006150  000004                  TST26:  SCOPE
1720  006152  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1721  006160  012700  000002                  MOV     #2,R0           ;R0 WHEN ZERO SAYS CLR 'NEX' WITH RESET
1722  006164  004537  007602          1$:     JSR     R5,SETVEC       ;GO SET UP INTERRUPT RETURN
1723  006170  006262                          2$                      ;RETURN TO 2$ ON TIMEOUT INTR
1724  006172  012777  177777  173320          MOV     #-1,@DRVWCR     ;SET UP FOR ONE XFER'S
1725  006200  012777  160000  173314          MOV     #160000,@DRVBAR ;SET UP CA TO 160000 (RESERVED)
1726  006206  106427  000000                  MTPS,   0               ;ALLOW INTR
1727  006212  012777  000161  173304          MOV     #161,@DRVCSR    ;SET IE, XAD 17,16 & GO
1728  006220  052777  000400  173276          BIS     #400,@DRVCSR    ;SET CYCLE
1729  006226  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS - SHOULD NEVER GET HERE
1730  006234  012737  140760  001124          MOV     #140760,$GDDAT  ;LD EXPECTED
1731  006242  017737  173256  001126          MOV     @DRVCSR,$BDDAT  ;GIVE THEM THE STATUS
1732  006250  042777  000100  173246          BIC     #100,@DRVCSR    ;CLR IE
1733  006256  104016                          ERROR   16              ;NEX FAILED TO CAUSE AN INTERRUPT
1734  006260  000521                          BR      7$              ;GO RESTORE VECTOR
1735  006262  022626                  2$:     CMP     (SP)+,(SP)+     ;SHOULD INTR RETURN HERE - FIX STACK
1736  006264  017737  173234  001126          MOV     @DRVCSR,$BDDAT  ;READ THE CSR
```

```
1737  006272  012737  140760  001124          MOV     #140760,$GDDAT  ;LD EXPECTED
1738  006300  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1739  006306  001405                          BEQ     3$              ;BR IF SO
1740  006310  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1741  006316  104016                          ERROR   16              ;STATUS ER - EXPECTED
1742                                                                  ;ER, NEX, CYCLE, READY, IE, XAD17 & XAD16
1743  006320  000501                          BR      7$              ;GO RESTORE VECTOR
1744  006322  017737  173172  001126  3$:     MOV     @DRVWCR,$BDDAT  ;READ WORD COUNT
1745  006330  012737  177777  001124          MOV     #-1,$GDDAT      ;SHOULD STILL HAVE -1
1746  006336  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1747  006344  001405                          BEQ     4$              ;BR IF SO
1748  006346  013737  001520  001122          MOV     DRVWCR,$BDADR   ;SET UP WCR ADRS
1749  006354  104010                          ERROR   10              ;WC INCREMENTED ON A TIMEOUT ER
1750  006356  000462                          BR      7$              ;GO RESTORE VECTOR
1751  006360  017737  173136  001126  4$:     MOV     @DRVBAR,$BDDAT  ;READ BUFFER ADRS
1752  006366  012737  160000  001124          MOV     #160000,$GDDAT  ;SHOULD NOT HAVE INCREMENTED
1753  006374  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1754  006402  001405                          BEQ     5$              ;BR IF SO
1755  006404  013737  001522  001122          MOV     DRVBAR,$BDADR   ;SET UP BAR ADRS
1756  006412  104011                          ERROR   11              ;BAR INCREMENTED ON A TIMEOUT ER
1757  006414  000443                          BR      7$              ;GO RESTORE VECTOR
1758  006416  005300                  5$:     DEC     R0              ;KEEP TRACK ON HOW TO CLR
1759  006420  001422                          BEQ     6$              ;BR IF CLR BY RESET
1760  006422  042777  040000  173074          BIC     #40000,@DRVCSR  ;WRITE NEX TO ZERO
1761  006430  017737  173070  001126          MOV     @DRVCSR,$BDDAT  ;READ CSR
1762  006436  012737  000760  001124          MOV     #760,$GDDAT     ;LD EXPECTED
1763  006444  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1764  006452  001644                          BEQ     1$              ;BR IF SO + REPEAT TEST FOR RESET TEST
1765  006454  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1766  006462  104016                          ERROR   16              ;'NEX' FAILED TO WRITE TO ZERO
1767  006464  000417                          BR      7$              ;GO RESTORE VECTOR
1768  006466  000005                  6$:     RESET                   ;ISSUE BUS RESET
1769  006470  017737  173030  001126          MOV     @DRVCSR,$BDDAT  ;READ THE CSR
1770  006476  012737  000200  001124          MOV     #200,$GDDAT     ;EXPECT ONLY READY
1771  006504  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1772  006512  001404                          BEQ     7$              ;BR IF SO
1773  006514  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1774  006522  104016                          ERROR   16              ;RESET FAILED TO CLR 'NEX'
1775  006524  004737  007622          7$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1776                                  ;;**************************************************************
1777                                  ;*TEST 27        TEST 200 NPR TRANSFERS IN MAINT MODE
1778                                  ;;**************************************************************
1779  006530  000004                  TST27:  SCOPE
1780  006532  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1781  006540  004537  007602                  JSR     R5,SETVEC       ;GO SET UP INTR RETURN
1782  006544  006656                          2$                      ;RETURN TO 2$ ON INTR
1783  006546  004737  010100                  JSR     PC,LDBUF1       ;GO SET UP DBUF (SPECIAL COM PATTERN)
1784  006552  012777  014476  172742          MOV     #DBUF,@DRVBAR   ;SET UP CURRENT ADRS
1785  006560  012777  177470  172732          MOV     #-200.,@DRVWCR  ;SET UP FOR 200 XFER'S
1786  006566  106427  000000                  MTPS,   0               ;ALLOW INTR
1787  006572  012777  010101  172724          MOV     #10101,@DRVCSR  ;SET MAINT, IE & GO
1788  006600  052777  000400  172716          BIS     #400,@DRVCSR    ;SET CYCLE
1789  006606  012737  000000  001126          MOV     #0,$BDDAT       ;SET UP A COUNTER
1790  006614  005237  001126          1$:     INC     $BDDAT          ;COUNT AWAY
```

```
1791  006620  001375                          BNE     1$              ;WAIT TILL DONE - SHOULD INTR BEFORE OVFLO
1792  006622  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1793  006630  012737  010700  001124          MOV     #10700,$GDDAT   ;LD EXPECTED
1794  006636  017737  172662  001126          MOV     @DRVCSR,$BDDAT  ;READ STATUS
1795  006644  042777  000100  172652          BIC     #100,@DRVCSR    ;DISABLE IE
1796  006652  104005                          ERROR   5               ;NO INTR AFTER 200 MAINT MODE XFER'S
1797  006654  000420                          BR      3$              ;GO RESTORE VECTOR
1798  006656  022626                  2$:     CMP     (SP)+,(SP)+     ;RETURN HERE ON INTR - FIX STACK SINCE NO RTI
1799  006660  004537  007646                  JSR     R5,CKSTAT       ;GO CHECK STATUS
1800  006664  010700                          10700                   ;EXPECTED STATUS
1801  006666  000310                          200.                    ;# OF XFERS
1802  006670  104007                          ERROR   7               ;RETURN HERE IF STATUS ER - EXPECTED MAINT,
1803                                                                  ;CYCLE, READY & IE
1804  006672  000411                          BR      3$              ;GO RESTORE VECTOR
1805  006674  104010                          ERROR   10              ;RETURN HERE IF WC ER - SHOULD BE 0
1806  006676  000407                          BR      3$              ;GO RESTORE VECTOR
1807  006700  104011                          ERROR   11              ;RETURN HERE IF BAR ER - SHOULD = DBUF+620
1808  006702  000405                          BR      3$              ;GO RESTORE VECTOR
1809  006704  004537  010216                  JSR     R5,CKDAT1       ;RETURN HERE IF OK - NOW GO CHECK DATA
1810  006710  000310                          200.                    ;# OF XFER'S TO CHECK
1811  006712  104013                          ERROR   13              ;RETURN HERE IF DATA ER - (WITH MAINT SET)
1812  006714  000240                          NOP                     ;RESTORE VECTOR NEXT
1813  006716  004737  007622          3$:     JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1814                                  ;;**********************************************************
1815                                  ;*TEST 30        TEST A 200 WORD MAINT MODE XFER TO EACH ADDITIONAL AVAILABLE 4K
1816                                  ;;**********************************************************
1817  006722  000004                  TST30:  SCOPE
1818  006724  012737  000005  001160          MOV     #5,$TIMES       ;;DO 5 ITERATIONS
1819  006732  004537  007602                  JSR     R5,SETVEC       ;GO SET UP INTR RETURN
1820  006736  007072                          3$                      ;RETURN TO 3$ ON INTR
1821  006740  005037  001542                  CLR     DBUFP           ;GET LOWEST BUFFER ADRS
1822  006744  062737  020000  001542  1$:     ADD     #20000,DBUFP    ;POINT TO NEXT 4K
1823  006752  023737  001540  001542          CMP     CORSZ,DBUFP     ;IS THE 4K THERE?
1824  006760  001465                          BEQ     4$              ;BR IF NOT
1825  006762  004737  010100                  JSR     PC,LDBUF1       ;GO SET UP SPECIAL COMPEMENT PATTERN
1826  006766  013777  001542  172526          MOV     DBUFP,@DRVBAR   ;SET UP BUFFER ADRS
1827  006774  012777  177470  172516          MOV     #-200.,@DRVWCR  ;SET UP FOR 200 XFER'S
1828  007002  106427  000000                  MTPS,   0               ;ALLOW INTR
1829  007006  012777  010101  172510          MOV     #10101,@DRVCSR  ;SET MAINT,IE & GO
1830  007014  052777  000400  172502          BIS     #400,@DRVCSR    ;SET CYCLE
1831  007022  012737  000000  001126          MOV     #0,$BDDAT       ;SET UP A COUNTER
1832  007030  005237  001126          2$:     INC     $BDDAT          ;COUNT AWAY
1833  007034  001375                          BNE     2$              ;SHOULD ALWAYS INTR FROM THIS LOOP
1834  007036  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRA
1835  007044  012737  010700  001124          MOV     #10700,$GDDAT   ;LD EXPECTED
1836  007052  017737  172446  001126          MOV     @DRVCSR,$BDDAT  ;READ CSR
1837  007060  042777  000100  172436          BIC     #100,@DRVCSR    ;DISABLE IE
1838  007066  104005                          ERROR   5               ;NO INTR AFTER 200 MAINT MODE XFER'S
1839  007070  000421                          BR      4$              ;GO RESTORE VECTOR
1840  007072  022626                  3$:     CMP     (SP)+,(SP)+     ;RETURN HERE ON INTR - FIX STACK SINCE NO RTI
1841  007074  004537  007646                  JSR     R5,CKSTAT       ;GO CHECK STATUS
1842  007100  010700                          10700                   ;EXPECTED STATUS
1843  007102  000310                          200.                    ;# OF XFER'S
1844  007104  104007                          ERROR   7               ;RETURN HERE IF STATUS ER - EXPECTED MAINT,
```

```
1845
1846  007106  000412                          BR      4$              ;CYCLE, READY & IE
1847  007110  104010                          ERROR   10              ;GO RESTORE VECTOR
1848  007112  000410                          BR      4$              ;RETURN HERE IF WC ER - SHOULD = 0
1849  007114  104011                          ERROR   11              ;GO RESTORE VECTOR
1850  007116  000406                          BR      4$              ;RETURN HERE IF BAR ER - SHOULD = DBUFF+620
1851  007120  004537  010216                  JSR     R5,CKDAT1       ;GO RESTORE VECTOR
1852  007124  000310                          200.                    ;RETURN HERE IF OK - NOW GO CK DATA
1853  007126  104013                          ERROR   13              ;# OF XFER'S TO CK
1854  007130  000401                          BR      4$              ;RETURN HERE IF DATA ER
1855  007132  000704                          BR      1$              ;GO RESTORE VECTOR
1856  007134  012737  014476  001542  4$:     MOV     #DBUF,DBUFP     ;TRY NEXT BANK
1857  007142  004737  007622                  JSR     PC,RSTVEC       ;RESTORE BUFFER ADRS TO LOWEST 4K
1858                                          ;;**************************************************************
1859                                          ;*TEST 31        TEST THE ADDRESS (I/O) ABILITY TO THE TTY PRINTER CSR
1860                                          ;;**************************************************************
1861  007146  000004                  TST31:  SCOPE
1862  007150  004537  007602                  JSR     R5,SETVEC       ;GO SET UP INTR RETURN
1863  007154  007260                          1$                      ;RETURN TO 1$ ON INTR
1864  007156  012777  177777  172334          MOV     #-1,@DRVWCR     ;SET UP WC - 1 XFER
1865  007164  013737  001150  001542          MOV     $TPS,DBUFP      ;SET UP BUFFER ADRS TO PRINTER CSR ADRS
1866  007172  013777  001150  172322          MOV     $TPS,@DRVBAR    ;SET UP BUFFER ADRS - FROM PRINTER CSR
1867  007200  106427  000000                  MTPS,   0               ;ALLOW INTR
1868  007204  005077  172316                  CLR     @DRVDBR         ;ZERO THE DBR
1869  007210  012777  000161  172306          MOV     #161,@DRVCSR    ;SET IE, XAD17 & XAD16, & GO
1870  007216  052777  000400  172300          BIS     #400,@DRVCSR    ;SET CYCLE
1871  007224  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
1872  007232  012737  000760  001124          MOV     #760,$GDDAT     ;LD EXPECTED STATUS
1873  007240  017737  172260  001126          MOV     @DRVCSR,$BDDAT  ;READ THE CSR
1874  007246  042777  000100  172250          BIC     #100,@DRVCSR    ;DISABLE IE
1875  007254  104005                          ERROR   5               ;NO INTR ON 1 WD XFER FROM XCSR
1876  007256  000434                          BR      2$              ;GO RESTORE VECTOR
1877  007260  022626                  1$:     CMP     (SP)+,(SP)+     ;INTR RETURNS HERE - FIX STK SINCE NO RTI
1878  007262  004537  007646                  JSR     R5,CKSTAT       ;GO CK STATUS
1879  007266  000760                          760                     ;CSR EXPECTED STATUS
1880  007270  000001                          1                       ;# OF XFER'S
1881  007272  104007                          ERROR   7               ;RETURN HERE IF STATUS ER - EXPECTED CYCLE,
1882                                                                  ;XAD17 & XAD16, RDY & IE
1883  007274  000425                          BR      2$              ;GO RESTORE VECTOR
1884  007276  104010                          ERROR   10              ;RETURN HERE IF WC ER - EXPECTED 0
1885  007300  000423                          BR      2$              ;GO RESTORE VECTOR
1886  007302  104011                          ERROR   11              ;RETURN HERE IF BAR ER - SHOULD = XCSR+2
1887  007304  000421                          BR      2$              ;GO RESTORE VECTOR
1888  007306  017737  171636  001124          MOV     @$TPS,$GDDAT    ;RETURN HERE IF OK - GET XCSR CONTENTS
1889  007314  017737  172206  001126          MOV     @DRVDBR,$BDDAT  ;READ DBR
1890  007322  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1891  007330  001407                          BEQ     2$              ;BR IF SO
1892  007332  013737  001520  001122          MOV     DRVDBR,$BDADR   ;SET UP DBR ADRS
1893  007340  013737  001150  001120          MOV     $TPS,$GDADR     ;GET ADRS OF DATA (XCSR)
1894  007346  104020                          ERROR   20              ;DATA ER FROM TTY PRINTER CSR
1895  007350  012737  014476  001542  2$:     MOV     #DBUF,DBUFP     ;RESTORE BUFFER ADRS TO LOWEST 4K
1896  007356  004737  007622                  JSR     PC,RSTVEC       ;GO RESTORE VECTOR
1897
1898                                          ;;**************************************************************
```

```
                                        ;DON'T REPORT 'END OF PASS' UNTIL ALL SELECTED DRV11'S HAVE BEEN TESTED
                                        ;;**************************************************************
007362  000004                        NXDEV:  SCOPE
007364  000241                        NXDEV1: CLC                     ;CLR CARRY
007366  006037  001536                        ROR     DMAP            ;LOOK FOR NEXT
007372  001432                                BEQ     $EOP            ;BR IF ALL TESTED
007374  062737  000010  001520                ADD     #10,DRVWCR      ;OFFSET BASE BUS ADRS TO NEXT DRV11B
007402  062737  000010  001522                ADD     #10,DRVBAR      ;
007410  062737  000010  001524                ADD     #10,DRVCSR      ;
007416  062737  000010  001526                ADD     #10,DRVDBR      ;
007424  062737  000004  001530                ADD     #4,DRVCT0       ;OFFSET VECTOR ADRS TO NEXT
007432  062737  000004  001532                ADD     #4,DRVCT2       ;
007440  005237  001206                        INC     $UNIT           ;COUNT DEVICE
007444  032737  000001  001536                BIT     #1,DMAP         ;IS IT SELECTED?
007452  001744                                BEQ     .NXDEV1         ;BR IF NOT
007454  000137  002330                        JMP     RESTRT          ;TEST NEXT
                                        .SBTTL  END OF PASS ROUTINE
                                        ;;**************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO START1
                                        $EOP:
007460                                          NOP
007460  000240                                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
007462  005037  001102                          CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
007466  005037  001160                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
007472  005237  001202                          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
007476  042737  100000  001202                DEC     (PC)+           ;;LOOP?
007504  005327                          $EOPCT: .WORD   1
007506  000001                                  BGT     $DOAGN          ;;YES
007510  003022                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
007512  012737                          $ENDCT: .WORD   1
007514  000001                                  $EOPCT
007516  007506                                  TYPE    ,$ENDMG         ;;TYPE "END PASS #"
007520  104400  007565                          MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
007524  013746  001202                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
007530  104404                                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
007532  104400  007562                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
007536  013700  000042                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
007542  001405                                  RESET                   ;;CLEAR THE WORLD
007544  000005                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
007546  004710                                  NOP                     ;;SAVE ROOM
007550  000240                                  NOP                     ;;FOR
007552  000240                                  NOP                     ;;ACT11
007554  000240                          $DOAGN:
007556                                          JMP     @(PC)+          ;;RETURN
007556  000137                          $RTNAD: .WORD   START1
007560  002006                          $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
007562     377     377     000          $ENDMG: .ASCIZ  <15><12>/END PASS #/
007565     015  042412  042116
007572  050040  051501  020123
007600  000043
```

```
                                        .SBTTL  PROGRAM SUBROUTINES


                                        ;;*****************************************************************
                                        ;THIS ROUTINE SETS THE PRIORITY LEVEL FOR NO INTERRUPT -
                                        ;SETS UP THE DRV11B INTERRUPT TO RETURN ON INTERRUPT
                                        ;TO THE ADDRESS INDICATED ((R5)) BY THE CALL +2
                                        ;;*****************************************************************
007602  106427  000200          SETVEC: MTPS,   200             ;SET UP FOR NO INTERRUPT
007606  012577  171716                  MOV     (R5)+,@DRVCTO   ;SET UP INTR RETURN ADRS
007612  012777  000200  171712          MOV     #200,@DRVCT2    ;KEEP PRIORITY LEVEL AT TOP ON INTR
007620  000205                          RTS     R5              ;EXIT

                                        ;;*****************************************************************
                                        ;THIS ROUTINE CLEARS THE DRV11B CSR - RESTORES THE DRV11B
                                        ;INTERRUPT VECTOR TO A HALT - RAISES PRIORITY LEVEL
                                        ;;*****************************************************************
007622  005077  171676          RSTVEC: CLR     @DRVCSR         ;CLR STATUS & CONTROL
007626  013777  001532  171674          MOV     DRVCT2,@DRVCTO  ;POINT VECTOR TO HALT
007634  005077  171672                  CLR     @DRVCT2         ;SET UP HALT
007640  104427  000200                  MTPS,   200             ;RAISE PRIORITY LEVEL
007644  000207                          RTS     PC              ;EXIT

                                        ;;*****************************************************************
                                        ;THIS ROUTINE CHECKS ON ALL DATA TRANSFERS FOR: CORRECT STATUS,
                                        ;CORRECT WORD COUNT & CORRECT BUFFER ADDRESS - THE EXPECTED DATA IS
                                        ;SUPPLIED IN THE CALL +2 & +4 - THE RETURN IS TO +22 IF NO ERRORS
                                        ;DETECTED - IF AN ERROR IS DETECTED THE RETURN IS TO THE APPROPRIATE ERROR EMT
                                        ;;*****************************************************************
007646  017737  171652  001126  CKSTAT: MOV     @DRVCSR,$BDDAT  ;READ THE STATUS
007654  042777  000100  171642          BIC     #100,@DRVCSR    ;DISABLE THE IE BIT
007662  012537  001124                  MOV     (R5)+,$GDDAT    ;SET UP EXPECTED STATUS
007666  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
007674  001406                          BEQ     1$              ;BR IF SO
007676  013737  001524  001122          MOV     DRVCSR,$BDADR   ;SET UP CSR ADRS
007704  062705  000002                  ADD     #2,R5           ;POINT TO THE CSR ER
007710  000205                          RTS     R5              ;EXIT HERE ON STATUS ERROR
007712  017737  171602  001126  1$:     MOV     @DRVWCR,$BDDAT  ;GET WC
007720  001410                          BEQ     2$              ;BR IF ZERO
007722  013737  001520  001122          MOV     DRVWCR,$BDADR   ;SET UP WCR ADRS
007730  005037  001124                  CLR     $GDDAT          ;EXPECTED 0
007734  062705  000006                  ADD     #6,R5           ;POINT TO THE WCR ER
007740  000205                          RTS     R5              ;EXIT HERE ON WCR ER
007742  011537  001124          2$:     MOV     (R5),$GDDAT     ;GET XFER #
007746  006337  001124                  ASL     $GDDAT          ;CONVERT TO WORD
007752  063737  001542  001124          ADD     DBUFP,$GDDAT    ;POINT TO LAST XFER +2
007760  017737  171536  001126          MOV     @DRVBAR,$BDDAT  ;GET BA
007766  042737  000001  001126          BIC     #BIT00,$BDDAT   ;DON'T WANT BIT00
007774  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
010002  001406                          BEQ     3$              ;BR IF SO
010004  013737  001522  001122          MOV     DRVBAR,$BDADR   ;SET UP BAR ADRS
010012  062705  000012                  ADD     #12,R5          ;POINT TO BAR ER
010016  000205                          RTS     R5              ;EXIT HERE ON BAR ER
010020  062705  000016          3$:     ADD     #16,R5          ;ALL OK - POINT TO GOOD EXIT
```

```
2007  010024  000205                        RTS     R5              ;EXIT HERE IF NO ERRORS
2008
2009                                  ;;*****************************************************************
2010                                  ;THIS ROUTINE LOADS 'DBUF' WITH A FLOATING ZERO/ONE PATTERN
2011                                  ;FOR 199 LOCATIONS - THE LAST LOCATION IS LOADED WITH THE
2012                                  ;#70707 WHICH SHOULD BE THE DATA WORD AVAILABLE IN THE DBR
2013                                  ;AT THE COMPLETION OF A 200 WORD TRANSFER
2014                                  ;;*****************************************************************
2015  010026  012703  014476          LDBUF:  MOV     #DBUF,R3        ;GET BUFFER ADRS
2016  010032  012704  177776          1$:     MOV     #177776,R4      ;SET UP FLOATING ZERO PATRN
2017  010036  010423                  2$:     MOV     R4,(R3)+        ;LOAD IT (FLOATING 0)
2018  010040  005104                          COM     R4              ;MAKE INTO FLOATING 1
2019  010042  022703  015314                  CMP     #DBUF+616,R3    ;AT END OF BUFFER?
2020  010046  001003                          BNE     4$              ;BR IF NOT
2021  010050  012713  070707          3$:     MOV     #70707,(R3)     ;LOAD LAST DATVAR (SPECIAL)
2022  010054  000207                          RTS     PC              ;GET OUT
2023  010056  010423                  4$:     MOV     R4,(R3)+        ;LOAD IT (FLOATING 1)
2024  010060  022703  015314                  CMP     #DBUF+616,R3    ;AT END OF BUFFER?
2025  010064  001771                          BEQ     3$              ;BR IF SO
2026  010066  005104                          COM     R4              ;BACK TO FLOATING ZERO
2027  010070  006304                          ASL     R4              ;SHIFT LEFT
2028  010072  005204                          INC     R4              ;KEEP LSB SET
2029  010074  103356                          BCC     1$              ;GO RESET FLOATING PATRN
2030  010076  000757                          BR      2$              ;GO LOAD NEXT PATRN
2031
2032                                  ;;*****************************************************************
2033                                  ;THIS ROUTINE LOADS 'DBUF' WITH A UNIQUE FLOATING ZERO/ONE PATTERN
2034                                  ;(177776,0,1,0,177775,0,2,0,177773,0,4,0,177767,0,10,0 ETC.)
2035                                  ;IT IS USED WITH MAINT BIT SET (DATI/DATO SEQUENCE) - 200 LOCS
2036                                  ;ARE LOADED WITH THIS PATTERN
2037                                  ;;*****************************************************************
2038  010100  013703  001542          LDBUF1: MOV     DBUFP,R3        ;GET BUFFER ADRS
2039  010104  010305                          MOV     R3,R5           ;SAVE IN R5
2040  010106  062705  000620                  ADD     #620,R5         ;POINT TO END OF BUFFER
2041  010112  012704  177776          1$:     MOV     #177776,R4      ;SET UP FLOATING ZERO PATRN
2042  010116  010423                  2$:     MOV     R4,(R3)+        ;LOAD IT (FLOATING 0)
2043  010120  005023                          CLR     (R3)+           ;ZERO NEXT
2044  010122  005104                          COM     R4              ;SET UP FLOATING 1
2045  010124  010423                          MOV     R4,(R3)+        ;LOAD IT
2046  010126  005023                          CLR     (R3)+           ;ZERO NEXT
2047  010130  020503                          CMP     R5,R3           ;200 LOCS DONE?
2048  010132  001001                          BNE     3$              ;BR IF NOT
2049  010134  000207                          RTS     PC              ;GET OUT
2050  010136  005104                  3$:     COM     R4              ;BACK TO FLOATING ZERO
2051  010140  006304                          ASL     R4              ;SHIFT LEFT
2052  010142  005204                          INC     R4              ;KEEP LSB SET
2053  010144  103362                          BCC     1$              ;GO RESET FLOATING PATRN
2054  010146  000763                          BR      2$              ;GO FLOAT NEXT PATRN
2055
2056                                  ;;*****************************************************************
2057                                  ;THIS ROUTINE CHECKS 200 LOCATIONS IN "DBUF" FOR GOOD TRANSFERED
2058                                  ;DATA (#177377) ON 'DATO' TRANSFERS - IF AN ERROR IS DETECTED
2059                                  ;THE RETURN IS TO CALL +2 - IF NO ERROR THE RETURN IS TO CALL +4
2060                                  ;;*****************************************************************
```

```
 2061  010150  012701  014476          CKDAT:  MOV     #DBUF,R1        ;GET BUFFER ADRS
 2062  010154  022721  177377          1$:     CMP     #177377,(R1)+   ;DATA OK?
 2063  010160  001410                          BEQ     2$              ;BR IF SO
 2064  010162  013737  001526  001122          MOV     DRVDBR,$BDADR   ;SET UP DBR ADRS
 2065  010170  014137  001126                  MOV     -(R1),$BDDAT    ;GET ACTUAL DATA XFERED
 2066  010174  010137  001120                  MOV     R1,$GDADR       ;GET MEMORY ADRS
 2067  010200  000207                          RTS     PC              ;RETURN TO ERROR
 2068  010202  022701  015316          2$:     CMP     #DBUF+620,R1    ;AT END OF 'DBUF'?
 2069  010206  001362                          BNE     1$              ;BR IF MORE
 2070  010210  062716  000002                  ADD     #2,(SP)         ;ADJUST STACK FOR GOOD RETURN
 2071  010214  000207                          RTS     PC              ;GET OUT
 2072
 2073                                  ;;*****************************************************************
 2074                                  ;THIS ROUTINE CHECK 200 LOCATIONS IN 'DBUF' FOR GOOD TRANSFERED
 2075                                  ;DATA (177776,177776,1,1,177775,177775,2,2,177773,177773,ETC.)
 2076                                  ;ON MAINT MODE TRANSFERS - THE NUMBER OF CHECKS REQUIRED IS INDICATED
 2077                                  ;BY THE CALL +2 - IF AN ERROR IS DETECTED THE RETURN IS TO CALL +4 -
 2078                                  ;IF NO ERROR THE RETURN IS TO CALL +10
 2079                                  ;;*****************************************************************
 2080  010216  012500                  CKDAT1: MOV     (R5)+,R0        ;GET # OF CHECKS
 2081  010220  013702  001542                  MOV     DBUFP,R2        ;GET BUFFER ADRS
 2082  010224  012701  177776          1$:     MOV     #177776,R1      ;SET UP FLOATING ZERO PATRN
 2083  010230  005003                          CLR     R3              ;R3 SAYS WHEN TO SHIFT PATRN
 2084  010232  020122                  2$:     CMP     R1,(R2)+        ;DATA OK?
 2085  010234  001010                          BNE     3$              ;BR IF NOT
 2086  010236  020122                          CMP     R1,(R2)+        ;DATA WRITTEN OK?
 2087  010240  001006                          BNE     3$              ;BR IF NOT
 2088  010242  162700  000002                  SUB     #2,R0           ;ACCOUNT FOR TWO ADRS'S
 2089  010246  003015                          BGT     4$              ;BR IF MORE
 2090  010250  062705  000004                  ADD     #4,R5           ;ADJUST FOR GOOD RETURN
 2091  010254  000205                          RTS     R5              ;EXIT
 2092  010256  014237  001126          3$:     MOV     -(R2),$BDDAT    ;GET BAD DATA
 2093  010262  010237  001120                  MOV     R2,$GDADR       ;GET MEM ADRS
 2094  010266  010137  001124                  MOV     R1,$GDDAT       ;LD EXPECTED DATA
 2095  010272  013737  001526  001122          MOV     DRVDBR,$BDADR   ;SET UP DBR ADRS
 2096  010300  000205                          RTS     R5              ;RETURN TO ERROR
 2097  010302  005101                  4$:     COM     R1              ;NOW EXPECT COMPLEMENT
 2098  010304  005103                          COM     R3              ;TIME TO SHIFT?
 2099  010306  001351                          BNE     2$              ;BR IF NOT
 2100  010310  006301                          ASL     R1              ;SHIFT LEFT
 2101  010312  005201                          INC     R1              ;KEEP LSB SET
 2102  010314  103343                          BCC     1$              ;GO RESET FLOATING PATRN
 2103  010316  000745                          BR      2$              ;DO NEXT
```

```
                                        .SBTTL  SYSMAC ROUTINES

                                .SBTTL  TYPE ROUTINE

                                ;;*****************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*

010320  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
010324  100002                          BPL     1$              ;;BR IF YES
010326  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
010330  000430                          BR      3$              ;;LEAVE
010332  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
010334  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
010340  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
010346  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
010350  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
010356  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
010360  010037  010370                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
010364  004737  010610                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
010370  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
010372  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
010400  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
010402  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
010404  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
010406  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
010410  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
010412  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
010416  000002                          RTI                     ;;RETURN
010420  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
010424  001430                          BEQ     8$
010426  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
010432  001006                          BNE     5$
010434  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
010436  104400                          TYPE                    ;;TYPE A CR AND LF
010440  001171                          $CRLF
010442  105037  010576                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
010446  000755                          BR      2$              ;;GET NEXT CHARACTER
010450  004737  010532          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
010454  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
010460  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
010462  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
```

```
2158  010466  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2159  010472  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2160  010474  004737  010532                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
2161  010500  105337  010576                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2162  010504  000770                          BR      7$              ;;LOOP
2163
2164                                  ;HORIZONTAL TAB PROCESSOR
2165
2166  010506  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2167  010512  004737  010532          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2168  010516  132737  000007  010576          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2169  010524  001372                          BNE     9$              ;;TAB STOP
2170  010526  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
2171  010530  000724                          BR      2$              ;;GET NEXT CHARACTER
2172  010532  105777  170412          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2173  010536  100375                          BPL     $TYPEC
2174  010540  116677  000002  170404          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2175  010546  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2176  010554  001003                          BNE     1$              ;;BRANCH IF NO
2177  010556  105037  010576                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2178  010562  000406                          BR      $TYPEX          ;;EXIT
2179  010564  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2180  010572  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
2181  010574  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
2182  010576  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2183  010600  000207          $TYPEX: RTS     PC
2184
2185                                  .SBTTL  APT COMMUNICATIONS ROUTINE
2186
2187                                  ;;**************************************************************
2188  010602  112737  000001  011046  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
2189  010610  112737  000001  011044  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
2190  010616  000403                          BR      $ATYC
2191  010620  112737  000001  011046  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
2192  010626                          $ATYC:
2193  010626  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2194  010630  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2195  010632  105737  011044                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
2196  010636  001450                          BEQ     5$              ;;IF NOT:  BR
2197  010640  122737  000001  001214          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
2198  010646  001031                          BNE     3$              ;;IF NOT:  BR
2199  010650  132737  000100  001215          BITB    #APTSPOOL.$ENVM ;;SHOULD SPOOL MESSAGES?
2200  010656  001425                          BEQ     3$              ;;IF NOT:  BR
2201  010660  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
2202  010664  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2203  010672  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
2204  010676  001375                          BNE     1$              ;;IF NOT:  WAIT
2205  010700  010037  001210                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
2206  010704  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
2207  010706  001376                          BNE     2$
2208  010710  163700  001210                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
2209  010714  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
2210  010716  010037  001212                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
2211  010722  012737  000004  001174          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
```

```
2212  010730  000413                          BR      5$
2213  010732  017637  000004  010756  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
2214  010740  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
2215  010746  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
2216  010752  004737  010320                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
2217  010756  000000                  4$:     .WORD   0
2218  010760                          5$:
2219  010760  105737  011046          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
2220  010764  001416                          BEQ     12$             ;;IF NOT:  BR
2221  010766  005737  001214                  TST     $ENV            ;;RUNNING UNDER APT?
2222  010772  001413                          BEQ     12$             ;;IF NOT:  BR
2223  010774  005737  001174          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
2224  011000  001375                          BNE     11$             ;;IF NOT:  WAIT
2225  011002  017637  000004  001176          MOV     @4(SP),$FATAL   ;;GET ERROR #
2226  011010  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2227  011016  005237  001174                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
2228  011022  105037  011046          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
2229  011026  105037  011045                  CLRB    $LFLG           ;;CLEAR LOG FLAG
2230  011032  105037  011044                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
2231  011036  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2232  011040  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2233  011042  000207                          RTS     PC              ;;RETURN
2234  011044     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
2235  011045     000          $LFLG:  .BYTE   0               ;;LOG FLAG
2236  011046     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
2237                                  .EVEN
2238          011050          APTSIZE=200
2239          000001          APTENV=001
2240          000100          APTSPOOL=100
2241          000040          APTCSUP=040
2242                          .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
2243
2244                          ;;**************************************************************
2245                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2246                          ;*OCTAL (ASCII) NUMBER AND TYPE IT.
2247                          ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2248                          ;*CALL:
2249                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2250                          ;*      TYPOS                   ;;CALL FOR TYPEOUT
2251                          ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2252                          ;*      .BYTE   M               ;;M=1 OR 0
2253                          ;*                                      ;;1=TYPE LEADING ZEROS
2254                          ;*                                      ;;0=SUPPRESS LEADING ZEROS
2255                          ;*
2256                          ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2257                          ;*$TYPOS OR $TYPOC
2258                          ;*CALL:
2259                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2260                          ;*      TYPON                   ;;CALL FOR TYPEOUT
2261                          ;*
2262                          ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
2263                          ;*CALL:
2264                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2265                          ;*      TYPOC                   ;;CALL FOR TYPEOUT
```

```
2266
2267  011050  017646  000000          $TYPOS: MOV     0(SP),-(SP)     ;;PICKUP THE MODE
2268  011054  116637  000001  011273          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
2269  011062  112637  011275                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
2270  011066  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
2271  011072  000406                          BR      $TYPON
2272  011074  112737  000001  011273  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
2273  011102  112737  000006  011275          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
2274  011110  112737  000005  011272  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
2275  011116  010346                          MOV     R3,-(SP)        ;;SAVE R3
2276  011120  010446                          MOV     R4,-(SP)        ;;SAVE R4
2277  011122  010546                          MOV     R5,-(SP)        ;;SAVE R5
2278  011124  113704  011275                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
2279  011130  005404                          NEG     R4
2280  011132  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
2281  011136  110437  011274                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
2282  011142  113704  011273                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2283  011146  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
2284  011152  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
2285  011154  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2286  011156  000404                          BR      3$              ;;GO DO MSB
2287  011160  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
2288  011162  006105                          ROL     R5
2289  011164  006105                          ROL     R5
2290  011166  010503                          MOV     R5,R3
2291  011170  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2292  011172  105337  011274                  DECB    $OMODE          ;;TYPE THIS DIGIT?
2293  011176  100016                          BPL     7$              ;;BR IF NO
2294  011200  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
2295  011204  001002                          BNE     4$              ;;TEST FOR 0
2296  011206  005704                          TST     R4              ;;SUPPRESS THIS 0?
2297  011210  001403                          BEQ     5$              ;;BR IF YES
2298  011212  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2299  011214  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2300  011220  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2301  011224  110337  011270                  MOVB    R3,8$           ;;SAVE FOR TYPING
2302  011230  104400  011270                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
2303  011234  105337  011272          7$:     DECB    $OCNT           ;;COUNT BY 1
2304  011240  003347                          BGT     2$              ;;BR IF MORE TO DO
2305  011242  002402                          BLT     6$              ;;BR IF DONE
2306  011244  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2307  011246  000744                          BR      2$              ;;GO DO THE LAST DIGIT
2308  011250  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
2309  011252  012604                          MOV     (SP)+,R4        ;;RESTORE R4
2310  011254  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2311  011256  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2312  011264  012616                          MOV     (SP)+,(SP)
2313  011266  000002                          RTI                     ;;RETURN
2314  011270     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2315  011271     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2316  011272     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2317  011273     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2318  011274  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2319                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
```

```
2320
2321                                    ;;**********************************************************************
2322                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
2323                                    ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
2324                                    ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
2325                                    ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
2326                                    ;*REPLACED WITH SPACES.
2327                                    ;*CALL:
2328                                    ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
2329                                    ;*      TYPDS                   ;;GO TO THE ROUTINE
2330
2331  011276                            $TYPDS:
2332  011276  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2333  011300  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2334  011302  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2335  011304  010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2336  011306  010546                           MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2337  011310  012746  020200                   MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
2338  011314  016605  000020                   MOV     20(SP),R5       ;;GET THE INPUT NUMBER
2339  011320  100004                           BPL     1$              ;;BR IF INPUT IS POS.
2340  011322  005405                           NEG     R5              ;;MAKE THE BINARY NUMBER POS.
2341  011324  112766  000055  000001           MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
2342  011332  005000                    1$:    CLR     R0              ;;ZERO THE CONSTANTS INDEX
2343  011334  012703  011512                   MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
2344  011340  112723  000040                   MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
2345  011344  005002                    2$:    CLR     R2              ;;CLEAR THE BCD NUMBER
2346  011346  016001  011502                   MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
2347  011352  160105                    3$:    SUB     R1,R5           ;;FORM THIS BCD DIGIT
2348  011354  002402                           BLT     4$              ;;BR IF DONE
2349  011356  005202                           INC     R2              ;;INCREASE THE BCD DIGIT BY 1
2350  011360  000774                           BR      3$
2351  011362  060105                    4$:    ADD     R1,R5           ;;ADD BACK THE CONSTANT
2352  011364  005702                           TST     R2              ;;CHECK IF BCD DIGIT=0
2353  011366  001002                           BNE     5$              ;;FALL THROUGH IF 0
2354  011370  105716                           TSTB    (SP)            ;;STILL DOING LEADING 0'S?
2355  011372  100407                           BMI     7$              ;;BR IF YES
2356  011374  106316                    5$:    ASLB    (SP)            ;;MSD?
2357  011376  103003                           BCC     6$              ;;BR IF NO
2358  011400  116663  000001  177777           MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
2359  011406  052702  000060            6$:    BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
2360  011412  052702  000040            7$:    BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
2361  011416  110223                           MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
2362  011420  005720                           TST     (R0)+           ;;JUST INCREMENTING
2363  011422  020027  000010                   CMP     R0,#10          ;;CHECK THE TABLE INDEX
2364  011426  002746                           BLT     2$              ;;GO DO THE NEXT DIGIT
2365  011430  003002                           BGT     8$              ;;GO TO EXIT
2366  011432  010502                           MOV     R5,R2           ;;GET THE LSD
2367  011434  000764                           BR      6$              ;;GO CHANGE TO ASCII
2368  011436  105726                    8$:    TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
2369  011440  100003                           BPL     9$              ;;BR IF NO
2370  011442  116663  177777  177776           MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
2371  011450  105013                    9$:    CLRB    (R3)            ;;SET THE TERMINATOR
2372  011452  012605                           MOV     (SP)+,R5        ;;POP STACK INTO R5
2373  011454  012603                           MOV     (SP)+,R3        ;;POP STACK INTO R3
```

```
2374  011456  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2375  011460  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2376  011462  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2377  011464  104400  011512                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
2378  011470  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
2379  011476  012616                          MOV     (SP)+,(SP)
2380  011500  000002                          RTI                     ;;RETURN TO USER
2381  011502  023420                  $DTBL:  10000.
2382  011504  001750                          1000.
2383  011506  000144                          100.
2384  011510  000012                          10.
2385  011512  000004                  $DBLK:  .BLKW   4
2386                                  .SBTTL  ERROR HANDLER ROUTINE
2387
2388                                  ;;**********************************************************************
2389                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2390                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2391                                  ;*AND GO TO SWRCK ON ERROR
2392                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2393                                  ;*SW15=1         HALT ON ERROR
2394                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
2395                                  ;*SW10=1         BELL ON ERROR
2396                                  ;*SW09=1         LOOP ON ERROR
2397                                  ;*CALL
2398                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
2399
2400  011522                          $ERROR:
2401  011522  104406                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2402  011524  104406                          CKSWR           ;GO LOOK FOR SWR CHANGE
2403  011526  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
2404  011532  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
2405  011534  013777  001102  167400          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
2406  011542  032777  002000  167370          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
2407  011550  001402                          BEQ     1$              ;;NO - SKIP
2408  011552  104400  001164                  TYPE    ,$BELL          ;;RING BELL
2409  011556  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
2410  011562  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
2411  011566  162737  000002  001116          SUB     #2,$ERRPC
2412  011574  117737  167316  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
2413  011602  032777  020000  167330          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
2414  011610  001004                          BNE     20$             ;;SKIP TYPEOUTS
2415  011612  004737  011712                  JSR     PC,SWRCK        ;;GO TO USER ERROR ROUTINE
2416  011616  104400  001171                  TYPE    ,$CRLF
2417  011622                          20$:
2418  011622  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
2419  011630  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
2420  011632  113737  001114  011644          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
2421  011640  004737  010620                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
2422  011644     000          21$:    .BYTE   0
2423  011645     000                  .BYTE   0
2424  011646  000777          22$:    BR      22$             ;;APT ERROR LOOP
2425  011650  005777  167264  2$:     TST     @SWR            ;;HALT ON ERROR
2426  011654  100002                  BPL     3$              ;;SKIP IF CONTINUE
2427  011656  000000                  HALT                    ;;HALT ON ERROR!
```

```
2428  011660  104406                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2429  011662  032777  001000  167250  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
2430  011670  001402                          BEQ     4$              ;;BR IF NO
2431  011672  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
2432  011676  005737  001162          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
2433  011702  001402                          BEQ     5$              ;;BR IF NONE
2434  011704  013716  001162                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
2435  011710                          5$:
2436  011710  000002                          RTI                     ;;RETURN
2437                                  ;;**************************************************************
2438                                  ;GO TYPE ERROR
2439                                  ;GO UPDATE SOFTWARE SWR IF 'CNTRL/G'
2440                                  ;;**************************************************************
2441  011712  113737  001102  001534  $WRCK:  MOVB    $TSTNM,TSTNUM   ;SET UP TEST # ON ER
2442  011720  004737  011730                  JSR     PC,$ERRTYP      ;GO TYPE ERROR
2443  011724  104406                          CKSWR                   ;GO LOOK FOR SWR CHANGE
2444  011726  000207                          RTS     PC              ;RETURN TO ERROR HANDLER
2445                                  .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
2446
2447                                  ;;**************************************************************
2448                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
2449                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
2450                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
2451
2452  011730                          $ERRTYP:
2453  011730  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2454  011734  010046                          MOV     R0,-(SP)        ;;SAVE R0
2455  011736  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
2456  011740  153700  001114                  BISB    @#$ITEMB,R0
2457  011744  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
2458                                                                  ;;TYPE THE PC OF THE ERROR
2459  011746  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
2460                                                                  ;;ERROR ADDRESS
2461  011752  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2462  011754  000426                          BR      6$              ;;GET OUT
2463  011756  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
2464  011760  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
2465  011762  006300                          ASL     R0
2466  011764  006300                          ASL     R0
2467  011766  062700  001320                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
2468  011772  012037  012002                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
2469  011776  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
2470  012000  104400                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
2471  012002  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
2472  012004  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2473  012010  012037  012020          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
2474  012014  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
2475  012016  104400                          TYPE                    ;;TYPE THE "DATA HEADER"
2476  012020  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
2477  012022  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
2478  012026  011000                  5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
2479  012030  001004                          BNE     7$              ;;GO TYPE THE DATA
2480  012032  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
2481  012034  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
```

```
2482  012040  000207                        RTS     PC              ;;RETURN
2483  012042                        7$:
2484  012042  013046                        MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
2485  012044  104401                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2486  012046  005710                        TST     (R0)            ;;IS THERE ANOTHER NUMBER?
2487  012050  001770                        BEQ     6$              ;;BR IF NO
2488  012052  104400  012060              TYPE    ,8$             ;;TYPE TWO(2) SPACES
2489  012056  000771                        BR      7$              ;;LOOP
2490  012060  020040     000        8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
2491          012064                        .EVEN
2492                                .SBTTL  SCOPE HANDLER ROUTINE
2493
2494                                ;;**************************************************************
2495                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
2496                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
2497                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
2498                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2499                                ;*SW14=1         LOOP ON TEST
2500                                ;*SW11=1         INHIBIT ITERATIONS
2501                                ;*SW09=1         LOOP ON ERROR
2502                                ;*SW08=1         LOOP ON TEST IN SWR<7:0>
2503                                ;*CALL
2504                                ;*      SCOPE           ;;SCOPE=IOT
2505
2506  012064                        $SCOPE:
2507  012064  104406                        CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
2508  012066  104406                        CKSWR                   ;GO LOOK FOR SWR CHANGE
2509  012070  032777  040000  167042  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
2510  012076  001114                        BNE     $OVER           ;;YES IF SW14=1
2511                                ;*****START OF CODE FOR THE XOR TESTER*****
2512  012100  000416                $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
2513                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
2514  012102  013746  000004               MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
2515  012106  012737  012126  000004        MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
2516  012114  005737  177060               TST     @#177060        ;;TIME OUT ON XOR?
2517  012120  012637  000004               MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
2518  012124  000463                        BR      $SVLAD          ;;GO TO THE NEXT TEST
2519  012126  022626                5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
2520  012130  012637  000004               MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
2521  012134  000423                        BR      7$              ;;LOOP ON THE PRESENT TEST
2522  012136                        6$:;*****END OF CODE FOR THE XOR TESTER*****
2523  012136  032777  000400  166774        BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
2524  012144  001404                        BEQ     2$              ;;BR IF NO
2525  012146  127737  166766  001102        CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
2526  012154  001465                        BEQ     $OVER           ;;BR IF YES
2527  012156  105737  001103         2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
2528  012162  001421                        BEQ     3$              ;;BR IF NO
2529  012164  123737  001115  001103        CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
2530  012172  101015                        BHI     3$              ;;BR IF NO
2531  012174  032777  001000  166736        BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
2532  012202  001404                        BEQ     4$              ;;BR IF NO
2533  012204  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
2534  012212  000446                        BR      $OVER
2535  012214  105037  001103         4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
```

```
2536  012220  005037  001160                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
2537  012224  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
2538  012226  032777  004000  166704  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
2539  012234  001011                          BNE     1$              ;;BR IF YES
2540  012236  005737  001202                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
2541  012242  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
2542  012244  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
2543  012250  023737  001160  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
2544  012256  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
2545  012260  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
2546  012266  013737  012344  001160          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
2547  012274  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
2548  012300  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
2549  012306  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
2550  012312  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
2551  012316  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
2552  012322  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
2553  012330  013777  001102  166604  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
2554  012336  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
2555  012342  000002                          RTI                     ;;FIXES PS
2556  012344  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
2557                                  .SBTTL  TTY INPUT ROUTINE
2558
2559                                  ;;**************************************************************
2560                                  .ENABL  LSB
2561
2562                                  ;;**************************************************************
2563                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
2564                                  ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
2565                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
2566                                  ;*WHEN OPERATING IN TTY FLAG MODE.
2567  012346  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
2568  012354  001074                          BNE     15$             ;;BRANCH IF NO
2569  012356  105777  166562                  TSTB    @$TKS           ;;CHAR THERE?
2570  012362  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
2571  012364  117746  166556                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
2572  012370  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
2573  012374  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
2574  012400  001062                          BNE     15$             ;;NO, RETURN TO USER
2575  012402  123727  001134  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
2576  012410  001456                          BEQ     15$             ;;BRANCH IF YES
2577
2578  012412  104400  013073                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
2579  012416  104400  013100          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
2580  012422  013746  000176                  MOV     $SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
2581  012426  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2582  012430  104400  013111                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
2583  012434  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
2584  012436  005046                          CLR     -(SP)           ;;THE NEW SWR
2585  012440  105777  166500          7$:     TSTB    @$TKS           ;;CHAR THERE?
2586  012444  100375                          BPL     7$              ;;IF NOT TRY AGAIN
2587
2588  012446  117746  166474                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
2589  012452  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
```

```
2590
2591
2592
2593  012456  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
2594  012462  001005                          BNE     10$             ;;BRANCH IF NOT
2595  012464  104400  013066                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (↑U)
2596  012470  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
2597  012474  000757                          BR      19$             ;;LET'S TRY IT AGAIN
2598
2599
2600  012476  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
2601  012502  001022                          BNE     16$             ;;BRANCH IF NO
2602  012504  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
2603  012510  001403                          BEQ     11$             ;;BRANCH IF YES
2604  012512  016677  000002  166420          MOV     2(SP),@SWR      ;;SAVE NEW SWR
2605  012520  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
2606  012524  104400  001171          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
2607  012530  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
2608  012536  001003                          BNE     15$             ;;BRANCH IF NOT
2609  012540  012777  000100  166376          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
2610  012546  000002          15$:    RTI                             ;;RETURN
2611  012550  004737  010532          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
2612  012554  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
2613  012560  002420                          BLT     18$             ;;BRANCH IF YES
2614  012562  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
2615  012566  003015                          BGT     18$             ;;BRANCH IF YES
2616  012570  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
2617  012574  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
2618  012600  001403                          BEQ     17$             ;;BRANCH IF YES
2619  012602  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
2620  012604  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
2621  012606  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
2622  012610  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
2623  012614  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
2624  012620  000707                          BR      7$              ;;GET THE NEXT ONE
2625  012622  104400  001170          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
2626  012626  000720                          BR      20$             ;;SIMULATE CONTROL-U
2627                                  .DSABL  LSB
2628
2629
2630                                  ;;**************************************************************
2631                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2632                                  ;*CALL:
2633                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
2634                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
2635                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
2636                                  ;
2637
2638  012630  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
2639  012632  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
2640  012640  105777  166300          1$:     TSTB    @$TKS           ;;WAIT FOR
2641  012644  100375                          BPL     1$              ;;A CHARACTER
2642  012646  117766  166274  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
2643  012654  042766  177600  000004          BIC     #↑C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
```

```
2644  012662  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
2645  012670  001013                          BNE     3$              ;;BRANCH IF NO
2646  012672  105777  166246          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
2647  012676  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
2648  012700  117746  166242                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
2649  012704  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
2650  012710  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
2651  012714  001366                          BNE     2$              ;;IF NOT DISCARD IT
2652  012716  000750                          BR      1$              ;;YES, RESUME
2653  012720  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
2654  012726  002407                          BLT     4$              ;;BRANCH IF YES
2655  012730  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
2656  012736  003003                          BGT     4$              ;;BRANCH IF YES
2657  012740  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
2658  012746  000002                  4$:     RTI                     ;;GO BACK TO USER
2659                                  ;;**********************************************************************
2660                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2661                                  ;*CALL:
2662                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
2663                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2664                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2665
2666  012750  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
2667  012752  012703  013056          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
2668  012756  022703  013066          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
2669  012762  101405                          BLOS    4$              ;;BR IF YES
2670  012764  104407                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2671  012766  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
2672  012770  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
2673  012774  001003                          BNE     3$              ;;SKIP IF NOT
2674  012776  104400  001170          4$:     TYPE    ,$QUES          ;;TYPE A '?'
2675  013002  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
2676  013004  111337  013054          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2677  013010  104400  013054                  TYPE    ,9$
2678  013014  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2679  013020  001356                          BNE     2$              ;;LOOP IF NOT RETURN
2680  013022  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2681  013026  104400  001172                  TYPE    ,$LF            ;;TYPE A LINE FEED
2682  013032  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2683  013034  011546                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2684  013036  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2685  013044  012766  013056  000004          MOV     #$TTYIN,4(SP)
2686  013052  000002                          RTI                     ;;RETURN
2687  013054     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2688  013055     000                          .BYTE   0               ;;TERMINATOR
2689  013056  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
2690  013066  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
2691  013073     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
2692  013100  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
2693  013106  020075     000
2694  013111     040  047040  053505  $MNEW:  .ASCIZ  /  NEW =
2695  013116  036440  000040
2696                                  .SBTTL  POWER DOWN AND UP ROUTINES
2697
```

```
2698                                    ;;**********************************************************************
2699                                    ;POWER DOWN ROUTINE
2700  013122  012737  013266  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2701  013130  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2702  013136  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2703  013140  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2704  013142  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2705  013144  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2706  013146  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2707  013150  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2708  013152  017746  165762                 MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
2709  013156  010637  013272                 MOV     SP,$SAVR6       ;;SAVE SP
2710  013162  012737  013174  000024         MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
2711  013170  000000                         HALT
2712  013172  000776                         BR      .-2             ;;HANG UP
2713
2714                                    ;;**********************************************************************
2715                                    ;POWER UP ROUTINE
2716  013174  012737  013266  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
2717  013202  013706  013272                 MOV     $SAVR6,SP       ;;GET SP
2718  013206  005037  013272                 CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
2719  013212  005237  013272          1$:    INC     $SAVR6          ;;WAIT FOR THE INC
2720  013216  001375                         BNE     1$              ;;OF  WORD
2721  013220  012677  165714                 MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
2722  013224  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
2723  013226  012604                         MOV     (SP)+,R4        ;;POP STACK INTO R4
2724  013230  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
2725  013232  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
2726  013234  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
2727  013236  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
2728  013240  012737  013122  000024         MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
2729  013246  012737  000340  000026         MOV     #340,@#PWRVEC+2 ;;PRIO:7
2730  013254  104400                         TYPE                    ;;REPORT THE POWER FAILURE
2731  013256  013274                 $PWRMG: .WORD   PWRMSG           ;;POWER FAIL MESSAGE POINTER
2732  013260  012716                         MOV     (PC)+,(SP)      ;;RESTART AT RESTRT
2733  013262  002330                 $PWRAD: .WORD   RESTRT           ;;RESTART ADDRESS
2734  013264  000002                         RTI
2735  013266  000000                 $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
2736  013270  000776                         BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
2737  013272  000000                 $SAVR6: 0                       ;;PUT THE SP HERE
2738  013274  005015  042522  052123 PWRMSG: .ASCIZ  <15><12>/RESTARTED FROM PWR FAIL/
2739  013302  051101  042524  020104
2740  013310  051106  046517  050040
2741  013316  051127  043040  044501
2742  013324  000114
2743                                    .EVEN
2744                                    .SBTTL  TRAP DECODER
2745
2746                                    ;;**********************************************************************
2747                                    ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2748                                    ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2749                                    ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2750                                    ;*GO TO THAT ROUTINE.
2751
```

```
2752  013326  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2753  013330  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
2754  013334  005740                          TST     -(R0)           ;;BACKUP BY 2
2755  013336  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2756  013340  006300                          ASL     R0              ;;POSITION FOR INDEXING
2757  013342  016000  013350                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2758  013346  000200                          RTS     R0              ;;GO TO ROUTINE
2759
2760                                  .SBTTL  TRAP TABLE
2761
2762                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2763                                  ;*BY THE "TRAP" INSTRUCTION.
2764
2765                                  ;       ROUTINE
2766                                  ;       -------
2767  013350                          $TRPAD:
2768  013350  010320                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
2769  013352  011074                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2770  013354  011050                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2771  013356  011110                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2772  013360  011276                          $TYPDS  ;;CALL=TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
2773
2774  013362  012416                          $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
2775
2776  013364  012346                          $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
2777  013366  012630                          $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
2778  013370  012750                          $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
2779
2780                                  .SBTTL  ASCII MESSAGES
2781  013372  042522  020107  044524  EM1:    .ASCIZ  /REG TIMEOUT ER/
2782  013400  042515  052517  020124
2783  013406  051105     000
2784  013411     122  043505  051040  EM2:    .ASCIZ  'REG READ/WRITE ER'
2785  013416  040505  027504  051127
2786  013424  052111  020105  051105
2787  013432     000
2788  013433     102  051525  051040  EM3:    .ASCIZ  /BUS RESET ER/
2789  013440  051505  052105  042440
2790  013446  000122
2791  013450  047106  052103  041040  EM4:    .ASCIZ  /FNCT BITS FAILED TO SET STAT BITS/
2792  013456  052111  020123  040506
2793  013464  046111  042105  052040
2794  013472  020117  042523  020124
2795  013500  052123  052101  041040
2796  013506  052111  000123
2797  013512  042522  042101  020131  EM5:    .ASCIZ  /READY INTR FAILURE/
2798  013520  047111  051124  043040
2799  013526  044501  052514  042522
2800  013534     000
2801  013535     122  040505  054504  EM6:    .ASCIZ  /READY CLR OR SET ER/
2802  013542  041440  051114  047440
2803  013550  020122  042523  020124
2804  013556  051105     000
2805  013561     123  040524  052524  EM7:    .ASCIZ  /STATUS ER ON XFER/
```

```
2806  013566  020123  051105  047440
2807  013574  020116  043130  051105
2808  013602     000
2809  013603     127  051117  020104  EM10:   .ASCIZ  /WORD COUNT ER ON XFER/
2810  013610  047503  047125  020124
2811  013616  051105  047440  020116
2812  013624  043130  051105     000
2813  013631     102  043125  042506  EM11:   .ASCIZ  /BUFFER ADRS ER ON XFER/
2814  013636  020122  042101  051522
2815  013644  042440  020122  047117
2816  013652  054040  042506  000122
2817  013660  040504  040524  042440  EM12:   .ASCIZ  /DATA ER FROM MEM/
2818  013666  020122  051106  046517
2819  013674  046440  046505     000
2820  013701     104  052101  020101  EM13:   .ASCIZ  /DATA ER TO MEM/
2821  013706  051105  052040  020117
2822  013714  042515  000115
2823  013720  044523  043516  042514  EM14:   .ASCIZ  /SINGLE CYCLE OFF DID NOT LOCK OUT CPU/
2824  013726  041440  041531  042514
2825  013734  047440  043106  042040
2826  013742  042111  047040  052117
2827  013750  046040  041517  020113
2828  013756  052517  020124  050103
2829  013764  000125
2830  013766  044523  043516  042514  EM15:   .ASCIZ  /SINGLE CYCLE ON LOCKED OUT CPU/
2831  013774  041440  041531  042514
2832  014002  047440  020116  047514
2833  014010  045503  042105  047440
2834  014016  052125  041440  052520
2835  014024     000
2836  014025     015  050012  042514  WARN:   .ASCII  <15><12>/PLEASE DISABLE "REV11" MEMORY REFRESH OPTION/
2837  014032  051501  020105  044504
2838  014040  040523  046102  020105
2839  014046  051042  053105  030461
2840  014054  020042  042515  047515
2841  014062  054522  051040  043105
2842  014070  042522  044123  047440
2843  014076  052120  047511     116
2844  014103     015  040412  042116          .ASCIZ  <15><12>/AND ENABLE PROCESSOR MEMORY REFRESH   /
2845  014110  042440  040516  046102
2846  014116  020105  051120  041517
2847  014124  051505  047523  020122
2848  014132  042515  047515  054522
2849  014140  051040  043105  042522
2850  014146  044123  020040  000040
2851  014154  042516  020130  047514  EM16:   .ASCIZ  /NEX LOGIC ER/
2852  014162  044507  020103  051105
2853  014170     000
2854  014171     103  041531  042514  EM17:   .ASCIZ  /CYCLE FAILED TO CLK DBR (IN)/
2855  014176  043040  044501  042514
2856  014204  020104  047524  041440
2857  014212  045514  042040  051102
2858  014220  024040  047111  000051
2859  014226  040504  040524  042440  EM20:   .ASCIZ  "DATA ER FROM I/O PAGE (XCSR)"
```

```
2860  014234  020123  051106  046517
2861  014242  044440  047457  050040
2862  014250  043501  020105  054050
2863  014256  051503  024522     000
2864  014263     105  051122  041520  DH1:    .ASCIZ  ERRPC   TSTNUM  BUSADR  EXPCT   RCVD/
2865  014270  020040  052040  052123
2866  014276  052516  020115  041040
2867  014304  051525  042101  020122
2868  014312  042440  050130  052103
2869  014320  020040  051040  053103
2870  014326  000104
2871  014330  051105  050122  020103  DH2:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  ADRS    EXPCT   RCVD/
2872  014336  020040  051524  047124
2873  014344  046525  020040  052502
2874  014352  040523  051104  020040
2875  014360  042101  051522  020040
2876  014366  020040  054105  041520
2877  014374  020124  020040  041522
2878  014402  042126     000
2879  014405     105  051122  041520  DH3:    .ASCIZ  /ERRPC   TSTNUM  BUSADR/
2880  014412  020040  052040  052123
2881  014420  052516  020115  041040
2882  014426  051525  042101  000122
2883
2884                                          .EVEN
2885  014434  001116  001534  001122  DT1:    $ERRPC,TSTNUM,$BDADR,$GDDAT,$BDDAT,0
2886  014442  001124  001126  000000
2887  014450  001116  001534  001122  DT2:    $ERRPC,TSTNUM,$BDADR,$GDADR,$GDDAT,$BDDAT,0
2888  014456  001120  001124  001126
2889  014464  000000
2890  014466  001116  001534  001122  DT3:    $ERRPC,TSTNUM,$BDADR,0
2891  014474  000000
2892                                          ;;**************************************************************
2893                                          ;'DBUF' IS THE WORKING AREA IN EACH 4K MEM FOR ALL
2894                                          ;NPR OPERATIONS - IT IS 200 WORDS LONG
2895                                          ;;**************************************************************
2896  014476  000000                          DBUF:   0               ;1ST ADRS OF DATA BUFFER
2897          000001                                  .END
```

DVDRAA.P11 CROSS REFERENCE TABLE

```
ABASE = 172410          748#    846     887
ACDW1 = 000000          946     889
ACDW2 = 000000          846     890
ACPUOP= 000000          946     861
ADDW0 = 000000          846     891
ADDW1 = 000000          846     892
ADDW10= 000000          846     901
ADDW11= 000000          846     902
ADDW12= 000000          846     903
ADDW13= 000000          946     904
ADDW14= 000000          846     905
ADDW15= 000000          846     906
ADDW2 = 000000          846     893
ADDW3 = 000000          946     894
ADDW4 = 000000          846     895
ADDW5 = 000000          846     896
ADDW6 = 000000          846     897
ADDW7 = 000000          846     898
ADDW8 = 000000          846     899
ADDW9 = 000000          846     900
ADEVCT= 000000          846     852
ADEVM = 000001          750#    846     889
AENV  = 000000          846     857
AENVM = 000000          846     858
AFATAL= 000000          846     849
AMADR1= 000000          846     874
AMADR2= 000000          846     878
AMADR3= 000000          846     881
AMADR4= 000000          846     884
AMAMS1= 000000          846     868
AMAMS2= 000000          846     876
AMAMS3= 000000          846     879
AMAMS4= 000000          846     882
AMSGAD= 000000          846     854
AMSGLG= 000000          846     855
AMSGTY= 000000          846     848
AMTYP1= 000000          846     869
AMTYP2= 000000          846     877
AMTYP3= 000000          846     880
AMTYP4= 000000          846     883
APASS = 000000          846     851
APRIOR= 000000          846
APTCSU= 000040         2136    2241#
APTENV= 000001         2129    2197    2239#   2416
APTSIZ= 000200         1079    2238#
APTSPO= 000100         2131    2199    2240#
ASWREG= 000000          846     859
ATESTN= 000000          846     850
AUNIT = 000000          846     853
AUSWR = 000000          846     860
AVECT1= 000124          749#    846     885
AVECT2= 000000          846     886
BIT0  = 000001          733#
BIT00 = 000001          723#    733     1196    1201    1260    2000
```

DVDRAA.P11 CROSS REFERENCE TABLE

```
BIT01 = 000002            722#    732
BIT02 = 000004            721#    731
BIT03 = 000010            720#    730
BIT04 = 000020            719#    729
BIT05 = 000040            718#    728
BIT06 = 000100            717#    727
BIT07 = 000200            716#    726
BIT08 = 000400            715#    725     2523
BIT09 = 001000            714#    724     2429    2531
BIT1  = 000002            732#
BIT10 = 002000            713#    2406
BIT11 = 004000            712#    2538
BIT12 = 010000            711#
BIT13 = 020000            710#    1704    2413
BIT14 = 040000            709#    2509
BIT15 = 100000            708#
BIT2  = 000004            731#
BIT3  = 000010            730#
BIT4  = 000020            729#
BIT5  = 000040            728#
BIT6  = 000100            727#    1360
BIT7  = 000200            726#
BIT8  = 000400            725#
BIT9  = 001000            724#
BPTVEC= 000014            740#
CKDAT   010150           1596    2061#
CKDAT1  010216           1710    1809    1851    2080#
CKSTAT  007646           1456    1502    1548    1586    1692    1799    1841    1878    1982#
CKSWR = 104406           2401    2402    2428    2443    2507    2508    2776#
CORSZ   001540           1037#   1115    1126*   1823
CORSZR  002226           1116    1121#
CR    = 000015            648#   2175    2185
CRLF  = 000200            649#   1114    2146    2185
DBUF    014476           1038    1128    1444    1445*   1465    1491    1512    1514    1537    1557    1574    1611    1644
                         1680    1784    1856    1895    2015    2019    2024    2061    2068    2896#
DBUFP   001542           1038#   1128*   1821*   1822*   1823    1826    1856*   1865*   1895*   1998    2038    2081
DDISP = 177570            655#    827    1067
DH1     014263            928     934     940     946     952     958     963     969     975    1005    1011    2864#
DH2     014330            981     987    1016    2871#
DH3     014405            993     999    2879#
DISPLA  001142            827#   1067*   1075*   2405*   2553*
DISPRE  000174            759#   1075
DMAP    001536           1036#   1130*   1131*   1133    1903*   1912
DRVBAR  001522           1024#   1191    1194*   1195    1252*   1259    1262    1400    1402*   1403    1408    1444*   1491*
                         1537*   1574*   1611*   1644*   1680*   1725*   1751    1755    1784*   1826*   1866*   1906*   1999
                         2003
DRVCSR  001524           1025#   1278    1282*   1283    1293    1303    1305*   1307    1316    1334    1337*   1338    1355
                         1360*   1362    1366    1370    1381    1383*   1384    1388*   1390    1399*   1407*   1412*   1421*
                         1427*   1446*   1447*   1449    1451    1452*   1493*   1494*   1496    1498    1539*   1540*   1541
                         1543    1544*   1577*   1578*   1579    1581    1582*   1605    1613*   1614*   1621    1622*   1638
                         1646*   1647*   1652    1653*   1662    1683*   1684*   1685    1687    1688*   1727*   1728*   1729
                         1731    1732*   1736    1740    1760*   1761    1765    1769    1773    1787*   1788*   1792    1794
                         1795*   1829*   1830*   1834    1836    1837*   1869*   1870*   1871    1873    1874*   1907*   1970*
                         1982    1983*   1987
```

DVDRAA.P11 CROSS REFERENCE TABLE

```
DRVCTO  001530          1030#  1089   1354*  1359*  1909*  1962*  1971*
DRVCT2  001532          1031#  1094   1910*  1963*  1971   1972*
DRVDBR  001526          1026#  1087   1214   1217*  1218   1233   1253*  1264   1266   1418   1420*  1422*  1423
                        1429   1467   1470   1492*  1517   1559   1562   1575*  1868*  1889   1892   1908*  2064
                        2095
DRVWCR  001520          1023#  1083   1141   1154   1171   1174*  1175   1251*  1255   1257   1443*  1490*  1536*
                        1573*  1610*  1643*  1681*  1682*  1724*  1744   1748   1785*  1827*  1864*  1905*  1990
                        1992
DSWR  = 177570           654#   826   1066
DT1     014434           929    935    941    947    953    959    964    970    976    1006   1012   2885#
DT2     014450           982    988   1017   2887#
DT3     014466           994   1000   2890#
EMTVEC= 000030           743#  1051*  1052*
EM1     013372           927   2781#
EM10    013603           968   2809#
EM11    013631           974   2813#
EM12    013660           980   2817#
EM13    013701           986   2820#
EM14    013720           992   2823#
EM15    013766           998   2830#
EM16    014154          1004   2851#
EM17    014171          1010   2854#
EM2     013411           933   2784#
EM20    014226          1015   2859#
EM3     013433           939   2788#
EM4     013450           945   2791#
EM5     013512           951   2797#
EM6     013535           957   2801#
EM7     013561           962   2805#
ERRVEC= 000004           736#  1064   1065*  1076*  1122*  1127*  1153*  1164*  2514   2515*  2517*  2520*
GMS   = ****** U         758   1113   2768   2769   2770   2771   2772   2774   2776   2777   2778
GTSWR = 104405          1108   2774#
HT    = 000011           646#  2144   2185
IOTVEC= 000020           741#  1049*  1050*
LDBUF   010026          1535   2015#
LDBUF1  010100          1676   1783   1825   2038#
LF    = 000012           647#  2179   2185
MTPS  = 106427           751#  1136   1274   1352   1357   1380   1448   1495   1538   1576   1612   1645   1679
                        1726   1786   1828   1867   1961   1973
NXDEV   007362          1901#
NXDEV1  007364          1135   1902#  1913
PC    =%000007           667#  1296*  1374*  1479*  1526*  1535*  1564*  1596*  1598*  1631*  1667*  1676*  1714*
                        1775*  1783*  1813*  1825*  1857*  1896*  1929*  1932*  1942*  1947   1974*  2022*  2049*
                        2067*  2071*  2134*  2153*  2160*  2167*  2181*  2183*  2216*  2233*  2415*  2421*  2442*
                        2444*  2482*  2611*  2732
PIRQ  = 177772           653#
PIRQVE= 000240           747#
PR0   = 000000           670#
PR1   = 000040           671#
PR2   = 000100           672#
PR3   = 000140           673#
PR4   = 000200           674#
PR5   = 000240           675#
PR6   = 000300           676#
```

DVDRAA.P11 CROSS REFERENCE TABLE

DVDRAA.P11 CROSS REFERENCE TABLE

```
                        2732*   2752*   2753
STACK = 001100           641*    1047    1137
START   001544           762     1040*
START1  002006          1083*    1948
STKLMT= 177774           652*
SWR     001140           926*    1045    1066*   1068    1074*   1081*   1106    2406    2413    2425    2429    2509    2523
                        2525     2531    2538    2567    2604*   2708    2721*
SWRCK   011712          2415     2441*
SWREG   000176           760*    1074    1106    2567    2580
SW0   = 000001           705*
SW00  = 000001           695*     705
SW01  = 000002           694*     704
SW02  = 000004           693*     703
SW03  = 000010           692*     702
SW04  = 000020           691*     701
SW05  = 000040           690*     700
SW06  = 000100           689*     699
SW07  = 000200           688*     698
SW08  = 000400           687*     697
SW09  = 001000           686*     696
SW1   = 000002           704*
SW10  = 002000           685*
SW11  = 004000           684*
SW12  = 010000           683*
SW13  = 020000           682*
SW14  = 040000           681*
SW15  = 100000           680*
SW2   = 000004           703*
SW3   = 000010           702*
SW4   = 000020           701*
SW5   = 000040           700*
SW6   = 000100           699*
SW7   = 000200           698*
SW8   = 000400           697*
SW9   = 001000           696*
TBITVE= 000014           738*
TKVEC = 000060           745*
TPVEC = 000064           746*
TRAPVE= 000034           744*    1053*   1054*
TRTVEC= 000014           739*
TSTNUM  001534          1035*    2441*   2885    2887    2890
TST1    002364          1146*
TST10   003364          1301*
TST11   003440          1309    1315*
TST12   003522          1332*
TST13   003620          1350*
TST14   003760          1379*
TST15   004062          1392    1398*
TST16   004164          1417*
TST17   004300          1431    1437*
TST2    002474          1169*
TST20   004530          1484*
TST21   004752          1531*
TST22   005160          1569*
```

```
TST23   005334          1603*
TST24   005510          1636*
TST25   005712          1672*
TST26   006150          1719*
TST27   006530          1779*
TST3    002572          1189*
TST30   006722          1817*
TST31   007146          1861*
TST4    002710          1205    1212*
TST5    003006          1232*
TST6    003066          1242    1248*
TST7    003214          1265    1272*
TYPDS = 104404          1937    2772*
TYPE  = 104400          1100    1117    1935    1938    2149    2302    2377    2408    2416    2453    2470    2472    2475
                        2477    2481    2488    2578    2579    2582    2595    2606    2625    2674    2677    2681    2730
                        2768*
TYPOC = 104401          2461    2485    2581    2769*
TYPON = 104403          2771*
TYPOS = 104402          2770*
WARN    014025          1117    2836*
$APTHD  001000           786     792*
$ASTAT= ****** U        2219    2234
$ATYC   010626          2190    2192*
$ATY1   010602          2188*
$ATY3   010610          2134    2189*
$ATY4   010620          2191*   2421
$AUTOB  001134           823*   1110*   2575    2696
$BASE   001250           887*   1084
$BDADR  001122           818*   1156*   1171*   1191*   1214*   1234*   1257*   1262*   1266*   1278*   1303*   1317*   1334*
                        1355*   1381*   1400*   1418*   1449*   1470*   1496*   1517*   1541*   1562*   1579*   1605*   1638*
                        1685*   1729*   1740*   1748*   1755*   1765*   1773*   1792*   1834*   1871*   1892*   1987*   1992*
                        2003*   2064*   2095*   2885    2887    2890
$BDDAT  001126           820*   1152*   1175*   1176    1195*   1196*   1197    1218*   1219    1237*   1238    1239    1240*
                        1241    1255*   1259*   1260*   1264*   1283*   1284*   1285    1293*   1294*   1307*   1308    1320*
                        1321    1322    1323*   1324    1338*   1339*   1340    1362*   1366*   1370*   1371    1384*   1385
                        1390*   1391    1403*   1404    1408*   1423*   1424    1429*   1430    1451*   1467*   1468    1498*
                        1514*   1515    1543*   1559*   1560    1581*   1609*   1618*   1621*   1628    1642*   1649*   1652*
                        1659    1662*   1664    1687*   1731*   1736*   1738    1744*   1746    1751*   1753    1761*   1763
                        1769*   1771    1789*   1790*   1794*   1831*   1832*   1836*   1873*   1889*   1890    1982*   1985
                        1990*   1999*   2000*   2001    2065*   2092*   2885    2887
$BELL   001164           838*   2408    2437
$CDW1   001254           889*
$CDW2   001256           890*
$CHARC  010576          2151*   2161*   2168    2177*   2182*
$CKSWR  012346          2567*   2776
$CMTAG  001100           806*   1042    1043    1051    1057    1058
$CM3  = 000000           836*
$CNTLG  013073          2578    2691*
$CNTLU  013066          2595    2690*
$CPUOP  001222           861*
$CRLF   001171           840*   2150    2185    2416    2437    2453    2472    2477    2481    2606    2690
$DBLK   011512          2343    2377    2385*
$DDW0   001260           891*   1141*
$DDW1   001262           892*
```

MO5

DVDRAA.P11 CROSS REFERENCE TABLE

```
$DDW10  001304            901#
$DDW11  001306            902#
$DDW12  001310            903#
$DDW13  001312            904#
$DDW14  001314            905#
$DDW15  001316            906#
$DDW2   001264            893#
$DDW3   001266            894#
$DDW4   001270            895#
$DDW5   001272            896#
$DDW6   001274            897#
$DDW7   001276            898#
$DDW8   001300            899#
$DDW9   001302            900#
$DEVCT  001204            852#   1138*
$DEVM   001252            888#   1130
$DOAGN  007556           1931    1940    1946#
$DTBL   011502           2346    2381#
$ENDAD  007546            772    1942#
$ENDCT  007514           1933#
$ENDMG  007565           1935    1950#
$ENULL  007562           1938    1949#
$ENV    001214            857#   1104    2129    2197    2221    2418
$ENVM   001215            858#   1079    2131    2136    2199
$EOP    007460           1904    1923#
$EOPCT  007506           1930#   1934
$ERFLG  001103            809#   2403*   2437    2497    2527    2529    2535*   2557
$ERMAX  001115            815#   1059*   2529    2552*   2557
$ERROR  011522           1051    2400#
$ERRPC  001116            816#   2410*   2411*   2412    2437    2459    2885    2887    2890
$ERRTB  001320            925#   2467
$ERRTY  011730           2442    2452#
$ERTTL  001112            813#   2409*   2437
$ESCAP  001162            837#   1058*   2432    2434    2437    2551*
$ETABL  001214            856#
$ETEND  001320            798     909#
$FATAL  001176            849#   2225*
$FFLG   011046           2188*   2191*   2219    2228*   2236#
$FILLC  001156            834#   2154    2185
$FILLS  001155            833#   2185
$GDADR  001120            817#   1465*   1512*   1557*   1893*   2066*   2093*   2887
$GDDAT  001124            819#   1151*   1173*   1174    1176    1179*   1182*   1183*   1193*   1194    1197    1200*   1201*
                         1204*   1206*   1216*   1217    1219    1222*   1225*   1226*   1236*   1241    1250*   1280*   1281*
                         1285    1292*   1304*   1308    1319*   1324    1335*   1340    1343*   1356*   1371    1382*   1385
                         1389*   1391    1401*   1404    1410*   1419*   1420    1424    1428*   1430    1450*   1466*   1468
                         1497*   1513*   1515    1542*   1558*   1560    1580*   1620*   1651*   1663*   1664    1686*   1730*
                         1737*   1738    1745*   1746    1752*   1753    1762*   1763    1770*   1771    1793*   1835*   1872*
                         1888*   1890    1984*   1985    1993*   1996*   1997*   1998*   2001    2094*   2885    2887
$GET42  007536           1939#
$GTSWR  012416           2579#   2774
$HD     =  000003         623     624
$HIBTS  001000            793#
$ICNT   001104            810#   2542*   2543    2545*   2556
$ILLUP  013266           2700    2716    2735#
```

```
$INTAG  001135            824#  2607  2696
$ITEMB  001114            814#  2412* 2420  2437  2456
$LF     001172            841#  2185  2437  2681  2690
$LFLG   011045           2229*  2235#
$LPADR  001106            811#  1060* 1147* 2533* 2549* 2554  2556
$LPERR  001110            812#  1061* 1150* 1170* 1190* 1213* 1277* 1333* 1438* 1495* 2431  2533  2550*
                         2556
$MADR1  001226            874#
$MADR2  001232            878#
$MADR3  001236            881#
$MADR4  001242            884#
$MAIL   001174            794    798   847#  1078  1104  1148  2129  2418  2548
$MAMS1  001224            868#
$MAMS2  001230            876#
$MAMS3  001234            879#
$MAMS4  001240            882#
$MBADR  001002            794#
$MFLG   011044           2189*  2195  2230* 2234#
$MNEW   013111           2582   2694#
$MSGAD  001210            854#  2205* 2208
$MSGLG  001212            855#  2210*
$MSGTY  001174            848#  2203  2211* 2223  2227*
$MSWR   013100           2579   2692#
$MTYP1  001225            869#
$MTYP2  001231            877#
$MTYP3  001235            880#
$MTYP4  001241            883#
$MXCNT  012344           2546   2556#
$NULL   001154            832#  2156  2185
$NWTST= 000001           1143#  1166# 1186# 1209# 1229# 1245# 1269# 1298# 1312# 1329# 1347# 1376# 1395#
                         1414#  1434# 1481# 1528# 1566# 1600# 1633# 1669# 1716# 1776# 1814# 1858#
$OCNT   011272           2274*  2303# 2316#
$OMODE  011274           2269*  2273# 2278  2281* 2292* 2318#
$OVER   012330           2510   2526  2534  2544  2553#
$PASS   001202            851#  1078* 1927* 1928* 1936  1949  2540  2557
$PASTM  001006            796#
$PWRAD  013262           2733#
$PWRDN  013122           1055   2700# 2728
$PWRMG  013256           2731#
$PWRUP  013174           2710   2716#
$QUES   001170            839#  2185  2437  2625  2674  2690
$RDCHR  012630           2638#  2777
$RDDEC= ****** U         2779
$RDLIN  012750           2666#  2778
$RDOCT= ****** U         2779
$RDSZ = 000010           2659#
$RTNAD  007560           1948#
$R2A  = ****** U         2779
$SAVRE= ****** U         2779
$SAVR6  013272           2709*  2717  2718* 2719* 2737#
$SCOPE  012064           1049   2506#
$SETUP= 000117           1039#  1048  1049  1051  1053  1055  1057  1058  1060  1100  1101  1925  2401
                         2428   2436  2507  2562  2696
$STUP = 177777           1039#
```

```
$SVLAD  012274          2518    2547*
$SVPC = 000106           770*    775
$SWR  = 167400           623     624*    625*    631     632     633     634     635     636     637     836     837     838
                        1057    1058    1060    1061    1107    1170    1190    1213    1233    1249    1273    1302    1316
                        1333    1351    1380    1399    1418    1438    1485    1532    1570    1604    1637    1673    1720
                        1780    1818    1862    1920    1926    1941    1947    1949    2392    2393    2394    2395    2396
                        2406    2413    2425    2429    2437    2498    2499    2500    2501    2502    2509    2521    2523
                        2524    2527    2528    2529    2536    2537    2538    2550    2553    2556    2734
$SWREG  001216           859*    1081
$SWRMK= 000000           637     638    2502    2503    2525
$TESTN  001200           850*    1148*   2548*
$TIMES  001160           836*    1057*   1249*   1273*   1302*   1351*   1532*   1570*   1604*   1637*   1673*   1720*   1780*
                        1818*   1926*   2536*   2543    2546*   2556
$TKB    001146           829*    2560    2571    2588    2642    2648
$TKS    001144           828*    2560    2569    2585    2609*   2640    2646
$TN   = 000032           623*     626*   1143    1147*   1148    1166    1170*   1186    1190*   1205    1209    1213*   1229
                        1233*   1242    1245    1249*   1265    1269    1273*   1298    1302*   1309    1312    1316*   1329
                        1333*   1347    1351*   1376    1380*   1392    1395    1399*   1414    1418*   1431    1434    1438*
                        1481    1485*   1528    1532*   1566    1570*   1600    1604*   1633    1637*   1669    1673*   1716
                        1720*   1776    1780*   1814    1818*   1858    1862*
$TPB    001152           831*    2174*   2185
$TPFLG  001157           835*    2123    2185
$TPS    001150           830*    1865    1866    1888    1893    2172    2185
$TRAP   013326          1053    2752*
$TRP  = 000011          2760*   2769*   2770*   2771*   2772*   2773*   2774    2775*   2776    2777*   2778*   2779*
$TRPAD  013350          2757    2767*
$TSTM   001004           795*
$TSTNM  001102           808*    1149*   1925*   2405    2437    2441    2497    2525    2547*   2548    2553    2557
$TTYIN  013056          2667    2668    2685    2689*
$TYPBN= ****** U        2773
$TYPDS  011276          2331*   2772
$TYPE   010320          2123*   2215    2760    2768
$TYPEC  010532          2153    2160    2167    2172*   2173    2611
$TYPEX  010600          2178    2180    2183*
$TYPOC  011074          2272*   2769
$TYPON  011110          2271    2274*   2771
$TYPOS  011050          2267*   2770
$UNIT   001206           853*    1129*   1138    1139    1911*
$UNITM  001010           757*
$USWR   001220           860*
$VECT1  001244           885*    1090
$VECT2  001246           886*
$XTSTR  012100          2512*
$$GET4= 000000          1941*
$OFILL  011273          2268*   2272*   2282    2317*
$$LOCAT= ****** U       2415    2509
.     = 014500           754*    758*    763*    770     771*    773*    775*    776*    782     783*    785*    787*    805*
                         842    1046    1060    1061    1114*   1949    1953    2185    2237*   2385*   2437    2491*   2556
                        2557    2560    2689*   2690    2696    2712    2736
.$$STA= ****** U        2189    2192
.$/   = 001000           782*    787
```

DVDRAA.P11 CROSS REFERENCE TABLE

```
COMMEN      1#    748#
ENDCOM      1#    748#
ERROR     642#   1163    1178    1199    1221    1243    1258    1263    1267    1287    1295    1310    1326    1342    1363
         1367    1373    1387    1393    1406    1411    1426    1432    1453    1459    1461    1463    1471    1499    1505
         1528    1510    1518    1545    1551    1553    1555    1563    1583    1589    1592    1594    1597    1623    1630
         1654    1661    1666    1689    1695    1699    1701    1712    1733    1741    1749    1756    1766    1774    1796
         1802    1805    1807    1811    1838    1844    1847    1849    1853    1875    1881    1884    1886    1894
ESCAPE      1#    748#
GETPRI      1#    748#
GETSWR      1#    749#   1101#
HLP         1#    748#
MEMST       1#    748#   1143    1166    1186    1209    1229    1245    1269    1298    1312    1329    1347    1376    1395
         1414    1434    1481    1528    1566    1600    1633    1669    1716    1776    1814    1858
OK       1014#   1176    1197    1219    1241    1285    1308    1324    1340    1371    1385    1391    1404    1424    1430
         1468    1515    1560    1664    1738    1746    1753    1763    1771    1890    1985    2001
POP         1#    748#   2231    2232    2372    2721    2722
PUSH        1#    748#   2192    2194    2215    2331    2702    2708
REPORT      1#    748#
SCOPE     643#   1169    1189    1212    1232    1248    1272    1301    1315    1332    1350    1379    1392    1417    1437
         1484    1531    1569    1603    1636    1672    1719    1779    1817    1861    1901
SETPRI      1#    748#
SETTRA    2760#   2769    2770    2771    2772    2774    2776    2777    2778
SETUP       1#    748#   1040
SKIP        1#    748#   1205    1242    1265    1309    1392    1431
SLASH       1#    748#
SPACE     746#
STARS       1#    748#    768     779     781     788     801     842     845    1118    1120    1143    1145    1166    1168
         1186    1188    1209    1211    1229    1231    1245    1247    1269    1271    1298    1300    1312    1314    1323
         1331    1347    1349    1376    1378    1395    1397    1414    1416    1434    1436    1481    1483    1528    1530
         1566    1568    1600    1602    1633    1635    1669    1671    1716    1718    1776    1778    1814    1816    1858
         1860    1838    1900    1917    1956    1960    1966    1969    1976    1981    2009    2014    2032    2037    2056
         2060    2073    2079    2108    2187    2244    2321    2388    2437    2440    2447    2494    2559    2562    2630
         2659    2698    2714    2746    2892    2895
SWPSU       1#    748#   1062#
TRPTRP    2760#
TYPBIN      1#    748#
TYPDEC      1#    748#   1936
TYPNAM      1#    748#   1096
TYPNUM      1#    748#
TYPOCS      1#    613#    748#
TYPOCT      1#    748#   2459    2483    2580
TYPTXT      1#    748#
SSCMRE    799#
SSCMTM    799#
SSESCA      1#    748#
SSNEXT      1#    748#   1143    1166    1186    1209    1229    1245    1269    1298    1312    1329    1347    1376    1395
         1414    1434    1481    1528    1566    1600    1633    1669    1716    1776    1814    1858
SSSET     2760#   2769    2770    2771    2772    2774    2776    2777    2778
SSSETM    1078#
SSSKIP      1#    748#   1205    1242    1265    1309    1392    1431
.EQUAT      1#    613#    638
.HERDE      1#    613#
.PTIM       1#
.SETUP      1#    613#   1039
```

```
.SWRHI      1#    613#    627
.SWRLO    638#
.SACTI      1#    613#    766
.SAPTB      1#    613#    843#
.SAPTH      1#    613#    ###
.SAPTY      1#    613#   2185
.SASTA      1#
.SCATC      1#    613#    752
.SCMTA      1#    613#    799
.SDB2D      1#
.SDB2O      1#
.SDIV       1#
.SEOP       1#    613#   1915
.SERRO      1#    613#   2386
.SERRT      1#    613#   2445
.SMULT      1#
.SPOWE      1#    613#   2696
.SRAND      1#
.SRDDE      1#
.SRDOC      1#    613#
.SREAD      1#    613#   2557
.SR2AZ      1#
.SSAVE      1#
.SSB2D      1#
.SSB2O      1#
.SSCOP      1#    613#   2492
.SSIZE      1#
.SSUPR      1#
.STRAP      1#    613#   2744
.STYPB      1#
.STYPD      1#    613#   2319
.STYPE      1#    613#   2105
.STYPO      1#    613#   2242
.SWDCA      1#
.TITD       1#
```

DVDRAA.P11 CROSS REFERENCE TABLE

```
ADD     1086  1093  1206  1298  1703  1822  1905  1906  1907  1908  1909  1910  1988  1994  1998
        2004  2006  2040  2070  2090  2142  2202  2214  2226  2270  2280  2351  2457  2536  2605
ASL     1140  1182  1204  1225  1476  1523  1957  2027  2051  2100  2464  2465  2466  2619  2620
        2621  2756
ASLB    2356
ASR     2209
BCC     1205  2029  2053  2102  2357
BCS     1184  1227  1478  1525  1660
BEQ     1080  1105  1132  1160  1177  1198  1220  1242  1256  1261  1265  1286  1309  1325  1341
        1372  1386  1392  1405  1423  1425  1431  1469  1516  1561  1629  1665  1705  1739  1747
        1754  1759  1764  1772  1824  1891  1904  1913  1940  1986  1991  2002  2025  2063  2132
        2145  2180  2196  2200  2220  2222  2297  2404  2407  2430  2433  2469  2474  2487  2524
        2526  2528  2532  2541  2576  2603  2618
BGE     2544
BGT     1931  2089  2304  2365  2615  2656
BHI     2530
BIC     1091  1125  1131  1196  1201  1260  1281  1284  1294  1339  1427  1452  1544  1582  1622
        1653  1688  1732  1760  1795  1837  1874  1928  1983  2000  2294  2572  2589  2616  2643
        2649  2657
BIS     1292  1360  1388  1447  1494  1540  1578  1614  1647  1684  1728  1788  1830  1870  2299
        2300  2359  2360  2623
BISB    1238  1239  1321  1322  2456
BIT     1133  1704  1912  2406  2413  2429  2509  2523  2531  2538
BITB    1079  2131  2136  2168  2199
BLOS    2669
BLT     2159  2305  2348  2364  2613  2654
BMI     2355
BNE     1046  1069  1088  1095  1099  1103  1107  1116  1134  1181  1203  1224  1289  1475  1522
        1619  1627  1650  1658  1709  1791  1833  2020  2048  2069  2085  2087  2099  2130  2137
        2139  2147  2155  2169  2176  2198  2204  2207  2224  2295  2353  2414  2419  2457  2479
        2510  2539  2568  2574  2594  2601  2608  2645  2651  2673  2679  2720
BPL     1345  2124  2173  2293  2339  2369  2426  2570  2586  2641  2647
BR      1071  1109  1112  1124  1161  1207  1290  1364  1368  1454  1460  1462  1464  1472  1500
        1507  1509  1511  1519  1546  1552  1554  1556  1584  1591  1593  1595  1624  1655  1690
        1698  1700  1702  1734  1743  1750  1757  1767  1797  1804  1806  1808  1839  1846  1848
        1850  1854  1855  1876  1883  1885  1887  2030  2054  2103  2126  2152  2162  2171  2178
        2190  2212  2271  2286  2307  2350  2367  2424  2462  2489  2512  2518  2521  2534  2537
        2597  2624  2626  2652  2675  2712  2736
BVC     1902
BVS     1044  1057  1058  1078  1151  1152  1172  1192  1215  1235  1250  1318  1327  1382  1402
        1410  1442  1489  1609  1821  1868  1925  1926  1970  1972  1993  2043  2046  2083  2284
        2342  2345  2455  2536  2551  2583  2584  2718
CLRB    2151  2177  2228  2229  2230  2371  2535  2680
CMP     1045  1068  1087  1094  1106  1162  1176  1197  1219  1241  1285  1291  1308  1324  1340
        1358  1361  1365  1369  1371  1385  1391  1404  1424  1430  1455  1468  1501  1515  1547
        1560  1585  1625  1656  1659  1664  1691  1708  1735  1738  1746  1753  1763  1771  1798
        1823  1840  1877  1890  1985  2001  2019  2024  2047  2062  2068  2084  2086  2363  2519
        2543  2567  2573  2593  2600  2612  2614  2644  2650  2653  2655  2668
CMPB    1104  2129  2144  2146  2154  2175  2179  2197  2418  2525  2529  2575  2607  2672  2678
COM     1179  1180  1200  1202  1222  1223  1473  1474  1520  1521  2018  2026  2044  2050  2097
        2096
DEC     1159  1626  1657  1758  1929  2463
DECB    2158  2161  2292  2303
EMT      642
```

DVDRAA.P11 CROSS REFERENCE TABLE

```
.ALT     756  2125  2427  2711  2735
INC     1098  1183  1226  1477  1524  1618  1649  1707  1790  1832  1911  1927  2028  2052  2101
        2227  2298  2306  2349  2409  2542  2622  2719
INCB    2181  2403  2547
IOT      643
JMP      762  1135  1914  1947
JSR     1275  1296  1374  1439  1456  1479  1486  1502  1526  1533  1535  1543  1564  1571  1586
        1596  1598  1606  1631  1639  1667  1674  1676  1692  1710  1714  1722  1775  1781  1783
        1799  1809  1813  1819  1825  1841  1851  1857  1862  1878  1896  1942  2134  2153  2160
        2167  2216  2415  2421  2442  2611
MOV     1043  1047  1049  1050  1051  1052  1053  1054  1055  1056  1060  1061  1064  1065  1066
        1067  1072  1074  1075  1076  1081  1083  1084  1085  1089  1090  1092  1121  1122  1126
        1127  1128  1129  1130  1137  1138  1139  1141  1147  1148  1150  1153  1154  1155  1156
        1164  1170  1171  1173  1174  1175  1180  1191  1193  1194  1195  1213  1214  1216  1217
        1218  1233  1234  1236  1237  1240  1249  1251  1252  1253  1255  1257  1259  1262  1264
        1266  1273  1277  1278  1279  1280  1282  1283  1293  1302  1303  1304  1305  1307  1316
        1317  1319  1320  1323  1333  1334  1335  1336  1337  1338  1351  1354  1355  1356  1359
        1362  1366  1370  1381  1383  1384  1389  1390  1399  1400  1401  1403  1407  1408  1412
        1418  1419  1420  1421  1422  1423  1428  1429  1438  1441  1443  1444  1445  1446  1449
        1450  1451  1465  1466  1467  1470  1485  1488  1490  1491  1492  1493  1496  1497  1498
        1512  1513  1514  1517  1532  1536  1537  1539  1541  1542  1543  1557  1558  1559  1562
        1570  1573  1574  1575  1577  1579  1580  1581  1604  1605  1608  1610  1611  1613  1620
        1621  1637  1638  1641  1642  1643  1644  1646  1651  1652  1662  1663  1673  1677  1678
        1680  1681  1683  1685  1686  1687  1706  1720  1721  1724  1725  1727  1729  1730  1731
        1736  1737  1740  1744  1745  1748  1751  1752  1755  1761  1762  1765  1769  1770  1773
        1780  1784  1785  1787  1789  1792  1793  1794  1818  1826  1827  1829  1831  1834  1835
        1836  1856  1864  1865  1866  1869  1871  1872  1873  1888  1889  1892  1893  1895  1932
        1936  1939  1962  1963  1971  1982  1984  1987  1990  1992  1996  1999  2003  2015  2016
        2017  2021  2023  2038  2039  2041  2042  2045  2061  2064  2065  2066  2080  2081  2082
        2092  2093  2094  2095  2127  2128  2133  2141  2156  2193  2194  2201  2205  2210  2211
        2213  2215  2225  2231  2232  2267  2275  2276  2277  2283  2290  2306  2309  2310  2311
        2312  2332  2333  2334  2335  2336  2337  2338  2343  2346  2366  2372  2373  2374  2375
        2376  2378  2379  2405  2410  2431  2434  2454  2459  2468  2473  2478  2480  2484  2514
        2515  2517  2520  2533  2545  2546  2549  2550  2553  2554  2580  2604  2609  2638  2639
        2666  2667  2682  2683  2684  2685  2700  2701  2702  2703  2704  2705  2706  2707  2708
        2709  2710  2716  2717  2721  2722  2723  2724  2725  2726  2727  2728  2729  2732  2752
        2753  2757
MOVB    1059  1110  1149  2138  2166  2174  2188  2189  2191  2268  2269  2272  2273  2274  2278
        2281  2282  2301  2341  2344  2358  2361  2370  2412  2420  2441  2548  2552  2571  2588
        2642  2648  2671  2676  2755
NEG     1682  2279  2340
NOP     1146  1615  1616  1617  1648  1713  1812  1924  1943  1944  1945
RESET   1142  1254  1306  1353  1768  1941
ROL     2285  2287  2288  2289  2291
ROR     1903
RTI     1073  2143  2313  2380  2436  2555  2610  2658  2686  2734
RTS     1964  1974  1989  1995  2005  2007  2022  2049  2067  2071  2091  2096  2133  2233  2444
        2482  2758
SUB     1343  1344  2088  2208  2347  2411
TRAP    2760  2769  2770  2771  2772  2774  2776  2777  2778
TST     1102  1115  1123  1157  1158  1628  2140  2148  2170  2203  2221  2223  2296  2352  2362
        2425  2432  2486  2515  2540  2602  2617  2754
TSTB    2123  2172  2195  2206  2219  2354  2368  2527  2569  2585  2640  2646
.ASCII   839   840  2836
```

```
            843   846  1047  1099  1100  1119  1121  1143  1144  1145  1146  1147  1167  1168  1169
           1170  1187  1188  1189  1190  1206  1210  1211  1212  1213  1230  1231  1232  1233  1243
           1246  1247  1248  1249  1266  1270  1271  1272  1273  1299  1300  1301  1302  1310  1313
           1314  1315  1316  1330  1331  1332  1333  1348  1349  1350  1351  1377  1378  1379  1380
           1393  1396  1397  1398  1399  1415  1416  1417  1418  1432  1435  1436  1437  1438  1482
           1483  1484  1485  1529  1530  1531  1532  1567  1568  1569  1570  1601  1602  1603  1604
           1634  1635  1636  1637  1670  1671  1672  1673  1717  1718  1719  1720  1777  1778  1779
           1780  1815  1816  1817  1818  1859  1860  1861  1862  1899  1901  1918  1921  1924  1931
           1934  1949  1957  1961  1967  1970  1977  1982  2010  2015  2033  2038  2057  2061  2074
           2080  2109  2188  2245  2322  2389  2391  2406  2436  2437  2438  2441  2448  2463  2492
           2495  2522  2525  2526  2529  2556  2557  2560  2563  2631  2633  2638  2659  2660  2669
           2673  2690  2699  2715  2730  2747  2753  2893  2896
.IFT        1114  2416  2537  2633  2638
.IFTF       1114  2415  2535  2578  2631  2634
.IIF         613   618   623   624   631   632   633   634   637   638   758   842   846  1048  1051
           1057  1058  1060  1061  1100  1919  1925  1926  1937  1949  1953  2185  2392  2393  2394
           2395  2396  2401  2428  2436  2437  2460  2485  2498  2499  2500  2501  2502  2503  2507
           2536  2537  2553  2556  2557  2560  2581  2682  2690  2696  2768  2769  2770  2771  2772
           2774  2776  2777  2778
.IRP        1039  1143  1148  1166  1186  1209  1229  1245  1269  1298  1312  1329  1347  1376  1395
           1414  1434  1481  1528  1566  1600  1633  1669  1716  1776  1814  1858  1924  2193  2194
           2215  2231  2232  2332  2372  2402  2508  2702  2708  2721  2722
.LIST       -1   613   637   748   758   836   843   846  1039  1062  1100  1101  1114  1143  1147
           1166  1170  1186  1190  1209  1213  1229  1233  1245  1249  1269  1273  1298  1302  1312
           1316  1329  1333  1347  1351  1376  1380  1395  1399  1414  1418  1434  1438  1481  1485
           1528  1532  1566  1570  1600  1604  1633  1637  1669  1673  1716  1720  1776  1780  1814
           1818  1858  1862  1925  1941  2436  2502  2659  2760  2768  2769  2770  2771  2772  2773
           2774  2775  2776  2777  2778  2779
.MACRO       1   638   799  1014  1078  2760
.MCALL     613   748   843  1062  1101
.NLIST       1   613   637   748   758   836   843   846  1039  1062  1100  1101  1114  1143  1147
           1166  1170  1186  1190  1209  1213  1229  1233  1245  1249  1269  1273  1298  1302  1312
           1316  1329  1333  1347  1351  1376  1380  1395  1399  1414  1418  1434  1438  1481  1485
           1528  1532  1566  1570  1600  1604  1633  1637  1669  1673  1716  1720  1776  1780  1814
           1818  1858  1862  1925  1941  2436  2502  2659  2760  2768  2769  2770  2771  2772  2773
           2774  2775  2776  2777  2778  2779
.PAGE       766   799   911   961  1014  1039  1143  1953  2104
.REM          1   307
.REPT       758
.SBTTL      627   638   752   761   766   777   799   843   911  1039  1041  1096  1101  1143  1166
           1186  1209  1229  1245  1269  1298  1312  1329  1347  1376  1395  1414  1434  1481  1528
           1566  1600  1633  1669  1716  1776  1814  1858  1915  1953  2104  2106  2185  2242  2319
           2386  2445  2492  2557  2696  2744  2760  2780
.TITLE      613
.WORD       758   759   760   764   774   793   794   795   796   797   798   807   810   811   812
            813   816   817   818   819   820   821   822   825   826   827   848   849   850   851
            852   853   854   855   859   860   861   874   878   881   884   885   886   887   888
            889   890   891   892   893   894   895   896   897   898   899   900   901   902   903
            904   905   906  1930  1933  1948  2135  2182  2217  2318  2471  2476  2731  2733
```

ERRORS DETECTED: 0

```
*,DVDRAA.SEQ/SOL/CRF/NL:TOC=DVDRAA.SML,DVDRAA.P11
RUN-TIME:  48 58 5 SECONDS
CORE USED:  32K
```

JOB

Spooler runtime 11 Seconds, 49 KCS, 325 disk reads, 4 disk writes, 73 pages