# PDP11

KIT11-H EXERCISER

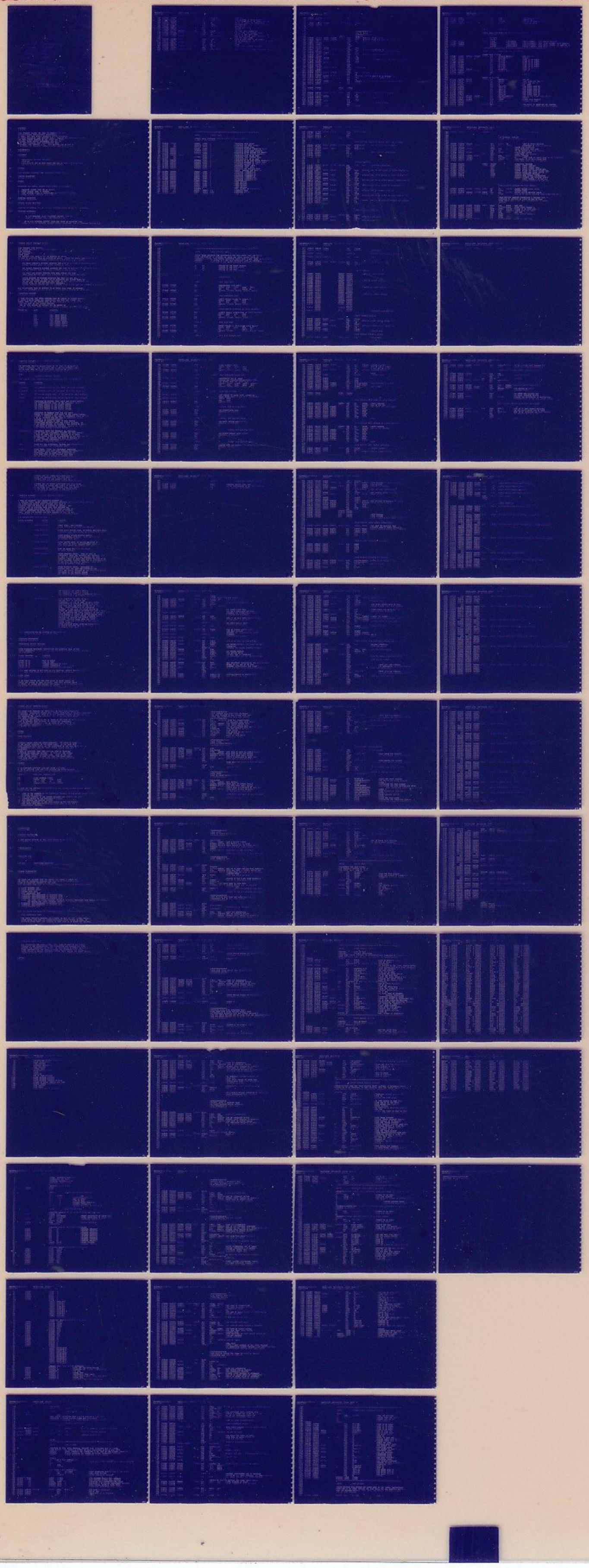
MD-11-DZKHA-B

EP-DZKHA-B-DL

FICHE 1 OF 1

JUN 1978

MADE IN USA



IDENTIFICATION

| | |
|---|---|
| PRODUCT CODE | MAINDEC-11-DZKHA-B-D |
| PRODUCT NAME | KIT11-H EXERCISER |
| DATE CREATED | JANUARY 2, 1974 |
| MAINTAINER | DIAGNOSTIC GROUP |
| AUTHOR | ED BADGER |

A1

```
1.      ABSTRACT
        ********

        THIS PROGRAM ALLOWS THE USER TO CHECKOUT OR DEBUG
        KIT11-H(UNIBUS INPUT/OUTPUT INTERFACE).
        TO TEST, THE USER SIMPLY CONNECTS OUTPUT MODULE(S)
        TO INPUT MODULE(S) (SEE SECTION 4.4).  THROUGH THE SOFTWARE MONITOR,
        THE USER ENTERS ADDRESS OF THE INPUT AND OUTPUT
        MODULES, THEIR VECTOR ADDRESSES, AND HOW THE
        USER HAS CONNECTED THEM TOGETHER(FOR TEST
        PURPOSES) (SEE SECTION 4.5).  THIS PROGRAM CAN BE RUN IF
        A TELETYPE (OR TERMINAL) DOESN'T EXIST (SEE SECTION 4.6).

2.      REQUIREMENTS
        ************

2.1     EQUIPMENT
        ---------

        A.  PDP-11/WITH 4K CORE (OR MORE)
        B.  KIT11-HT
        C.  KIT11-H/WITH ONE OR MORE INPUT AND ONE OR MORE OUTPUT MODULES
            AND ONE OR TWO M7821(OR COMPERABLE MODULE)

2.2     STORAGE
        -------

        THIS PROGRAM OCCUPIES CORE LOCATIONS 000000-12000

3.      LOADING PROCEDURE
        ******* *********

3.1     METHOD
        ------

        PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED:

        1.  ABSOLUTE LOADER MUST BE IN MEMORY.
        2.  PLACE BINARY TAPE IN READER.
        3.  LOAD ADDRESS *7500 (* DETERMINED BY LOCATION OF LOADER)
        4.  PRESS "START" (PROGRAM WILL LOAD).

4.      STARTING PROCEDURE
        ******** *********

4.1     CONTROL SWITCH SETTINGS
        ------- ------ --------

        STARTING AT ADDRESS 200 OR 210 ALL SWITCHES SHOULD BE SET AS INDICATED.

4.2     STARTING ADDRESSES
        -------- ---------

        (A)     IF I/O TERMINAL (I.E. TELETYPE) EXISTS  LOAD AND
                START AT LOCATION 200.  SEE SECTION 4.5.

        (B)    IF NO I/O TERMINAL EXISTS  LOAD AND START AT LOCATION 210.
             NOTE: IF NO I/O TERMINAL EXISTS, FOLLOW PROCEDURE  FOR NO TERMINAL SECTION 4.6.
```

## 4.3 PROGRAM AND/OR OPERATOR ACTION

LOAD PROGRAM INTO MEMORY.
SET SWITCH REGISTER TO STARTING ADDRESS.
LOAD ADDRESS.
SET SWITCHES = 0.
PRESS START
THE PROGRAM WILL TYPE AN "*" IF STARTED AT SA200,
OR HALT AT LOCATION 7212 IF STARTED AT SA210. USING THE FORMAT SPECIFIED IN 4.5
OR 4.6,ENTER IN THE NECESSARY INFORMATION ABOUT KIT11-H AS FOLLOWS.

1) 1ST INPUT MODULE'S ADDRESS (EXAMINE THE M105 IN SLOT B02 TO DETERMINE ADDR).
   INITIAL DEFAULT ADDR IS 164000 (JUMPER 11 CUT).

2) 1ST OUTPUT MODULE'S ADDRESS (EXAMINE THE M105 IN SLOT B03 TO DETERMINE ADDR).
   INITIAL DEFAULT ADDR IS 164010 (JUMPERS 11 AND 3 CUT).

3) HOW INPUT AND OUTPUT MODULES HAVE BEEN CABLED FOR TEST
   (WHAT INPUT MODULES HAVE BEEN CONNECTED TO WHAT OUTPUT MODULES).

4) VECTOR ADDRESS OF MODULES SELECTED FOR TEST (V1 AND V2).
   (EXAMINE THE M7821 IN SLOT F02 TO DETERMINE THE VECTOR ADDR. OF
   THE 1ST TWO INPUT MODULES OR THE M7821 IN SLOT F04 TO DETERMINE THE
   VECTOR ADDR. OF THE 2ND TWO INPUT MODULES).
   INITIAL DEFAULT VECTORS ARE 170 (JUMPERS 7 AND 8 CUT) AND 270 (JUMPERS 6 AND 8 CUT).

ALL INFORMATION MUST BE ENTERED OR AN ERROR WILL OCCUR IF STARTED.
ALSO, ALL ADDRESSES MAY NOT BE ODD, NOR ANY VECTOR ADDRESS BE ABOVE 1000.

## 4.4 CONNECTING MODULES

TO TEST KIT11-H, THE INPUT MODULES MUST BE CABLED TO OUTPUT MODULES.
IF USING A BC08R CABLE FOR CONNECTION, YOU MUST NOT "TWIST" THE
CABLE; THAT IS; THE SMOOTH SIDE OF THE CABLE MUST BE UP ON THE
INPUT AND DOWN ON THE OUTPUT MODULES.
  ONE OR MORE GROUPS OF MODULES MAY BE TESTED AT
ONE TIME. SEE THE CHART BELOW FOR MODULE'S NUMBER, FUNCTION AND SLOT.

| MODULE NO. | SLOT | FUNCTION |
|---|---|---|
| 1 | E01 | 1ST. INPUT MODULE |
| 2 | E02 | 2ND. INPUT MODULE |
| 3 | E03 | 3RD. INPUT MODULE |
| 4 | E04 | 4TH. INPUT MODULE |
| 5 | CD01 | 1ST. OUTPUT MODULE |
| 6 | CD02 | 2ND. OUTPUT MODULE |
| 7 | CD03 | 3RD. OUTPUT MODULE |
| 8 | CD04 | 4TH. OUTPUT MODULE |

C1

## 4.5 DIRECTIVE SUMMARY (IF I/O TERMINAL EXISTS)

--------- -------

THE SOFTWARE MONITOR ALWAYS TYPES AN "*" WHEN IT IS READY TO ACCEPT A COMMAND. THE FOLLOWING ARE A LIST OF COMMANDS THAT CAN BE MADE WHEN IT IS IN THIS MODE; THEY CAN BE ENTERED IN ANY ORDER AND CHANGED AT ANYTIME WHILE IN COMMAND MODE.

NOTE:

"_" INDICATES CARRIAGE RETURN.

ALSO RUBOUT MAYBE TYPED TO DELETE PREVIOUSLY TYPED CHARACTER(S).

| COMMAND | FUNCTION |
|---|---|
| AI:1XXXX0_ | SET ADDRESS 1XXXX0 AS 1ST. ADDR. OF INPUT MODULES(S). |
| AO:1XXXX0_ | SET ADDRESS 1XXXX0 AS 1ST ADDR. OF OUTPUT MODUL(S). |
| V1:XXX_ | SET XXX AS VECTOR ADDR. OF 1ST GROUP OF INPUT MODULES. |
| V2:XXX_ | SET XXX AS VECTOR ADDR. OF 2ND GROUP OF INPUT MODULES. |
| F(ULL)_ | INDICATES TO PROGRAM THAT INPUT AND OUTPUT MODULES ARE CONNECTED(FOR TEST) IN THE FOLLOWING MANNER<br>1ST INPUT MODULE TO 1ST OUTPUT MODULE<br>2ND INPUT MODULE TO 2ND OUTPUT MODULE<br>3RD INPUT MODULE TO 3RD OUTPUT MODULE<br>4TH INPUT MODULE TO 4TH OUTPUT MODULE. |
| 1<6_ | INDICATES TO PROGRAM THAT THE 1ST INPUT MODULE IS CONNECTED (FOR TEST) TO 2ND OUTPUT MODULE. NUMBERS 1 THROUGH 4 ARE USED TO REPRESENT INPUT MODULES 1 TO 4. NUMBERS 5-8 ARE USED TO REPRESENT OUTPUT MODULES 1 TO 4(RESPECTIVELY). THIS MODE OF ENTRY IS USED TO SHOW SINGLE CONNECTIONS BETWEEN INPUT AND OUTPUT MODULES. OR IF AN ERROR OCCURS, TO ISOLATE A BAD MODULE BY MAKING A NEW PAIR OF CONNECTED MODULES. ALWAYS USE THE FORM "INPUT MODULE < OUTPUT MODULE." |
| D(ISCONNECT)_ | DISCONNECT (FROM THE PROGRAM) ALL MODULES. CAN BE USED TO DISCONNECT MODULES AND ONLY CONNECT 1 PAIR OF INPUT AND OUTPUT MODULES IF AN ERROR OCCURS (UNDER TEST) OR IS DESIRABLE TO RUN ONLY ONE PAIR OF MODULES. IF TESTING ONE PAIR OF MODULES AT A TIME, ALWAYS DISCONNECT (FROM PROGRAM) THE PREVIOUSLY CONNECTED PAIR. |
| M(AP)_ | PRINT OUT ALL ADDRESSES, VECTORS AND CONNECTIONS AS THE PROGRAM HAS INTERPETED THEM. |
| S(TART)_ | START TEST. NOTE: ALL NECESSARY ADDRESSES AND VECTORS MUST HAVE BEEN ENTERED FOR CONNECTIONS INDICATED, AND AT LEAST ONE CONNECTION MUST HAVE BEEN MADE OR THE PROGRAM WILL TYPE OUT AN ERROR AND RETURN TO COMMAND MODE. |

D1

| | |
|---|---|
| ^C | CONTROL AND "C" TYPED SIMULTANEOUSLY WILL BRING THE PROGRAM FROM RUN MODE BACK TO COMMAND MODE (IF THE PRESENT TEST IS NOT TESTING INITIALIZATION [RESET INSTRUCTION]). |
| ^R | CONTROL AND "R" TYPED SIMULTANEOUSLY WILL CAUSE THE NUMBER OF PASSES AND NUMBER OF ERRORS (IN OCTAL) TO BE TYPED OUT. THE PROGRAM WILL THEN RETURN TO THE MODE OF OPERATION IT WAS DOING PRIOR TO ^R (EITHER "RUN MODE" OR "COMMAND MODE"). |

## 4.6 DIRECTIVE SUMMARY (IF NO I/O TERMINAL EXSISTS)

A HALT AT LOCATION 7212 INDICATES PROGRAM IS IN COMMAND MODE. ENTER COMMAND IN SWITCH REGISTER BIT 0-3 AND PRESS CONTINUE. ALL DIRECTIVES EXCEPT START COMMAND WILL HALT AT LOCATION 7226 FOR ENTRY OF AN ADDRESS IF NEEDED. WHEN RUNNING WITH NO TERMINAL MAKE SURE THAT PROGRAM HAS HALTED AT THESE LOCATIONS, SINCE IF AN ERROR OCCURED IN ENTERING ADDRESSES AN ERROR HALT WILL OCCUR AT LOCATION 7466. AFTER THE PROGRAM HAS BEEN STARTED AT 210, IT MAY BE RESTARTED AT 1000 AND IT WILL REMAIN IN "NO TERMINAL MODE."

FOR COMMAND MODE HALT AT 7212:

| SWITCH REGISTER | ACTION | FUNCTION |
|---|---|---|
| 0 | PRESS CONTINUE | START TEST. ANY FURTHUR HALTS INDICATES AN ERROR HAS OCCURED. |
| 2 | PRESS CONTINUE | ENTER INPUT MODULE ADDR. IN SWITCH REGISTER PRESS CONTINUE. NEXT HALT SHOULD BE COMMAND MODE HALT. |
| 4 | PRESS CONTINUE | ENTER OUTPUT MODULE ADDRESS SWITCH REGISTER-PRESS CONTINUE. NEXT HALT SHOULD BE COMMAND MODE HALT. |
| 6 | PRESS CONTINUE | ENTER VECTOR ADDR. IN SWITCH REGISTER OF 1ST. GROUP OF INPUT MODULES-PRESS CONTINUE NEXT HALT SHOULD BE COMMAND MODE HALT. |
| 10 | PRESS CONTINUE | SAME AS ABOVE ONLY FOR 2ND GROUP OF INPUT MODULES. |
| 12 | PRESS CONTINUE | PRESS CONTINUE AGAIN. THIS IS USED TO DISCONNECT (FROM THE PROGRAM) ALL MODULES. CAN BE USED TO DISCONNECT ALL MODULES AND ONLY CONNECT 1 PAIR OF INPUT AND OUTPUT MODULES IF AN ERROR OCCURRED (DURING TEST) OR IF IT'S DESIRABLE TO RUN ONLY ONE PAIR OF MODULES. NEXT HALT SHOULD BE COMMAND MODE HALT. |
| 14 | PRESS CONTINUE | PRESS CONTINUE AGAIN. INDICATES TO PROGRAM THAT THE USER HAS CONNECTED THE INPUT AND OUTPUT MODULES IN THE FOLLOWING ORDER: 1ST INPUT TO 1ST OUTPUT MODULE 2ND INPUT TO 2ND OUTPUT MODULE |

E1

```
                                        3RD INPUT TO 3RD OUTPUT MODULE
                                        4TH INPUT TO 4TH OUTPUT MODULE
                                        AND DESIRES TO TEST ALL 4 PAIRS AT ONE TIME,
                                        NEXI HALT SHOULD BE COMMAND MODE HALT

    16          PRESS CONTINUE          THIS DIRECTIVE IS USED TO SHOW
                                        HOW ONE PAIR OF MODULES ARE
                                        CONNECTED,  IT MAY BE USED TO SHOW UNUSAL
                                        CONNECTIONS OR SINGULAR CONNECTIONS FOR
                                        TEST WHEN ONLY ONE PAIR CAN BE RUN AT ONE
                                        TIME,  ENTER CONNECTION IN FOLLOWING
                                        MANNER:  IN SWR BITS 0-2 ENTER BINARY OF
                                        NUMBER OF INPUT MODULE (NUMBERS RUN FROM 1 TO 4
                                        FOR 1ST THROUGH 4TH INPUT MODULE); IN SWR BITS  3-6
                                        ENTER BINARY OF NUMBER OF OUTPUT MODULE THAT
                                        IS CONNECTED TO INPUT MODULE (NUMBERS RUN FROM
                                        5-8 FOR 1ST THROUGH 4TH OUTPUT MODULE).
                                        EXAMPLE: 1000001 WOULD SHOW 1ST INPUT
                                        MODULE (001) WAS CONNECTED TO 4TH OUTPUT
                                        MODULE (1000),
                                        PRESS CONTINUE AFTER ENTERING CONNECTIONS
                                        IN SWITCH REGISTER NEXT HALT SHOULD
                                        BE COMMAND MODE HALT,
```

NOTE: DIRECTIVES CAN BE ENTERED IN ANY ORDER AND
AT ANY TIME WHILE IN COMMAND MODE.

## 5. OPERATING PROCEDURE

### 5,1 OPERATIONAL SWITCH SETTINGS

AFTER ENTERING NECESSARY INFORMATION AND STARTING TEST IN THE
MANNER PRESCRIBED IN 4,5 OR 4,6 THE FOLLOWING SWITCH REGISTER OPTIONS
ARE AVAILABLE:

### 5,1,2

| SWITCH REGISTER | FUNCTION |
|---|---|
| SW15=1 OR UP | HALT ON ERROR |
| SW14=1 OR UP | LOOP ON TEST |
| SW13=1 OR UP | INHIBIT PRINTOUT OF ERROR |
| SW11=1 OR UP | INHIBIT ITERATIONS |
| SW10=1 OR UP | INHIBIT PRINTOUT OF "END PASS" |

NOTE: WHEN PROGRAM IS RUN WITH NO I/O TERMINAL (SA210) PROGRAM WILL
HALT UPON DETECTION OF ERROR WHETHER OR NOT SR15=1.

### 5,1,3 SCOPE LOOPS

IF AN ERROR OCCURS AND THE USER WISHS TO SCOPE ERROR, HE
SHOULD SET SW15=1 TO HALT ON ERROR, THEN WHEN PROGRAM HALTS
ON ERROR, HE SHOULD SET SW15=0, SET SW14=1 (LOOP ON CURRENT TEST),
AND SW13=1 (TO INHIBIT PRINTOUT OF ERROR).

```
5,2      PROGRAM AND/OR OPERATOR ACTION
         ------- ------ -------- ------

         THE FIRST PASS THROUGH THE TESTS WILL BE MADE WITH ITERATIONS
         INHIBITED.  SUCCESSIVE PASSES WILL ENABLE ITERATIONS IF SW11=0.  "END PASS"
         IS PRINTED AT END OF A  PASS IF AN I/O TERMINAL EXISTS. IF ONE DOES NOT
         THE OPERATOR CAN EXAMINE LOCATION 1216 TO SEE HOW MANY PASSES
         HAVE BEEN COMPLETED.
         ^C (CONTROL AND LETTER C) MAY BE TYPED AT ANY TIME TO BRING PROGRAM
         BACK TO COMMAND MODE IN ORDER TO CHANGE ANY PARAMETER.
         ^R (CONTROL AND LETTER R ) MAY BE TYPED AT ANY TIME TO GET
         A RUN SUMMARY CONSISTING OF NUMBER OF PASSES AND NUMBER
         OF ERRORS (IN OCTAL).


6.       ERRORS
         ******

6.1      ERROR PRINTOUT
         ----- --------


         PRINTOUT VARIES WITH THE ERROR DETECTED.  THE ERROR PC TYPED OUT
         IS THE ACTUAL LOCATION OF THE ERROR CALL.  IN THE DATA TEST
         ERROR TYPEOUT,"DATA EXP'D" IS THE COMPLIMENT OF THE DATA SENT, BUT
         IT REPRESENTS WHAT DATA THE INPUT MODULE SHOULD HAVE SENT TO
         PROCESSOR.
         A HALT AT LOCATION 7466 WHEN RUNNING WITH NO TERMINAL
         INDICATES AN ERROR HAS OCCURED.  TO FIND OUT THE NUMBER
         OF THE ERROR, EXAMINE LOCATION 1236.  THIS IS THE ITEM
         NUMBER OF THE ERROR.  TO FIND OUT WHAT THE ERROR TYPOUT
         WOULD HAVE BEEN GOTO TO THE ERROR POINTER TABLE BEGINNING
         AT LOCATION 1306.

6.1.1    EXAMPLE
         -------

         IF WE EXAMINED LOCATION 1236 AND FOUND A 5 (101)
         WE GO TO LOCATION 1306 AND LOOK THROUGH THE ERROR POINTER
         TABLE UNTIL WE FOUND ITEM 5.  THE INFORMATION WOULD LOOK LIKE:

         ;ITEM 5          FROM DUAL ADDRESS TEST

         EM5              ;DUAL ADDRESS ERROR
         DH3              ;ERROR  ADDR    ADDR     ADDR
                          ;PC     OUT     IN       DUAL
         DT3              ;$ERRAD,$TMPO,  $GDADR,  $BDADR
         DF1              ;0

         TO FIND OUT THE INFORMATION SPECIFIED BY DT3 ($ERRAD,$TEMPO,$GDADR,$BDADR)
         FOLLOW THESE STEPS:

         (1)  LOOK UP THE ADDRESS OF THE LABLE(I.E. $ERRAD) IN THE SYMBOL TABLE (WHICH
              FOLLOWS THE LISTING)
         (2)  PUT THIS ADDRESS IN THE SWITCH REGISTER AND DEPRESS THE LOAD
              ADDRESS SWITCH ON THE PROCESSOR'S CONSOUL.
         (3)  NOW DEPRESS THE EXAMINE SWITCH.
         (4)  THE DATA DISPLAYED IN THE DATA LIGHTS IS THE INFORMATION
             THAT WOULD HAVE BEEN PRINTED FOR THIS LABLE IF YOU HAD A
             INPUT/OUTPUT TERMINAL.
```

```
7.      RESTRICTIONS
        ************

7.1     STARTING RESTRICTION
        -------- -----------

        IF THE VECTOR ADDRESS OF ANY INPUT MODULE IS 200 OR 210
        THE PROGRAM MUST BE RESTARTED AT LOCATION 1000



8.0     MISCELLANEOUS
        *************


8.1     EXECUTION TIME
        --------- ----


        0.5 MIN.         ITERATIONS INHIBITED
        5.0 MIN.         WITH ITERATIONS (FOR EACH CONNECTION)



9.0     PROGRAM DISCRIPTION
        ******* ***********



        THE TESTS ARE DIVIDED INTO TWO PARTS: (1) TESTS TO CHECK OUT
        ONE PAIR OF CONNECTED MODULES; AND (2), TESTS TO CHECKOUT ALL
        PAIRS OF CONNECTED MODULES AT ONE TIME.
        PART ONE TAKES EACH PAIR OF CONNECTED MODULES THOUGTH THE FOLLOWING TESTS:

        (A) COUNT PATTERN (UP)
        (B) COUNT PATTERN (DOWN)
        (C) RANDOM DATA TEST
        (D) BYTE OPERATION TEST
        (E) INTERRUPT TEST-PROCESSOR AT PRIORITY ZERO
        (F) INTERRUPT TEST-PROCESSOR AT PRIORITY FOUR
        (G) INTERRUPT TEST-PROCESSOR AT PRIORITY FIVE
            NOTE: THE FIRST TWO INPUT MODULES' PRIORITY IS FIVE, THEREFORE THEY SHOULD NOT INTERRUPT.
                  THE SECOND TWO INPUT MODULES' PRIORITY IS SIX-THEY SHOULD INTERRUPT.
        (H) INTERRUPT TEST-PROCESSOR AT PRIORITY SIX
        (I) INTERRUPT TEST-PROCESSOR AT PRIORITY SEVEN
        (J) INITIALIZATION TEST


        PART TWO CHECKS ALL PAIRS OF CONNECTED MODULES IN THE FOLLOWING TESTS:

        (A) DUAL ADDRESSING TEST

            THE INPUT MODULE'S ADDRESS (IN A PAIR) IS SENT TO ITS' OUTPUT MODULE.
            AFTER DOING THIS FOR ALL CONNECTIONS,EACH INPUT MODULE IS READ. IF
            ANYTHING OTHER THAN ITS' ADDRESS IS READ FROM IT AN ERROR HAS OCCURED.
```

H1

(B) INTERRUPT ORDER TEST

ALL INTERRUPTS ARE FIRST LOCKED OUT. DATA IS SENT TO ALL OUTPUT MODULES IN THE CONNECTIONS. NOW INTERRUPTS ARE ENABLED AND TIME ALLOWED FOR INTERRUPTS TO OCCUR. AS THEY OCCUR, A NUMBER IS PLACED ON A STACK REPRESENTING THE ORDER IN WHICH THE INTERRUPT TOOK PLACE. INPUT MODULE #3 SHOULD INTERRUPT BEFORE INPUT MODULE #4, WHICH SHOULD INTERRUPT BEFORE INPUT MODULE #1. #1 SHOULD INTERRUPT BEFORE INPUT MODULE #2.

LISTING
*******

```

                                 .TITLE  MAINDEC-11-DZKHA-A
                                 ;*COPYRIGHT (C) 1973
                                 ;*DIGITAL EQUIPMENT CORP.
                                 ;*MAYNARD, MASS. 01754
                                 ;*
                                 ;*PROGRAM BY ED BADGER
          000001                 $TN=1
          000000                 $SS=0

                                 .SBTTL          OPERATIONAL SWITCH SETTINGS
                                 ;*
                                 ;*      SWITCH                  USE
                                 ;*      ------          --------------------
                                 ;*        15            HALT ON ERROR
                                 ;*        14            LOOP ON TEST
                                 ;*        13            INHIBIT ERROR TYPEOUTS
                                 ;*        11            INHIBIT ITERATIONS
                                 ;*        10            INHIBIT PRINTOUT OF "END PASS"

                                 .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
          177776                 PS=     177776                  ;PROCESSOR STATUS WORD
                                 .EQUIV  PS,PSW
          177570                 SWR=    177570                  ;SWITCH REGISTER
          177570                 DISPLAY=SWR

                                 ;*GENERAL PURPOSE REGISTER DEFINITIONS
          000000                 R0=     %0                      ;GENERAL REGISTER
          000001                 R1=     %1                      ;GENERAL REGISTER
          000002                 R2=     %2                      ;GENERAL REGISTER
          000003                 R3=     %3                      ;GENERAL REGISTER
          000004                 R4=     %4                      ;GENERAL REGISTER
          000005                 R5=     %5                      ;GENERAL REGISTER
          000006                 R6=     %6                      ;GENERAL REGISTER
          000007                 R7=     %7                      ;GENERAL REGISTER
                                 .EQUIV  R6,SP                   ;STACK POINTER
                                 .EQUIV  R7,PC                   ;PROGRAM COUNTER

                                 ;*"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**
```

L1

```
             000034                  TRAPVEC=34              ;"TRAP" TRAP
                                     .EQUIV  EMT,    ERROR

                                     .SBTTL          TRAP CATCHER

             000000                          .=0
                                     ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTIAN A ".+2,HALT"
                                     ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                     ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS

                                     .SBTTL          STARTING ADDRESS(ES)
             000200                          .=200

     000200  000137  005542                  JMP     @#STAR          ;JUMP TO STARTING ADDRESS OF PROGRAM
             000210                  .=210
     000210  000137  007200                  JMP     NTH             ;GOTO NO TERMINAL HANDLER
             001000                  .=1000
     001000  000137  005542                  JMP     STAR            ;RESTART ADDRESS 1000


             001100                  .=1100
                                     ;;*************************************************************

                                     .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

                                     ;*2) USING A JSR INSTRUCTION
                                     ;*      MOV     PS,-(SP)                        ;PUSH PROCESSOR STATUS WORD ON THE STACK
                                     ;*      JSR     PC,$TYPE                        ;CALL TYPE ROUTINE
                                     ;*      MESADDR                                 ;FIRST ADRESS OF MESSAGE

     001100  177564                  $TPS:   177564                                  ;TTY PRINTER STATUS REG. ADDRESS
     001102  177566                  $TPB:   177566                                  ;TTY PRINTER BUFFER REG. ADDRESS
     001104     000                  $NULL:  .BYTE   0                               ;CONTAINS NULL CHARACTER FOR FILLS
     001105     002                  $FILLS: .BYTE   2                               ;CONTAINS # OF FILLER CHARACTERS REQUIRED
     001106     012                  $FILLC: .BYTE   12                              ;FILL CHARS. AFTER A "LINE FEED"
     001107     000                  $TPFLG: .BYTE   0                               ;"TERMINAL AVAILABLE" FLAG (0=YES)

     001110  105737  001107          $TYPE:  TSTB    $TPFLG                          ;IS THERE A TERMINAL?
     001114  001402                          BEQ     1$                              ;BR IF YES
     001116  000000                          HALT                                    ;HALT HERE IF NO TERMINAL
     001120  000407                          BR      3$                              ;LEAVE
```

M1

```
001122  010046                     1$:     MOV     R0,-(SP)                ;SAVE R0
001124  017600  000002                     MOV     @2(SP),R0               ;GET ADDRESS OF ASCIZ STRING
001130  112046                     2$:     MOVB    (R0)+,-(SP)             ;PUSH CHARACTER TO BE TYPED ONTO STACK
001132  001005                             BNE     4$                      ;BR IF IT ISN'T THE TERMINATOR
001134  005726                             TST     (SP)+                   ;IF TERMINATOR POP IT OFF THE STACK
001136  012600                             MOV     (SP)+,R0                ;RESTORE R0
001140  062716  000002             3$:     ADD     #2,(SP)                 ;ADJUST RETURN PC
001144  000002                             RTI                             ;RETURN
001146  004737  001200             4$:     JSR     PC,7$                   ;GO TYPE THIS CHARACTER
001152  123726  001106             5$:     CMPB    $FILLC,(SP)+            ;IS IT TIME FOR FILLER CHARS.?
001156  001364                             BNE     2$                      ;IF NO GO GET NEXT CHAR.
001160  013746  001104                     MOV     $NULL,-(SP)             ;GET # OF FILLER CHARS. NEEDED
                                                                           ;AND THE NULL CHAR.
001164  105366  000001             6$:     DECB    1(SP)                   ;DOES A NULL NEED TO BE TYPED?
001170  002770                             BLT     5$                      ;BR IF NO--GO POP THE NULL OFF OF STACK
001172  004737  001200                     JSR     PC,7$                   ;GO TYPE A NULL
001176  000772                             BR      6$                      ;LOOP
001200  105777  177674             7$:     TSTB    @$TPS                   ;WAIT UNTIL PRINTER IS READY
001204  100375                             BPL     7$
001206  116677  000002  177666             MOVB    2(SP),@$TPB             ;LOAD CHAR TO BE TYPED INTO DATA REG.
001214  000207                             RTS     PC
```

A2

```
                                ;;*****************************************************************

                                .SBTTL          COMMON TAGS

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

001216  000000                  $PASS:  .WORD   0                       ;CONTAINS PASS COUNT
001220     000                  $TSTNM: .BYTE   0                       ;CONTAINS THE TEST NUMBER
001221     000                  $ERFLG: .BYTE   0                       ;CONTAINS ERROR FLAG
001222  000000                  $ICNT:  .WORD   0                       ;CONTAINS SUBTEST ITERATION COUNT
001224  000000                  $LPADR: .WORD   0                       ;CONTAINS SCOPE LOOP ADDRESS
001226  000000                  $LPERR: .WORD   0                       ;CONTAINS SCOPE RETURN FOR ERRORS
001230  000000                  $ERTTL: .WORD   0                       ;CONTAINS TOTAL ERRORS DETECTED
001232  000000  000000                  .WORD   0,0                     ;RESERVED--NOT TO BE USED
001236     000                  $ITEMB: .BYTE   0                       ;CONTAINS ITEM CONTROL BYTE
001237     000                          .BYTE   0                       ;RESERVED--NOT TO BE USED
001240  000000                  $ERRAD: .WORD   0                       ;CONTAINS PC OF LAST ERROR INSTRUCTION
001242  000000                  $GDADR: .WORD   0                       ;CONTAINS ADDRESS OF 'GOOD' DATA
001244  000000                  $BDADR: .WORD   0                       ;CONTAINS ADDRESS OF 'BAD' DATA
001246  000000                  $GDDAT: .WORD   0                       ;CONTAINS 'GOOD' DATA
001250  000000                  $BDDAT: .WORD   0                       ;CONTAINS 'BAD' DATA
001252  000000                  $REGAD: .WORD   0                       ;CONTAINS THE ADDRESS FROM
                                                                        ;WHICH  ($REG0) WAS OBTAINED
001254  000000                  $REG0:  .WORD   0                       ;CONTAINS (($REGAD)+0)
001256  000000                  $REG1:  .WORD   0                       ;CONTAINS (($REGAD)+2)
001260  000000                  $REG2:  .WORD   0                       ;CONTAINS (($REGAD)+4)
001262  000000                  $REG3:  .WORD   0                       ;CONTAINS (($REGAD)+6)
001264  000000                  $REG4:  .WORD   0                       ;CONTAINS (($REGAD)+10)
001266  000000                  $REG5:  .WORD   0                       ;CONTAINS (($REGAD)+12)
001270  000000                  $TMP0:  .WORD   0                       ;USER DEFINED
001272  000000                  $TMP1:  .WORD   0                       ;USER DEFINED
001274  000000                  $TMP2:  .WORD   0                       ;USER DEFINED
001276  000000                  $TMP3:  .WORD   0                       ;USER DEFINED
001300  000000                  $TIMES: 0                               ;MAX. NUMBER OF ITERATIONS
001302     077                  $QUES:  .ASCII  /?/                     ;QUESTION MARK
001303     015                  $CRLF:  .ASCII  <15>                    ;CARRIAGE RETURN
001304  000012                  $LF:    .ASCIZ  <12>                    ;LINE FEED
```

B1

```
                                 ;;*************************************************************

                                 .SBTTL          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 $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRAD).
                                 ;*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


 001306                          $ERRTB:
                                                         ;ERROR-TABLE


                                 ;ITEM 1                  FROM DATA TEST

 001306  007606                  EM1                     ;SND-RECIEVE DATA ERROR
 001310  010476                  DH1                     ;ERROR  ADDR    ADDR    DATA    DATA
                                                         ;PC     OUT     IN      EXP'D   IN
 001312  007524                  DT1                     ;$ERRAD, $GDADR, $BDADR, $GDDAT, $BDDAT
 001314  000000                  DF1                     ;0

                                 ;ITEM 2                  FROM INTERRUPT TEST

 001316  007640                  EM2                     ;INPUT MODULE FAILED TO INTERRUPT
 001320  010611                  DH2                     ;ERROR  ADDR    ADDR    PROS
                                                         ;PC     OUT     IN      STAT
 001322  007540                  DT2                     ;$ERRAD, $GDADR, $BDADR, $TMPO
 001324  000000                  DF1                     ;0

                                 ;ITEM 3                 ;INPUT MODULE INTERRUPT AT WRONG PRIORITY

 001326  007703                  EM3                     ;;INPUT MODULE INTERRUPTED AT WRONG PRIORITY
 001330  010611                  DH2                     ;ERROR  ADDR    ADDR    PROS
                                                         ;PC     OUT     IN      STAT
 001332  007540                  DT2                     ;$ERRAD, $GDADR, $BDADR, $TMPO
 001334  000000                  DF1                     ;0

                                 ;ITEM 4                  FROM INIT TEST

 001336  007760                  EM4                     ;RESET FAILED TO INITIALIZE INPUT MODULE
 001340  010476                  DH1                     ;ERROR  ADDR    ADDR    DATA    DATA
                                                         ;PC     OUT     IN      EXP'D   IN
 001342  007524                  DT1                     ;$ERRAD, $GDADR,         $BDADR, $GDDAT, $BDDAT
 001344  000000                  DF1                     ;0

                                 ;ITEM 5                  FROM DUAL ADDRESS TEST
```

C2

```
001346  010041                  EM5                             ;DUAL ADDRESS ERROR
001350  010706                  DH3                             ;ERROR  ADDR    ADDR    ADDR
                                                                ;PC     OUT     IN      DUAL
001352  007552                  DT3                             ;$ERRAD, $TMPO, $GDADR, $BDADR
001354  000000                  DF1                             ;0

                                ;ITEM 6         ERITR2  FROM INTERRUPT ORDER TEST

001356  010066                  EM6                             ;INTERRUPTS OUT OF ORDER
                                                                ;SHOULD BE: INTER3>INTER4>INTER1>INTER2
                                                                ;IF ADDR=0 THAN NOT UNDER TEST
001360  011003                  DH4                             ;ERROR  INTER1  INTER2  INTER3  INTER4
                                                                ;PC     ADDR    ADDR    ADDR    ADDR
001362  007564                  DT4                             ;$ERRAD, $REGO, $REG1, $REG2, $REG3
001364  000000                  DF1

                                ;ITEM 7         ERCSR   ;FROM CSR INT TEST

001366  010171                  EM7                             ;INT FAILED TO CLEAR INTR, ENABLE BIT
                                                                ;ADDR SHOWS INPUT MODULE THAT INTERRUPTED
001370  011102                  DH5                             ;ERROR  ADDR
                                                                ;PC     INTR
001372  007600                  DT5                             ;$ERRAD $TMPO
001374  000000                  DF1                             ;0

                                ;ITEM 10                         CONTROL TEST MONITOR ERROR

001376  010247                  EM11                            ;NO CONNECTIONS MADE
001400  011137                  DH7                             ;PROGRAM NOT RUNNING
001402  000000                  DT6                             ;0
001404  000000                  DF1                             ;0

                                ;ITEM 11                         CONTROL TEST MONITOR ERROR

001406  010275                  EM12                            ;NO INPUT MODULE ADDR ENTERED
001410  011137                  DH7                             ;PROGRAM NOT RUNNING
001412  000000                  DT6                             ;0
001414  000000                  DF1                             ;0

                                ;ITEM 12                                 CONTROL TEST MONITOR ERROR

001416  010335                  EM13                            ;NO OUTPUT MODULE ADDR ENTERED
001420  011137                  DH7                             ;PROGRAM NOT RUNNING
001422  000000                  DT6                             ;0
001424  000000                  DF1                             ;0

                                ;ITEM 13                                 CONTROL TEST MONITOR ERROR

001426  010376                  EM14                            ;VECTOR ADDR NOT ENTERED FOR SELECTED CONNECTION
001430  011137                  DH7                             ;PROGRAM NOT RUNNING
001432  000000                  DT6                             ;0
001434  000000                  DF1                             ;0


```

D2

```
   330                                                     ;ITEM 14                CONTROL TEST MONITOR FAILURE
   331
   332  001436  011176                                     MCRLF          ;CARRAGE RETURN LINE FEED
   333  001440  011536                                     UNKINW         ;WRONG INPUT-RETYP IT SO THAT OPERATOR
   334  001442  000000                                     DT6            ; 0  CAN SEE HIS MISTAKE
   335  001444  000000                                     DF1            ; 0
   336
```

F2

```
                                  .SBTTL         TESTS
                                                 ;START UP ROUTINE FOR SUBTESTS
   001446  012706  001100         START:  MOV    #1100,  SP       ;SET UP SP
   001452  012701  004270                 MOV    #MOD5C, R1
   001456  005721                 1$:     TST    (1)+             ;SEE IF ANY CONNECTIONS
   001460  001006                         BNE    START1
   001462  020127  004300                 CMP    R1,     #MOD8C+2
   001466  001373                         BNE    1$
                                                                  ;NO CONNECTIONS MADE
                                                                  ;WE CAN'T START-LET HIM KNOW
   001470  104010                         ERROR 10                ;MESSAGE "NO CONNECTIONS ;MADE"
                                                                  ;"PROGRAM NOT RUNNING"
   001472  000137  005542                 JMP    STAR
   001476  005737  004240         START1: TST    MOD1A            ;SEE IF WE HAVE INPUT MODULE ADDRS,
   001502  001003                         BNE    START2           ;CAN'T START IF WE DON'T

   001504  104011                         ERROR 11                ;NO INPUT MODULE ADDRS
                                                                  ;PROGRAM NOT RUNNING
   001506  000137  005542                 JMP    STAR

   001512  005737  004250         START2: TST    MOD5A            ;SEE IF OUTPUT MODULE ADDR.
   001516  001003                         BNE    START3           ;HAS BEEN ENTERED
                                                                  ;MESSAGE "NO OUTPUT MODULE ADDRS
   001520  104012                         ERROR 12                ;ENTERED"
                                                                  ;"PROGRAM NOT RUNNING"
   001522  000137  005542                 JMP    STAR

   001526  005737  004260         START3: TST    MOD1V            ;SEE IF WE HAVE VECTORS ENTERED
   001532  001006                         BNE    START4
   001534  005737  004270                 TST    MOD5C            ;NO VECTOR ENTERED, BUT MIGHT NOT BE
   001540  001403                         BEQ    START4           ;CONNECTED
   001542  104013                         ERROR 13                ;MESSAGE "NO VECTOR ADDRESS ENTERED"
   001544  000137  005542                 JMP    STAR
   001550  005737  004264         START4: TST    MOD3V            ;IS VECTOR ENTERED
   001554  001006                         BNE    START5           ;FOR SECOND GROUP
   001556  005737  004274                 TST    MOD7C            ;IF NOT, SEE IF CONNECTED
   001562  001403                         BEQ    START5           ;IS CONNECTED, BUT NOT VECTOR ENTERED
   001564  104013                         ERROR  13
   001566  000137  005542                 JMP    STAR

   001572  005037  004330         START5: CLR    CFLG             ;ALL NECESSARY INFORMATION HAS BEEN ENTERED
   001576  005737  001216                 TST    $PASS            ;READY TO GO, SEE IF WE TYPE
   001602  001005                         BNE    1$               ;MESSAGE RUNNING-ONLY ONE FIRST
   001604  105737  001107                 TSTB   $TPFLG           ;PASS AND ONLY IF WE HAVE A TERMINAL
   001610  001002                         BNE    1$
   001612  104400                         TYPE
   001614  011406                         MRUN
   001616  012703  004260         1$:     MOV    #MOD1V, R3       ;INITIALIZATION OF CONNECTIONS
   001622  012705  004270                 MOV    #MOD5C, R5       ;AND VECTORS FOR START
   001626  012704  004250                 MOV    #MOD5A, R4
```

F2

```
                                                ;*****************
                                                ;THIS HANDLER WILL FORM TEST ADDRESSES
                                                ;AND SEND PROGRAM TO ALL SINGLE LINE TESTS
                                                ;UNTIL ALL CONNECTIONS HAVE BEEN TESTED
                                                ;*****************
001632  020527  004300          SINGLE: CMP     R5,     #MOD8C+2 ;DONE ALL CONNECTIONS?
001636  001417                          BEQ     SINGLF          ;IF YES,GOTO DUAL ADDR TESTING
001640  012437  001242                  MOV     (4)+,   $GDADR  ;GET OUTPUT MODULE ADDR
001644  013537  001244                  MOV     @(5)+,  $BDADR  ;GET INPUT MODULE ADDR
001650  001770                          BEQ     SINGLE          ;IF NO CONNECTION THEN GET NEW ADDR,
001652  013700  001244                  MOV     $BDADR,R0       ;NOW GET VECTOR ADDRESS !
001656  042700  177770                  BIC     #177770,R0      ;FORM OFFSET
001662  062700  004260                  ADD     #MOD1V,R0       ;USE OFFSET TO GET VECTOR
001666  011037  004300                  MOV     (0),VECTOR      ;STORE VECTOR
001672  000137  001702                  JMP     DATASC          ;GOTO TO DATA TESTS
001676  000137  003064          SINGLF: JMP     DUAL1           ;GOTO DUAL TESTS

                                                ;*******************
                                                ;DATA TESTS
                                                ;PART 1: COUNT PATTERN (UP)
                                                ;*******************

001702  005037  004500          DATASC: CLR     NINP
001706  012737  001706  001224  1$:     MOV     #1$,$LPADR
001714  013737  004500  001246          MOV     NINP,   $GDDAT  ;GET DATA TO SEND TO OUTPUT MODULE
001722  013777  001246  177312          MOV     $GDDAT,@$GDADR ;SEND DATA TO OUT PUT MODULE
001730  017737  177310  001250          MOV     @$BDADR, $BDDAT ;GET DATA FROM INPUT MODULE
001736  005137  001246                  COM     $GDDAT          ;EXP'D = COMPLIEMT OF SENT DATA
001742  023737  001246  001250          CMP     $GDDAT, $BDDAT  ;DATA SENT = DATA RECIEVED?
001750  001401                          BEQ     .+4
001752  104001                          ERROR 1                 ;DATA SENT NOT EQUAL TO DATA RECIEVED
                                                                ;ITERATE
001754  000004                          SCOPE
001756  005237  004500                  INC     NINP            ;COUT UP FULL
001762  001351                          BNE     1$

                                                ;*****************
                                                ;DATA TESTS
                                                ;PART 2: COUNT PATTERN (DOWN)
                                                ;*****************

001764  012737  001772  001224  DATA1:  MOV     #1$,$LPADR
001772  013737  004500  001246  1$:     MOV     NINP,   $GDDAT  ;GET PATTERN
002000  013777  001246  177234          MOV     $GDDAT, @$GDADR ;SEND DATA TO OUTPUT MODULE
002006  017737  177232  001250          MOV     @$BDADR,$BDDAT  ;GET DATA FROM INPUT MODULE
002014  005137  001246                  COM     $GDDAT          ;EXP'D DATA = COMPLIEMT OF DATA SENT
002020  023737  001246  001250          CMP     $GDDAT, $BDDAT  ;DATA EXP'D = DATA RECIEVED ?
002026  001401                          BEQ     .+4             ;IF YES GET NEW DATA WORD

002030  104001                          ERROR 1                 ;DATA EXP'D NOT EQUAL TO DATA RECIEVED
002032  000004                          SCOPE
002034  005337  004500                  DEC     NINP            ;COUNT DOWN FULL
002040  001351                          BNE     DATA1
```

```


                                                ;****************
                                                ;DATA TESTS
                                                ;PART 3: RANDOM NUMBER TEST
                                                ;****************


002042  012737  002050  001224  DATAR:  MOV     #1$,$LPADR
002050  004737  006664          1$:     JSR     PC,     $RAND   ;GET A RANDOM NUMBER
002054  013737  007012  001246          MOV     $LONUM, $GDDAT  ;PUT RANDOM NO. IN GDDAT
002062  013777  001246  177152          MOV     $GDDAT, @$GDADR ;SEND RANDOM NO. TO OUTPUT MODULE
002070  017737  177150  001250          MOV     @$BDADR, $BDDAT ;GET DATA FROM INPUT MODULE
002076  005137  001246                  COM     $GDDAT
002102  023737  001246  001250          CMP     $GDDAT, $BDDAT  ;DATA SENT = DATA RECEIVED?
002110  001401                          BEQ     .+4
002112  104001                          ERROR 1                 ;DATA SENT NOT EQUAL TO DATA RECIEVED

                                                ;****************
                                                ;BYTE OPERATION TEST
                                                ;****************


002114  000004                  DATAO:  SCOPE
002116  005077  177120                  CLR     @$GDADR
002122  012737  177400  001246          MOV     #177400,$GDDAT  ;SET UP TO TEST FOR LOW BYTE OPERATION
002130  112777  177777  177104          MOVB    #-1,    @$GDADR ;SEND ALL ONES TO OUTPUT MODULE BUT
002136  017737  177102  001250          MOV     @$BDADR, $BDDAT ;EXPECT ONLY LOW BYTE TO GET THROUGH
                                                                ;JUSTIFY DATA
002144  105737  001250                  TSTB    $BDDAT          ;TEST FOR ZEROS LOW BYTE
002150  001401                          BEQ     .+4
002152  104001                          ERROR 1                 ;FAILED TO DUE A LOW BYTE OPERATION
002154  005137  001246                  COM     $GDDAT          ;SET UP TO TEST HIGH BYTE OPERATION
002160  013700  001242                  MOV     $GDADR,R0
002164  112760  177777  000001          MOVB    #-1,    1(0) ;SEND ONES TO HIGH BYTE
002172  017737  177046  001250          MOV     @$BDADR, $BDDAT ;EXPECT ZERO'S BACK IN LOW BYTE
                                                                ;JUSTIFY DATA
002200  105737  001251                  TSTB    $BDDAT+1                ;TEST FOR ZEROS IN HIGH BYTE
002204  001401                          BEQ     .+4
002206  104001                          ERROR 1                 ;FAILED TO DUE A HIGH BYTE OPERATION

                                                ;****************
                                                ;THIS ROUTINE MAKE SURE THE INPUT MODULES WILL
                                                ;INTERRUPT, AND TO THE RIGHT VECTOR
                                                ;****************

002210  000004                          SCOPE
002212  012737  002230  001224          MOV     #SINT,$LPADR
002220  000005                          RESET                   ;INITIALIZE ALL MODULES
002222  052777  000100  001470          BIS     #100,@$TKS
002230  000004                  SINT:   SCOPE
002232  012737  000340  177776          MOV     #340,   PSW     ;LOCK OUT INTERRUPTS
002240  012777  002310  002032          MOV     #SINTR, @VECTOR ;SET UP INTERUPT RETURN
002246  013777  005034  001764          MOV     $17,    @MOD1A  ;ENABLE INPUT MODULE TO INTERRUPT
002254  012777  000001  176760          MOV     #1,     @$GDADR ;SEND DATA TO OUTPUT MODULE
```

H2

```
002262  005037  177776                    CLR     PSW              ;ALLOW INTERRUPTS
002266  005000                            CLR     R0
002270  005200                            INC     R0               ;WAIT HERE FOR INTERRUPT
002272  001376                            BNE     .-2
002274  013737  177776  001270            MOV     PSW,    $TMP0
002302  104002                            ERROR 2                  ;INPUT MODULE FAILED TO INTERRUPT AT
                                                                   ;PROCESSOR PRIORITY ZERO
002304  000137  002230                    JMP     SINT
002310  022626                    SINTR:  POPSP2                   ;RETURN SP
002312  017737  176726  001270            MOV     @$BDADR,         $TMP0




                                                  ;****************
                                                  ;MAKE SURE INPUT MODULE WILL INTERRUPT WITH
                                                  ;PROCESSOR PRIORITY AT LEVEL 4
                                                  ;****************


002320  000004                    SINT4:  SCOPE
002322  012737  000340  177776            MOV     #340,   PSW      ;LOCK OUT INTERRUPTS
002330  012777  002402  001742            MOV     #SINT4R, @VECTOR ;SET UP INTERRUPT RETURN
002336  013777  005034  001674            MOV     $17,    @MOD1A   ;ENABLE INPUT MODULES TO INTR.
002344  012777  000001  176670            MOV     #1,     @$GDADR  ;SEND DATA TO OUTPUT MODULE
002352  012737  000200  177776            MOV     #200,   PSW      ;SET PRIORITY TO LEVAL FOUR
002360  005000                            CLR     R0
002362  005200                            INC     R0               ;WAIT HERE FOR INTERRUPT
002364  001376                            BNE     .-2
002366  012737  000200  001270            MOV     #200,   $TMP0
002374  104002                            ERROR 2                  ;INPUT MODULE FAILED TO INTERRUPT AT
                                                                   ;PROCESSOR PRIORITY FOUR
002376  000137  002412                    JMP     SINT5

002402  022626                    SINT4R: POPSP2                   ;RESET SP
002404  017737  176634  001270            MOV     @$BDADR,         $TMP0


                                                  ;****************
                                                  ;PRIORITY LEVEL FIVE INTERRUPT TEST
                                                  ;NOTE THAT SOME INPUT MODULES SHOULD INTR
                                                  ;AT PROCESSOR PRIORITY FIVE AND OTHERS SHOULDN'T
                                                  ;1ST TWO INPUT MODULES ARE AT LEVEL FIVE AND SHOULDN'T INTR.
                                                  ;2ND TWO INPUT MODULES ARE AT LEVEL SIX AND SHOULD INTR
                                                  ;****************


002412  000004                    SINT5:  SCOPE
002414  005037  004302                    CLR     SFIVE            ;DETERMINE IF CURRENT INPUT
002420  023737  004240  001244            CMP     MOD1A,  $BDADR   ;MODULE SHOULD INTR.
002426  001406                            BEQ     1$
002430  023737  004242  001244            CMP     MOD2A,  $BDADR
002436  001402                            BEQ     1$
002440  005137  004302                    COM     SFIVE            ;2ND GROUP OF INPUT MODULES
```

```
002444  012737  000240  001270  1$:     MOV     #240,   $TMP0
002452  012737  000340  177776          MOV     #340,   PSW     ;LOCK OUT INTERRUPTS
002460  012777  002532  001612          MOV     #$INT5R, @VECTOR ;SET UP INTERRUPT RETURN
002466  013777  005034  001544          MOV     $17,    @MOD1A  ;ENABLE INPUT MODULES TO INTERRUPT
002474  012777  000001  176540          MOV     #1,     @$GDADR ;SEND DATA TO OUTPUT MODULE
002502  012737  000240  177776          MOV     #240,   PSW     ;SET PROCESSOR PRIORITY TO LEVEL FIVE
002510  005000                          CLR     R0              ;WAIT HERE FOR AN INTERRUPT
002512  005200                          INC     R0
002514  001376                          BNE     .-2
002516  005737  004302                  TST     $FIVE           ;NO INTERRUPT OCCURED - SHOULD WE
002522  001413                          BEQ     $INT6           ;HAVE ONE?
002524  104002                          ERROR 2                 ;YES, INPUT MODULE AT LEVEL SIX
                                                                ;SHOULD HAVE INTR, WITH PROCESSOR
002526  000137  002552                  JMP     $INT6           ;PRIORITY AT LEVEL 5

002532  022626                  $INT5R: POPSP2                  ;INTERRUPTED - BUT SHOULD WE HAVE?
002534  017737  176504  001270          MOV     @$BDADR,        $TMP0
002542  005737  004302                  TST     $FIVE
002546  001001                          BNE     $INT6
002550  104003                          ERROR 3                 ;NO - $INPUT MODULE (PRIORITY 5) SHOULD NOT
                                                                ;HAVE INTERRUPTED WITH PROCESSOR
                                                                ;PRIORITY AT LEVEL 5


                                                ;****************
                                                ;PRIORTY LEVEL 6 INTERUPT TEST
                                                ;NO INPUT MODULES SHOULD INTERRUPT
                                                ;WITH PROCESSOR AT THIS LEVEL
                                                ;****************


002552  000004                  $INT6:  SCOPE
002554  012737  000340  177776          MOV     #340,   PSW     ;LOCK OUT INTERRUPTS
002562  012737  000300  001270          MOV     #300,   $TMP0
002570  012777  002310  001502          MOV     #$INTR, @VECTOR ;SET UP INTERRUPT RETURN
002576  013777  005034  001434          MOV     $17,    @MOD1A  ;ENABLE INPUT MODULES TO INTERRUPT
002604  012777  000001  176430          MOV     #1,     @$GDADR ;SEND DATA TO OUTPUT MODULE
002612  013737  001270  177776          MOV     $TMP0,  PSW     ;SET PROCESSOR PRIORITY TO LEVEL 6
002620  005000                          CLR     R0              ;WAIT HERE FOR ANY INTERRUPT
002622  005200                          INC     R0
002624  100376                          BPL     .-2
                                                                ;NO
002626  000137  002644                  JMP     $INT7
                                                ;INTERRUPT SERVICE ROUTINE
002632  022626                  $INT6R: POPSP2                  ;RESET SP
002634  017737  176404  001270          MOV     @$BDADR,        $TMP0
002642  104003                          ERROR 3
```

```



                                                        ;*****************
                                                        ;PRIORTY LEVEL 7 INTERRUPT TEST
                                                        ;NO INPUT MODULES SHOULD INTERRUPT
                                                        ;WITH PROCESSOR AT THIS LEVEL
                                                        ;*****************


 002644  000004                           SINT7:  SCOPE
 002646  012737  000340  177776                   MOV     #340,   PSW     ;LOCK OUT INTERRUPTS
 002654  012737  000340  001270                   MOV     #340,   $TMP0
 002662  012777  002310  001410                   MOV     #SINTR, @VECTOR ;SET UP INTERRUPT RETURN
 002670  013777  005034  001342                   MOV     S17,    @MOD1A  ;ENABLE INPUT MODULES TO INTERRUPT
 002676  012777  000001  176336                   MOV     #1,     @$GDADR ;SEND DATA TO OUTPUT MODULE
 002704  005000                                   CLR     R0              ;WAIT HERE FOR ANY INTERRUPT
 002706  005200                                   INC     R0
 002710  100376                                   BPL     .-2
                                                                          ;NO
 002712  000137  002730                           JMP     INIT
 002716  022626                           SINT7R: POPSP2                  ;RESET SP
 002720  017737  176320  001270                   MOV     @$BDADR,$TMP0
 002726  104003                                   ERROR 3

                                                        ;*****************
                                                        ;INITIALIZATION TEST FOR MODULES
                                                        ;*****************


 002730  012737  000340  177776           INIT:   MOV     #340,   PSW     ;LOCK OUT INTERRUPTS
 002736  012777  003044  001334                   MOV     #INITR, @VECTOR ;SET UP FOR POSSIBLE INTERRUPT
 002744  012737  177777  001246                   MOV     #-1,    $GDDAT  ;SHOULD GET ALL ONES BACK AFTER INITIALIZE
 002752  013777  005034  001260                   MOV     S17,    @MOD1A  ;ENABLE INPUT MODULE TO INTERRUPT
 002760  012777  052525  176254                   MOV     #52525, @$GDADR ;SEND DATA TO OUTPUT MODULE
 002766  000005                                   RESET                   ;SYSTEM INITIALIZE
 002770  052777  000100  000722                   BIS     #100,@$TKS
 002776  017737  176242  001250                   MOV     @$BDADR, $BDDAT ;GET DATA FROM INPUT MODULE
 003004  023737  001246  001250                   CMP     $GDDAT, $BDDAT  ;WAS DATA ALL ONES?
 003012  001403                                   BEQ     1$
 003014  104004                                   ERROR 4                 ;INIT FAILED TO INIALIZE MODULES
 003016  000137  002730                           JMP     INIT

 003022  005037  177776                   1$:     CLR     PSW             ;ALLOW INTERRUPTS, SEE IF INPUT
 003026  005000                                   CLR     R0              ;MODULE INTERRUPTS - IT SHOUDN'T
 003030  105200                                   INCB    R0              ;INTIALIZE SHOULD HAVE CLEARED
 003032  100376                                   BPL     .-2             ;INTERRUPT ENABLE BITS
 003034  010177  001240                           MOV     R1,     @VECTOR
 003040  000137  001632                           JMP     SINGLE          ;GO BACK AND PICK-UP NEW CONNECTION
                                                          ;INTERRUPT HANDLE
 003044  022626                           INITR:  POPSP2                  ;RESET SP
 003046  017737  176172  001270                   MOV     @$BDADR,$TMP0
 003054  104007                                   ERROR 7                 ;RESET (SYSTEM INITIALIZE) FAILED
 003056  000004                                   SCOPE                   ;TO CLEAR INTERRUPT ENABLE BIT IN
 003060  000137  001632                           JMP SINGLE              ;CSR (FOR THIS INPUT MODULE)
```

K2

```



                                                              ;****************
                                                              ;DUAL ADDRESS TEST FOR ALL
                                                              ;CONNECTED MODULES
                                                              ;****************


       003064  012737  003072  001224  DUAL1:  MOV     #DUAL,  $LPADR
       003072  012701  004270          DUAL:   MOV     #MOD5C, R1         ;GET LIST OF CONNECTIONS
       003076  012702  004250                  MOV     #MOD5A, R2         ;GET LIST OF OUTPUT MODULES
       003102  005711                  1$:     TST     (1)                ;CONNECTION EXIT?
       003104  001403                          BEQ     2$                 ;NO
       003106  017172  000000  000000          MOV     @(1),   @(2)       ;MOV ADR OF INPUT MODULE TO OUTPUT MODULE
       003114  005722                  2$:     TST     (2)+               ;UPDATE POINTERS
       003116  005721                          TST     (1)+
       003120  020127  004300                  CMP     R1,#MOD8C+2        ;CHECK FOR END OF LIST
       003124  001366                          BNE     1$

       003126  012701  004270          DUALT:  MOV     #MOD5C, R1         ;GET POINTER LIST AGAIN
       003132  012702  004250                  MOV     #MOD5A, R2
       003136  017137  000000  001242  1$:     MOV     @(1),   $GDADR     ;GET ADDR OF INPUT MODULE IF EXISTANT
       003144  001413                          BEQ     2$
       003146  011237  001270                  MOV     (2),    $TMPO      ;GET ADDR OF OUTPUT MODULE
       003152  017737  176064  001244          MOV     @$GDADR,$BDADR     ;GET DATA FROM INPUT MODULE
       003160  005137  001244                  COM     $BDADR             ;JUSTIFY DATA
       003164  023737  001242  001244          CMP     $GDADR, $BDADR     ;DATA RECIEVED IN INPUT MODULE SHOULD BE
       003172  001007                          BNE     3$                 ;ITS OWN ADDRESS
       003174  005721                  2$:     TST     (1)+               ;UPDATE POINTERS
       003176  005722                          TST     (2)+
       003200  020127  004300                  CMP     R1,     #MOD8C+2   ;END OF LIST?
       003204  001354                          BNE     1$
       003206  000137  003214                  JMP     AINT               ;YES, EXIT
       003212  104005                  3$:     ERROR 5                    ;ERROR - SENT ADDRESS OF ALL INPUT MODULES
                                                                          ;TO RESPECTIVE OUTPUT MODULES - BUT DIDN'T RECIEVE
                                                                          ;CORRESPONDING ADDRESS FROM INPUT MODULE

                                                              ;****************
                                                              ;THIS ROUTINE CHECKS THE ORDER IN WHICH WE RECIEVE
                                                              ;INTERRUPTS BACK FROM ALL MODULES
                                                              ;****************

       003214  000004                  AINT:   SCOPE
       003216  012704  004304                  MOV     #INTPO,R4
       003222  005024                  10$:    CLR     (4)+
       003224  020427  004314                  CMP     R4,     #INTPO+10
       003230  001374                          BNE     10$
       003232  012737  000340  177776          MOV     #340,   PSW        ;LOCK OUT INTERRUPTS
       003240  012702  004250                  MOV     #MOD5A, R2         ;GET OUTPUT MODULE POINTER
       003244  012703  004270                  MOV     #MOD5C, R3         ;GET CONNECTION POINTER
       003250  013777  005034  000762          MOV     $17,    @MOD1A     ;ENABLE INPUT MODULES TO INTERRUPT
       003256  012704  004304                  MOV     #INTPO, R4         ;SET UP TO STORE ORDER OF INTERRUPTS
       003262  012705  004316                  MOV     #INTPO1, R5        ;SETUP TO STORE VALUE OF INTERRUPT
       003266  013700  004260                  MOV     MOD1V,RO           ;SET UP INTERUPT SERVICE ROUTINES
       003272  012720  003476                  MOV     #INTRS1,(0)+
003276  012720  000340                    MOV     #340,    (0)+      ;LOCK OUT INTERRUPTS WHILE IN SERVICE ROUTINE
003302  012720  003510                    MOV     #INTRS2,(0)+
003306  012710  000340                    MOV     #340,    (0)
003312  013700  004264                    MOV     MOD3V,  R0         ;THE INTERRUPT SERV. ROUTINE WILL
003316  012720  003522                    MOV     #INTRS3,(0)+       ;ASSIGN A NUMBER AND STORE ON A STACK
003322  012720  000340                    MOV     #340,    (0)+      ;SO WE CAN DETERMINE LATER THE
003326  012720  003534                    MOV     #INTRS4,(0)+       ;ORDER THE INTERRUPTS CAME IN
003332  012710  000340                    MOV     #340,    (0)
003336  005037  000000                    CLR     0                  ;SURE WE CLEAR LOCATION ZERO.

003342  005713                   1$:      TST     (3)                ;DOES CONNECTION EXIST?
003344  001403                            BEQ     2$
003346  017372  000000  000000            MOV     @(3),    @(2)      ;SEND INPUT MODULES ADDR TO OUTPUT MODULE
003354  005723                   2$:      TST     (3)+               ;UPDATE POINTERS
003356  005722                            TST     (2)+
003360  020227  004260                    CMP     R2,      #MOD8A+2 ;AT END OF LIST?
003364  001366                            BNE     1$
003366  005037  177776                    CLR     PSW                ;NOW ALLOW THE WORLD TO INTR.
003372  005000                            CLR     R0                 ;MAKE SURE TO ALLOW PLENTY OF
003374  005200                            INC     R0                 ;TIME FOR EVERYONE
003376  001376                            BNE     .-2
003400  000005                            RESET                      ;TIMES-UP EVERYBODY SOULD BE
003402  052777  000100  000310            BIS     #100,@$TKS
                                                                     ;DONE
003410  012704  004302                    MOV     #INTPO-2,R4
003414  005724                   3$:      TST     (4)+
003416  021464  000002                    CMP     (4),     2(4)      ;FIRST < SECOND
003422  002403                            BLT     4$
003424  005764  000002                    TST     2(4)               ;WASN'T, BUT WAS IT A NON-EXSISTANT INITR.?
003430  001005                            BNE     AINTER
003432  020427  004310           4$:      CMP     R4,#INTPO+4        ;END OF LIST?
003436  001366                            BNE     3$
003440  000137  005762                    JMP     $EOP

003444  012705  004316           AINTER:  MOV     #INTPO1, R5        ;ERROR OCCURED - SETUP FOR TXPEQUT
003450  012537  001254                    MOV     (5)+,    $REG0
003454  012537  001256                    MOV     (5)+,    $REG1
003460  012537  001260                    MOV     (5)+,    $REG2
003464  012537  001262                    MOV     (5)+,    $REG3

003470  104006                            ERROR   6                  ;MODULES INTERRUPTED OUT OF SEQUENCE
                                                                     ;SHOULD BE (1) INPUT MODULE3, (2) INPUT MODULE4,
003472  000137  005762                    JMP     $EOP               ;(3) INPUT MODULE1, (4) INPUT MODULE2
                                                                     ;
                                                  ;INTERRUPT SERVICE ROUTINE FOR FIRST INPUT MODULE
003476  013725  004240           INTRS1:  MOV     MOD1A,   (5)+      ;STORE ADDRESS ON STACK
003502  012724  000003                    MOV     #3,      (4)+      ;STORE INTERRUPT VALUE ON STACK
003506  000002                            RTI

003510  013725  004242           INTRS2:  MOV     MOD2A,   (5)+
003514  012724  000004                    MOV     #4,      (4)+
003520  000002                            RTI

003522  013725  004244           INTRS3:  MOV     MOD3A,   (5)+
003526  012724  000001                  MOV     #1,     (4)+
003532  000002                          RTI

003534  013725  004246          INTRS4: MOV     MOD4A,  (5)+
003540  012724  000002                  MOV     #2,     (4)+
003544  000002                          RTI


                                                ;****************
                                                ;END OF TESTS
                                                ;****************
                                .SBTTL          HANDLERS
003546  117777  000150  005760  TTYIN:  MOVB    @STKB,  @TTYINP ;STORE INPUT
003554  142777  000200  005752          BICB    #200,   @TTYINP ;MASK FOR STANDARD INPUT
003562  122777  000003  005744          CMPB    #3,     @TTYINP ;CHECK FOR ^C
003570  001004                          BNE     .+12
003572  104400                          TYPE
003574  011167                          MCONC
003576  000137  005542                  JMP     STAR
003602  122777  000022  005724          CMPB    #22,    @TTYINP ;CHECK FOR ^R
003610  001002                          BNE     .+6             ; IF ^R THEN TYPE RUN SUMMARY
003612  000137  005476                  JMP     SUM
003616  122777  000177  005710          CMPB    #177,   @TTYINP ;CHECK FOR RUBOUT
003624  001002                          BNE     .+6
003626  000137  003724                  JMP     RUBH
003632  005737  003776                  TST     RUBF            ;NOT RUBOUT, BUT WAS FORMER A RUBOUT?
003636  001404                          BEQ     .+12
003640  005037  003776                  CLR     RUBF
003644  104400                          TYPE
003646  011174                          MBS
003650  117737  005660  005474          MOVB    @TTYINP,MAP1
003656  104400                          TYPE
003660  005474                          MAP1
003662  122777  000015  005644          CMPB    #15,    @TTYINP
003670  001010                          BNE     1$
003672  012737  011537  011534          MOV     #TTYINB-1,TTYINP ;DON'T GO TO DECODER IF
003700  005737  004330                  TST     CFLG            ;NOT IN COMMAND MODE
003704  001402                          BEQ     1$
003706  012716  004030                  MOV     #DESIFR,(6)     ;SET UP TO GO TO DECODER ROUTINE
003712  005237  011534          1$:     INC     TTYINP
003716  000002                          RTI

003720  177560                  $TKS:   177560
003722  177562                  $TKB:   177562

003724  022737  011540  011534  RUBH:   CMP     #TTYINB,TTYINP  ;AT BEGINNING OF BUFFER?
003732  001002                          BNE     .+6
003734  000137  005542                  JMP     STAR
003740  005737  003776                  TST     RUBF            ;FLAG SET?
003744  001004                          BNE     .+12
003746  005137  003776                  COM     RUBF
003752  104400                          TYPE                    ;NO, PRINT A SLASH
003754  011174                          MBS
003756  005337  011534                  DEC     TTYINP
```

A3

```
003762  117737  005546  005474           MOVB    @TTYINP,MAP1
003770  104400                           TYPE
003772  005474                           MAP1
003774  000002                           RTI
003776  000000                  RUBF:    000000


                                         ;THIS ROUTINE CHECKS TO SEE IF INPUT WAS A NUMBER

004000  127727  005530  000057  NUMBER:  CMPB    @TTYINP,#57      ;SEE IF INPUT IS A NUMBER
004006  002406                           BLT     .+16
004010  127727  005520  000071           CMPB    @TTYINP,#71
004016  003002                           BGT     .+6
004020  000262                           SEV                      ;SET V BIT IF WAS A NUMBER
004022  000207                           RTS PC
004024  000242                           CLV
004026  000207                           RTS PC


                                         ;COMMAND DECODER

004030  052737  000340  177776  DESIFR:  BIS     #340,   PSW      ;LOCK OUT INTERRUPTS
004036  104400                           TYPE
004040  011176                           MCRLF
004042  122777  000101  005464           CMPB    #101,   @TTYINP ;SEE IF THE WANTED TO INPUT ADDRESS "A"
004050  001002                           BNE     .+6
004052  000137  004332                   JMP     INADR
004056  122777  000126  005450           CMPB    #126,   @TTYINP ;SEE IF HE WANTED TO INPUT VECTOR "V"
004064  001002                           BNE     .+6
004066  000137  004660                   JMP     INVET
004072  122777  000106  005434           CMPB    #106,   @TTYINP ;SEE IF HE WANTED NORMAL CONNECTIONS "F"
004100  001002                           BNE     .+6
004102  000137  005036                   JMP     INNOR
004106  122777  000104  005420           CMPB    #104,   @TTYINP ;SEE IF HE WANTED TO DISCONNECT "D"
004114  001002                           BNE     .+6
004116  000137  005100                   JMP     INUNC
004122  122777  000115  005404           CMPB    #115,   @TTYINP ;SEE IF WANTED A MAP "M"
004130  001002                           BNE     .+6
004132  000137  005334                   JMP     INMAP
004136  122777  000123  005370           CMPB    #123,   @TTYINP ;SEE IF HE WANTED TO START TESTING "S"
004144  001002                           BNE     .+6
004146  000137  001446                   JMP     START
004152  004737  004000                   JSR     PC,     NUMBER  ;SEE IF HE WANTS TO CONNECT TWO MODULES
004156  102002                           BVC     .+6
004160  000137  005130                   JMP     SINCO


                                         ;
                                         ;UNKNOWN INPUT - TELL HIM
                                         ;

004164  005737  011534          UNKINP:  TST     TTYINP
004170  112777  000040  005336           MOVB    #40,    @TTYINP ;TYPE SPACE
004176  005237  011534                   INC     TTYINP
004202  112777  000077  005324           MOVB    #77,    @TTYINP ;TYPE "?"
```

```
 004210  005237  011534                     INC     TTYINP
 004214  105077  005314                     CLRB    @TTYINP                 ;TERMINATOR

 004220  104014                             ERROR 14                        ;UNKNOWN INPUT FROM OPERATOR

 004222  000137  005542                     JMP     STAR                    ;EXIT


 004226  032737  000001  004500   ODDADR:   BIT     #1,        NINP         ;IS IT AN ODD ADR?
 004234  001353                             BNE     UNKINP
 004236  000207                             RTS     PC


                                                    ;MAP OF MODULES

 004240  164000                   MOD1A:    164000             ;ADDR OF INPUT MODULES
 004242  164002                   MOD2A:    164002
 004244  164004                   MOD3A:    164004
 004246  164006                   MOD4A:    164006
 004250  164010                   MOD5A:    164010             ;ADDR OF OUTPUT MODULES
 004252  164012                   MOD6A:    164012
 004254  164014                   MOD7A:    164014
 004256  164016                   MOD8A:    164016
 004260  000170                   MOD1V:    000170             ;VECTOR OF INPUT MODULES
 004262  000174                   MOD2V:    000174
 004264  000270                   MOD3V:    000270
 004266  000274                   MOD4V:    000274
 004270  000000                   MOD5C:    000000             ;STATUS OF OUTPUT MODULES
 004272  000000                   MOD6C:    000000
 004274  000000                   MOD7C:    000000
 004276  000000                   MOD8C:    000000
 004300  000000                   VECTOR:   000000
 004302  000000                   SFIVE:    000000
 004304  000000                   INTPO:    000000
         004316                             .=.+10
 004316  000000                   INTPO1:   000000
         004330                             .=.+10
 004330  000000                   CFLG:     000000                          ;INDICATES SOFTWARE MODE



                                                    ;INPUT ADDRESS DECODER

 004332  005237  011534           INADR:    INC     TTYINP
 004336  122777  000111  005170             CMPB    #111,      @TTYINP ;INPUT MODULE ADDRS? "I"
 004344  001002                             BNE     .+6
 004346  000137  004504                     JMP     INADRI
 004352  122777  000117  005154             CMPB    #117,      @TTYINP ;OUTPUT MODULE ADDRS? "O"
 004360  001002                             BNE     .+6
 004362  000137  004572                     JMP     INADRO
 004366  000137  004164                     JMP     UNKINP                  ;UNKNOWN INPUT

                                                    ;THIS ROUTINE INPUTS A NUMBER

 004372  005037  004500           INUMB:    CLR     NINP                    ;CLR NUMBER
```

```
004376  062737  000002  011534            ADD     #2,      TTYINP  ;UPDATE POINTER
004404  004737  004000            1$:     JSR     PC,      NUMBER  ;SEE IF ITS A NUMBER
004410  102407                            BVS     2$               ;V BIT SET IF IT WAS
004412  122777  000015  005114            CMPB    #15,     @TTYINP ;CARRIAGE RETURN?
004420  001001                            BNE     .+4
004422  000207                            RTS     PC               ;EXIT
004424  000137  004164                    JMP     UNKINP           ;UNKNOWN INPUT
004430  000241                    2$:     CLC
004432  006137  004500                    ROL     NINP
004436  006137  004500                    ROL     NINP
004442  006137  004500                    ROL     NINP
004446  117737  005062  004502            MOVB    @TTYINP,NINPT    ;TEMP STORAGE OF NUMBER
004454  042737  177760  004502            BIC     #177760,NINPT
004462  063737  004502  004500            ADD     NINPT,   NINP    ;ADD NUMBER
004470  005237  011534                    INC     TTYINP
004474  000137  004404                    JMP     1$

004500  000000                    NINP:   000000
004502  000000                    NINPT:  000000


                                          ;THIS ROUTINE SETS ADDRS OF INPUT MODULES

004504  004737  004372            INADRI: JSR     PC,      INUMB   ;INPUT ADDRESS
004510  004737  004226            NTF0:   JSR     PC,      ODDADR  ;SEE IF ODD ADR
004514  013737  004500  004240            MOV     NINP,    MOD1A   ;SET INPUT MODULES ADDRESS
004522  062737  000002  004500            ADD     #2,      NINP
004530  013737  004500  004242            MOV     NINP,    MOD2A
004536  062737  000002  004500            ADD     #2,      NINP
004544  013737  004500  004244            MOV     NINP,    MOD3A
004552  062737  000002  004500            ADD     #2,      NINP
004560  013737  004500  004246            MOV     NINP,    MOD4A
004566  000137  005542                    JMP     STAR             ;EXIT


                                          ;THIS ROUTINE SETS ADDRESS OF OUTPUT MODULES

004572  004737  004372            INADRO: JSR     PC,      INUMB   ;INPUT ADDRESS
004576  004737  004226            NTF1:   JSR     PC,      ODDADR  ;SEE IF ODD ADDRESS
004602  013737  004500  004250            MOV     NINP,    MOD5A
004610  062737  000002  004500            ADD     #2,      NINP
004616  013737  004500  004252            MOV     NINP,    MOD6A
004624  062737  000002  004500            ADD     #2,      NINP
004632  013737  004500  004254            MOV     NINP,    MOD7A
004640  062737  000002  004500            ADD     #2,      NINP
004646  013737  004500  004256            MOV     NINP,    MOD8A
004654  000137  005542                    JMP     STAR             ;EXIT

                                          ;THIS ROUTINE SETS VECTOR ADDRESSES

004660  005237  011534            INVET:  INC     TTYINP           ;UPDATE POINTER
004664  005037  005024                    CLR     INVETT
004670  004737  004000                    JSR     PC,      NUMBER  ;SEE IF NUMBER FOLLOWS
004674  102402                            BVS     1$
```

D3

```
004676  000137  004164                    JMP     UNKINP
004702  117737  004626  005024  1$:       MOVB    @TTYINP,INVETT  ;TEMP STORAGE OF WHICH VECTORS
004710  004737  004372                    JSR     PC,     INUMB   ;GET ADDRESS
004714  004737  004226          NTF3B:    JSR     PC,     ODDADR  ;SEE IF ODD ADRESS
004720  162737  000001  005024            SUB     #1,     INVETT
004726  032737  177000  004500            BIT     #177000,NINP    ;SEE IF LEGAL VECTOR ADDR.
004734  001402                            BEQ     .+6
004736  000137  004164                    JMP     UNKINP          ;NO, NUMBER LARGER THAN 376
004742  042737  000176  005024            BIC     #176,   INVETT  ;ZERO OF ONE
004750  000241                            CLC
004752  006137  005024                    ROL     INVETT
004756  006137  005024                    ROL     INVETT
004762  062737  004260  005024            ADD     #MOD1V, INVETT  ;SET POINTER TO STORAGE OF VECTOR
004770  013777  004500  000026            MOV     NINP,   @INVETT ;STORE VECTOR ADDRESS
004776  062737  000002  005024            ADD     #2,     INVETT
005004  062737  000004  004500            ADD     #4,     NINP
005012  013777  004500  000004            MOV     NINP,   @INVETT
005020  000137  005542                    JMP     STAR
005024  000000                  INVETT:   000000
005026  000000                  INVET2:   000000
005030  000000                  S15:      000000                  ;TEMP STORAGE
005032  000000                  S16:      000000                  ;TEMP STORAGE
005034  000000                  S17:      000000                  ;INTR, ENABLE BITS TO SEND TO KIT H


                                                  ;THIS ROUTINE MAKES NORMAL CONNECTIONS

005036  012737  004240  004270  INNOR:    MOV     #MOD1A, MOD5C   ;MOD ADDR OF LOCATION THAT
                                                                  ;CONTAIN ADDR OF INPUT MODULE TO
005044  012737  004242  004272            MOV     #MOD2A, MOD6C   ;OUTPUT MODULES STATUS WORD (MEMORY)
005052  012737  004244  004274            MOV     #MOD3A, MOD7C
005060  012737  004246  004276            MOV     #MOD4A, MOD8C
005066  012737  000017  005034            MOV     #17,    S17
005074  000137  005542                    JMP     STAR


                                                  ;THIS ROUTINE UNCONNECTS MODULES

005100  005037  004270          INUNC:    CLR     MOD5C
005104  005037  004272                    CLR     MOD6C
005110  005037  004274                    CLR     MOD7C
005114  005037  004276                    CLR     MOD8C
005120  005037  005034                    CLR     S17
005124  000137  005542                    JMP     STAR


                                                  ;THIS ROUTINE CONNECTS TWO MODULES

005130  117737  004400  005024  SINCO:    MOVB    @TTYINP,INVETT  ;STORE INPUT MODULE # (1-4)
005136  042737  177760  005024            BIC     #177760,INVETT
005144  123727  005024  000004            CMPB    INVETT, #4      ;SEE IF ILLEGAL
005152  003011                            BGT     1$
005154  005337  005024                    DEC     INVETT
005160  100406                            BMI     1$
```

E3

```
005162  005237  011534                   INC     TTYINP
005166  122777  000074  004340           CMPB    #74,     @TTYINP ;SHOULD GET "<"
005174  001402                           BEQ     .+6
005176  000137  004164          1$:      JMP     UNKINP           ;UNKNOWN INPUT
005202  005237  011534                   INC     TTYINP
005206  013737  005024  005032           MOV     INVETT, S16      ;SET INTR, ENABLE BITS OF ONLY
005214  005037  005030                   CLR     S15              ;THOSE MODULES THAT WERE SELECTED
005220  000261                           SEC
005222  006137  005030          2$:      ROL     S15              ;FORM ENABLE BIT BY NUMBER OF
005226  005337  005032                   DEC     S16              ;INPUT MODULE SELECTED
005232  100373                           BPL     2$
005234  053737  005030  005034           BIS     S15,S17
005242  004737  004000                   JSR     PC,      NUMBER  ;CHECK FOR NUMBER
005246  117737  004262  005026           MOVB    @TTYINP,INVET2   ;STORE # OF OUTPUT MODULE (5-8)
005254  042737  177760  005026           BIC     #177760,INVET2
005262  162737  000005  005026           SUB     #5,      INVET2  ;NUMBER CANNOT BE LESS THAN 5
005270  100742                           BMI     1$
005272  000241                           CLC
005274  006137  005024                   ROL     INVETT           ;GET ACCUAL ADDR
005300  062737  004240  005024           ADD     #MOD1A, INVETT
005306  000241                           CLC
005310  006137  005026                   ROL     INVET2
005314  062737  004270  005026           ADD     #MOD5C, INVET2
005322  013777  005024  177476           MOV     INVETT, @INVET2 ;DO IT
005330  000137  005542                   JMP     STAR             ;EXIT



                                                 ;THIS ROUTINE MAPS AVAILABLE INFORMATION

005334  005037  177776          INMAP:   CLR     PSW              ;ALLOWS INTERRUPTS
005340  104400                           TYPE                     ;TYPE MAP HEADER
005342  011234                           MMHD
005344  012737  000261  005474           MOV     #261,    MAP1    ;SET FOR FIRST MODULE
005352  012701  004240                   MOV     #MOD1A, R1
005356  012702  004260                   MOV     #MOD1V, R2
005362  104400                  INMAP1:  TYPE
005364  011321                           M8SP                     ;TYPE 8S SPACES (CARRIAGE RETURN)
005366  104400                           TYPE
005370  005474                           MAP1                     ;TYPE MODULE NUMBER
005372  104400                           TYPE
005374  011334                           M3SP
005376  012146                           MOV     (1)+,-(SP)               ;SAVE (1)+ FOR TYPEOUT
005400  104402                           TYPOC                            ;GO TYPE--OCTAL ASCII(ALL DIGITS)
005402  104400                           TYPE                     ;TYPE "2SP INPUT 5SP"
005404  011340                           MFUN
005406  012246                           MOV     (2)+,-(SP)               ;SAVE (2)+ FOR TYPEOUT
005410  104402                           TYPOC                            ;GO TYPE--OCTAL ASCII(ALL DIGITS)
005412  005237  005474                   INC     MAP1
005416  022701  004250                   CMP     #MOD5A, R1       ;DONE ALL INPUT MODULES?
005422  001357                           BNE     INMAP1
005424  104400                  1$:      TYPE
005426  011321                           M8SP
005430  104400                           TYPE
005432  005474                           MAP1
```

F3

```
005434  104400                              TYPE
005436  011334                              M3SP
005440  012146                              MOV     (1)+,-(SP)                      ;SAVE (1)+ FOR TYPEOUT
005442  104402                              TYPOC                                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
005444  104400                              TYPE
005446  011355                              MFUNC2                          ;TYPE "2SP OUTPUT 6SP N/A 2SP"
005450  013246                              MOV     @(2)+,  -(SP)
005452  001401                              BEQ     2$
005454  104402                              TYPOC                           ;TYPE CONNECTION
005456  005237  005474              2$:     INC     MAP1
005462  020127  004260                      CMP     R1,     #MOD8A+2
005466  001356                              BNE     1$
005470  000137  005542                      JMP     STAR
005474  000000                      MAP1:   000000






                                                    ;
                                                    ;RUN SUMMARY TYPEOUT ROUTINE
                                                    ;
005476  104400                      SUM:    TYPE                            ;TYPE HEADER
005500  011440                              MSUM
005502  013746  001216                      MOV     $PASS,-(SP)                     ;SAVE $PASS FOR TYPEOUT
005506  104402                              TYPOC                                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
005510  104400                              TYPE
005512  011334                              M3SP
005514  013746  001230                      MOV     $ERTTL,-(SP)                    ;SAVE $ERTTL FOR TYPEOUT
005520  104402                              TYPOC                                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
005522  104400                              TYPE
005524  011176                              MCRLF
005526  005737  004330                      TST     CFLG                    ;IF IN COMMAND MODE RETURN TO STAR
005532  001402                              BEQ     .+6
005534  000137  005542                      JMP     STAR
005540  000002                              RTI                             ;RETURN TO WHAT WE WERE DOING
                                                    ;MONITOR HOME

005542                              STAR:
005542  012706  001100                      MOV     #STACK,SP                       ;SETUP THE STACK POINTER
005546  012737  006034  000020              MOV     #$SCOPE,@#IOTVEC                ;IOT VECTOR FOR SCOPE ROUTINE
005554  012737  000340  000022              MOV     #340,@#IOTVEC+2                 ;LEVEL 7
005562  105037  001220                      CLRB    $TSTNM                          ;INITIALIZE THE TEST NUMBER
005566  012737  005542  001224              MOV     #STAR,$LPADR                    ;INITIALIZE THE LOOP ADDRESS FOR SCOPE
005574  012737  006212  000030              MOV     #$ERROR,@#EMTVEC                        ;EMT VECTOR FOR ERROR(ERROR) ROUTINE
005602  012737  000340  000032              MOV     #340,@#EMTVEC+2                 ;LEVEL 7
005610  012737  007014  000034              MOV     #$TRAP,@#TRAPVEC                ;TRAP VECTOR FOR TRAP CALLS
005616  012737  000340  000036              MOV     #340,@#TRAPVEC+2                ;LEVEL 7
005624  012737  007044  000024              MOV     #$PWRDN,@#PWRVEC                ;POWER FAILURE VECTOR
005632  012737  000340  000026              MOV     #340,@#PWRVEC+2                 ;LEVEL 7
005640  005037  001216                      CLR     $PASS                           ;CLEAR THE PASS COUNT
005644  005037  001222                      CLR     $ICNT                           ;INITIALIZE THE ITERATION COUNTER
005650  005037  001300                      CLR     $TIMES                          ;INITIALIZE NUMBER OF ITERATIONS
```

63

```
005654  105037  001221                     CLRB    $ERFLG                       ;CLEAR THE ERROR FLAG

005660  005037  001230                     CLR     $ERTTL
005664  005237  004330                     INC     CFLG               ;SET COMMAND MODE
005670  012737  000340  177776             MOV     #340,    PSW
005676  005037  003776                     CLR     RUBF
005702  012737  011540  011534             MOV     #TTYINB,TTYINP
005710  105737  001107                     TSTB    $TPFLG             ;SEE IF THERE IS A TERMINAL
005714  001402                             BEQ     1$                 ;IF YES GO AHEAD NORMALLY
005716  000137  007200                     JMP     NTH                ;IF NOT THEN GO TO NO TERMINAL HANDLER
005722  012737  003546  000060     1$:     MOV     #TTYIN,@#60
005730  012737  000340  000062             MOV     #340,    @#62
005736  005037  177776                     CLR     PSW
005742  104400                             TYPE                       ;TYPE "*"
005744  011402                             MSTAR
005746  052777  000100  175744             BIS     #100,@$TKS
005754  000001                             WAIT                       ;SPEND REST OF TIME HERE
005756  000137  005754                     JMP     .-2


                                   ;;*************************************************************

                                   .SBTTL          END OF PASS ROUTINE

                                   ;*INCREMENT THE PASS NUMBER
                                   ;*IF THERES A MONITOR GO TO IT
                                   ;*IF THERE ISN'T JUMP TO START
005762  004737  007500             $EOP:   JSR     PC,EOPT
005766  005037  001220                     CLR     $TSTNM                       ;ZERO THE TEST NUMBER
005772  005037  001300                     CLR     $TIMES                       ;ZERO THE NUMBER OF ITERATIONS
005776  005237  001216                     INC     $PASS                        ;INCREMENT THE PASS NUMBER
006002  032737                             BIT     (PC)+,@(PC)+                 ;LOOP?
006004  000000                     $ENDCT: .WORD   0
006006  001216                             $PASS
006010  001007                             BNE     $DOAGN                       ;YUP
006012  013700  000042             $GET42: MOV     @#42,R0                      ;GET MONITOR ADDRESS
006016  001404                             BEQ     $DOAGN                       ;IF NONE
006020  004710                     $ENDAD: JSR     PC,(R0)                      ;GO TO MONITOR
006022  000240                             NOP                                  ;SAVE ROOM
006024  000240                             NOP                                  ;FOR
006026  000240                             NOP                                  ;ACT11
006030  000137  001446             $DOAGN: JMP     @#START                      ;RETURN
```

H3

```
                                        ;;*************************************************************

                                        .SBTTL          SCOPE HANDLER ROUTINE

                                        ;*SW14=1         LOOP ON TEST
                                        ;*SW11=1         INHIBIT ITERATIONS
                                        ;*THE TEST NUMBER ($TSTNM) IS INCREMENTED AND DISPLAYED IN DISPLAY<7:0>
                                        ;*AND THE ERROR FLAG ($ERFLG) IS DISPLAYED IN DISPLAY<15:08>
        006034                          $SCOPE:
        006034  006137  177570                  ROL     @#SWR                   ;LOOP ON PRESENT TEST?
        006040  100455                          BMI     $OVER                   ;YES IF SW14=1
                                                ;#####START OF CODE FOR THE XOR TESTER#####
        006042  000416                  $XTSTR: BR      6$                      ;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                                ;THIS INSTRUCTION TO A "NOP" (NOP=240)
        006044  013746  000004                  MOV     @#ERRVEC,-(SP)          ;SAVE THE CONTENTS OF THE ERROR VECTOR
        006050  012737  006070  000004          MOV     #5$,@#ERRVEC            ;SET FOR TIMEOUT
        006056  005737  177060                  TST     @#177060                ;TIME OUT ON XOR?
        006062  012637  000004                  MOV     (SP)+,@#ERRVEC          ;RESTORE THE ERROR VECTOR
        006066  000436                          BR      $SVLAD                  ;GO TO THE NEXT TEST
        006070  022626                  5$:     CMP     (SP)+,(SP)+             ;CLEAR THE STACK AFTER A TIME OUT
        006072  012637  000004                  MOV     (SP)+,@#ERRVEC          ;RESTORE THE ERROR VECTOR
        006076  000436                          BR      $OVER                   ;LOOP ON THE PRESENT TEST
        006100                          6$:     ;#####END OF CODE FOR THE XOR TESTER#####
        006100  105737  001221          2$:     TSTB    $ERFLG                  ;HAS AN ERROR OCCURRED?
        006104  001404                          BEQ     3$                      ;BR IF NO
        006106  105037  001221          4$:     CLRB    $ERFLG                  ;ZERO THE ERROR FLAG
        006112  005037  001300                  CLR     $TIMES                  ;CLEAR THE NUMBER OF ITERATIONS TO MAKE
        006116  032737  004000  177570  3$:     BIT     #SW11,@#SWR             ;INHIBIT ITERATIONS?
        006124  001011                          BNE     1$                      ;BR IF YES
        006126  005737  001216                  TST     $PASS                   ;IF FIRST PASS OF PROGRAM
        006132  001406                          BEQ     1$                      ;       INHIBIT ITERATIONS
        006134  005237  001222                  INC     $ICNT                   ;INCREMENT ITERATION COUNT
        006140  023737  001300  001222          CMP     $TIMES,$ICNT            ;CHECK THE NUMBER OF ITERATIONS MADE
        006146  002012                          BGE     $OVER                   ;BR IF MORE ITERATION REQUIRED
        006150  012737  000001  001222  1$:     MOV     #1,$ICNT                ;REINITIALIZE THE ITERATION COUNTER
        006156  013737  006210  001300          MOV     $MXCNT,$TIMES           ;SET NUMBER OF ITERATIONS TO DO
        006164  105237  001220          $SVLAD: INCB    $TSTNM                  ;COUNT TEST NUMBERS
        006170  011637  001224                  MOV     (SP),$LPADR             ;SAVE SCOPE LOOP ADDRESS
        006174  013737  001220  177570  $OVER:  MOV     $TSTNM,@#DISPLAY        ;DISPLAY TEST NUMBER
        006202  013716  001224                  MOV     $LPADR,(SP)             ;FUDGE RETURN ADDRESS
        006206  000002                          RTI                             ;FIXES PS
        006210  000020                  $MXCNT: 20                              ;MAX. NUMBER OF ITERATIONS
                                        ;;*************************************************************

                                        .SBTTL          ERROR HANDLER ROUTINE

                                        ;*SW15=1         HALT ON ERROR
                                        ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                        ;*GO TO $ERRTYP ON ERROR
        006212                          $ERROR:
        006212  004737  007416                  JSR PC,EEDNH
        006216  105237  001221          7$:     INCB    $ERFLG                  ;SET THE ERROR FLAG
        006222  001775                          BEQ     7$                      ;DON'T LET THE FLAG GO TO ZERO
        006224  005237  001230                  INC     $ERTTL                  ;INC THE ERROR COUNT
```

I3

```
006230  011637  001240                          MOV     (SP),$ERRAD             ;GET ADDRESS OF ERROR INSTRUCTION
006234  162737  000002  001240                  SUB     #2,$ERRAD
006242  117737  172772  001236                  MOVB    @$ERRAD,$ITEMB          ;STRIP AND SAVE THE ERROR ITEM CODE
006250  032737  020000  177570                  BIT     #SW13,@#SWR             ;SKIP TYPEOUT IF SET
006256  001004                                  BNE     2$                      ;SKIP TYPEOUTS
006260  004737  006302                          JSR     PC,@#$ERRTYP            ;GO TO USER ERROR ROUTINE
006264  104400  001303                          TYPE    ,$CRLF
006270  005737  177570                  2$:     TST     @#SWR                   ;HALT ON ERROR
006274  100001                                  BPL     3$                      ;SKIP IF CONTINUE
006276  000000                                  HALT                            ;HALT ON ERROR!
006300  000002                          3$:     RTI
                                        ;;*************************************************************

                                        .SBTTL          ERROR MESSAGE TYPEOUT ROUTINE

                                        ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                        ;*ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                                        ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR,

006302                                  $ERRTYP:
006302  104400  001303                          TYPE    ,$CRLF                  ;"CARRIAGE RETURN" & "LINE FEED"
006306  010046                                  MOV     R0,-(SP)                ;SAVE R0
006310  005000                                  CLR     R0                      ;PICKUP THE ITEM INDEX
006312  153700  001236                          BISB    @#$ITEMB,R0
006316  001004                                  BNE     1$                      ;IF ITEM NUMBER IS ZERO ONLY
                                                                                ;TYPE THE PC OF THE ERROR
006320  013746  001240                          MOV     $ERRAD,-(SP)            ;SAVE $ERRAD FOR TYPEOUT
                                                                                ;ERROR ADDRESS
006324  104402                                  TYPOC                           ;GO TYPE--OCTAL ASCII(ALL DIGITS)
006326  000426                                  BR      6$                      ;GET OUT
006330  005300                          1$:     DEC     R0                      ;ADJUST THE INDEX SO THAT IT WILL
006332  006300                                  ASL     R0                      ;       WORK FOR THE ERROR TABLE
006334  006300                                  ASL     R0
006336  006300                                  ASL     R0
006340  062700  001306                          ADD     #$ERRTB,R0              ;FORM TABLE POINTER
006344  012037  006354                          MOV     (R0)+,2$                ;PICKUP "ERROR MESSAGE" POINTER
006350  001404                                  BEQ     3$                      ;SKIP TYPEOUT IF NO POINTER
006352  104400                                  TYPE                            ;TYPE THE "ERROR MESSAGE"
006354  000000                          2$:     .WORD   0                       ;"ERROR MESSAGE" POINTER GOES HERE
006356  104400  001303                          TYPE    ,$CRLF                  ;"CARRIAGE RETURN" & "LINE FEED"
006362  012037  006372                  3$:     MOV     (R0)+,4$                ;PICKUP "DATA HEADER" POINTER
006366  001404                                  BEQ     5$                      ;SKIP TYPEOUT IF 0
006370  104400                                  TYPE                            ;TYPE THE "DATA HEADER"
006372  000000                          4$:     .WORD   0                       ;"DATA HEADER" POINTER GOES HERE
006374  104400  001303                          TYPE    ,$CRLF                  ;"CARRIAGE RETURN" & "LINE FEED"
006400  011000                          5$:     MOV     (R0),R0                 ;PICKUP "DATA TABLE" POINTER
006402  001004                                  BNE     7$                      ;GO TYPE THE DATA
006404  012600                          6$:     MOV     (SP)+,R0                ;RESTORE R0
006406  104400  001303                          TYPE    ,$CRLF                  ;"CARRIAGE RETURN" & "LINE FEED"
006412  000207                                  RTS     PC                      ;RETURN
006414                                  7$:
006414  013046                                  MOV     @(R0)+,-(SP)            ;SAVE @(R0)+ FOR TYPEOUT
006416  104402                                  TYPOC                           ;GO TYPE--OCTAL ASCII(ALL DIGITS)
006420  005710                                  TST     (R0)                    ;IS THERE ANOTHER NUMBER?
```

J3

```
006422  001770                          BEQ     6$                      ;BR IF NO
006424  104400  006432                  TYPE    ,8$                     ;TYPE TWO(2) SPACES
006430  000771                          BR      7$                      ;LOOP
006432  020040     000          8$:     .ASCIZ  /  /                    ;TWO(2) SPACES
        006436                          .EVEN
                                ;;*****************************************************************

                                .SBTTL          BINARY TO OCTAL (ASCII) AND TYPE

                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)               ;NUMBER TO BE TYPED
                                ;*      TYPOS                           ;CALL FOR TYPEOUT
                                ;*      .BYTE   N                       ;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                ;*      .BYTE   M                       ;M=1 OR 0
                                ;*                                              ;1=TYPE LEADING ZEROS
                                ;*                                              ;0=SUPPRESS LEADING ZEROS
                                ;*
                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                ;*$TYPOS OR $TYPOCT
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)               ;NUMBER TO BE TYPED
                                ;*      TYPON                           ;CALL FOR TYPEOUT
                                ;*
                                ;*$TYPOCT---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)               ;NUMBER TO BE TYPED
                                ;*      TYPOC                           ;CALL FOR TYPEOUT

006436  017646  000000          $TYPOS: MOV     @(SP),-(SP)             ;PICKUP THE MODE
006442  116637  000001  006661          MOVB    1(SP),$0FILL            ;LOAD ZERO FILL SWITCH
006450  112637  006663                  MOVB    (SP)+,$OMODE+1          ;NUMBER OF DIGITS TO TYPE
006454  062716  000002                  ADD     #2,(SP)                 ;ADJUST RETURN ADDRESS
006460  000406                          BR      $TYPON
006462                          $TYPOCT:
006462  112737  000001  006661          MOVB    #1,$0FILL               ;SET THE ZERO FILL SWITCH
006470  112737  000006  006663          MOVB    #6,$OMODE+1             ;SET FOR SIX(6) DIGITS
006476  112737  000005  006660  $TYPON: MOVB    #5,$OCNT                ;SET THE ITERATION COUNT
006504  010346                          MOV     R3,-(SP)                ;SAVE R3
006506  010446                          MOV     R4,-(SP)                ;SAVE R4
006510  010546                          MOV     R5,-(SP)                ;SAVE R5
006512  113704  006663                  MOVB    $OMODE+1,R4             ;GET THE NUMBER OF DIGITS TO TYPE
006516  005404                          NEG     R4
006520  062704  000006                  ADD     #6,R4                   ;SUBTRACT IT FOR MAX, ALLOWED
006524  110437  006662                  MOVB    R4,$OMODE               ;SAVE IT FOR USE
006530  113704  006661                  MOVB    $0FILL,R4               ;GET THE ZERO FILL SWITCH
006534  016605  000012                  MOV     12(SP),R5               ;PICKUP THE INPUT NUMBER
006540  005003                          CLR     R3                      ;CLEAR THE OUTPUT WORD
006542  006105                  1$:     ROL     R5                      ;ROTATE MSB INTO "C"
006544  000404                          BR      3$                      ;GO DO MSB
006546  006105                  2$:     ROL     R5                      ;FORM THIS DIGIT
006550  006105                          ROL     R5
006552  006105                          ROL     R5
006554  010503                          MOV     R5,R3
```

```
006556  006103                          3$:     ROL     R3                              ;GET LSB OF THIS DIGIT
006560  105337  006662                          DECB    $OMODE                          ;TYPE THIS DIGIT?
006564  100016                                  BPL     7$                              ;BR IF NO
006566  042703  177770                          BIC     #177770,R3                      ;GET RID OF JUNK
006572  001002                                  BNE     4$                              ;TEST FOR 0
006574  005704                                  TST     R4                              ;SUPPRESS THIS 0?
006576  001403                                  BEQ     5$                              ;BR IF YES
006600  005204                          4$:     INC     R4                              ;DON'T SUPPRESS ANYMORE 0'S
006602  052703  000060                          BIS     #'0,R3                          ;MAKE THIS DIGIT ASCII
006606  052703  000040                  5$:     BIS     #' ,R3                          ;MAKE ASCII IF NOT ALREADY
006612  110337  006656                          MOVB    R3,8$                           ;SAVE FOR TYPING
006616  104400  006656                          TYPE    ,8$                             ;GO TYPE THIS DIGIT
006622  105337  006660                  7$:     DECB    $OCNT                           ;COUNT BY 1
006626  003347                                  BGT     2$                              ;BR IF MORE TO DO
006630  002402                                  BLT     6$                              ;BR IF DONE
006632  005204                                  INC     R4                              ;INSURE LAST DIGIT ISN'T A BLANK
006634  000744                                  BR      2$                              ;GO DO THE LAST DIGIT
006636  012605                          6$:     MOV     (SP)+,R5                        ;RESTORE R5
006640  012604                                  MOV     (SP)+,R4                        ;RESTORE R4
006642  012603                                  MOV     (SP)+,R3                        ;RESTORE R3
006644  016666  000002  000004                  MOV     2(SP),4(SP)                     ;SET THE STACK FOR RETURNING
006652  012616                                  MOV     (SP)+,(SP)
006654  000002                                  RTI                                     ;RETURN
006656     000                          8$:     .BYTE   0                               ;STORAGE FOR ASCII DIGIT
006657     000                                  .BYTE   0                               ;TERMINATOR FOR TYPE ROUTINE
006660     000                          $OCNT:  .BYTE   0                               ;OCTAL DIGIT COUNTER
006661     000                          $OFILL: .BYTE   0                               ;ZERO FILL SWITCH
006662  000000                          $OMODE: 0                                       ;NUMBER OF DIGITS TO TYPE
```

L3

```
                              ;;*****************************************************************

                              .SBTTL          RANDOM NUMBER GENERATOR ROUTINE

                              ;*CALL:
                              ;*      JSR     PC,$RAND                        ;CALL THE ROUTINE
                              ;*      RETURN                                  ;RETURN HERE THE RANDOM
                              ;*                                              ;NUMBER WILL BE IN
                              ;*                                              ;$HINUM,$LONUM
      006664                  $RAND:
      006664  010046                  MOV     R0,-(SP)                        ;PUSH R0 ON STACK
      006666  010146                  MOV     R1,-(SP)                        ;PUSH R1 ON STACK
      006670  010246                  MOV     R2,-(SP)                        ;PUSH R2 ON STACK
      006672  010346                  MOV     R3,-(SP)                        ;PUSH R3 ON STACK
      006674  013700  007012          MOV     $LONUM,R0                       ;SET R0 WITH LOW
      006700  013701  007010          MOV     $HINUM,R1                       ;SET R1 WITH HIGH
      006704  012703  177771          MOV     #-7,R3                          ;SET SHIFT COUNT
      006710  005002                  CLR     R2                              ;ZERO R2
      006712  006300          1$:     ASL     R0                              ;SHIFT R0 LEFT AND
      006714  006101                  ROL     R1                              ;ROTATE CARRY INTO R1 AND
      006716  006102                  ROL     R2                              ;ROTATE CARRY INTO R2
      006720  005203                  INC     R3                              ;CHECK FOR DONE
      006722  001373                  BNE     1$                              ;CONTINUE SHIFT LOOP
      006724  063702  007012          ADD     $LONUM,R2                       ;ADD NUMBER TO MAKE X 129
      006730  005501                  ADC     R1                              ;PROPOGATE CARRY
      006732  063701  007010          ADD     $HINUM,R1                       ;ADD NUMBER TO MAKE X 129
      006736  005502                  ADC     R2                              ;PROPOGATE CARRY
      006740  062700  001057          ADD     #1057,R0                        ;ADD LOW CONSTANT
      006744  005501                  ADC     R1                              ;PROPOGATE CARRY
      006746  005502                  ADC     R2                              ;PROPOGATE CARRY
      006750  062701  047401          ADD     #47401,R1                       ;ADD HIGH CONSTANT
      006754  005502                  ADC     R2                              ;PROPOGATE CARRY
      006756  062702  000006          ADD     #6,R2                           ;ADD HIGHEST CONSTANT
      006762  060200                  ADD     R2,R0                           ;REPRIME R0 WITH HIGHEST DIGIT
      006764  005501                  ADC     R1                              ;PROPOGATE CARRY
      006766  010037  007012          MOV     R0,$LONUM                       ;SAVE R0
      006772  010137  007010          MOV     R1,$HINUM                       ;SAVE R1
      006776  012603                  MOV     (SP)+,R3                        ;POP STACK INTO R3
      007000  012602                  MOV     (SP)+,R2                        ;POP STACK INTO R2
      007002  012601                  MOV     (SP)+,R1                        ;POP STACK INTO R1
      007004  012600                  MOV     (SP)+,R0                        ;POP STACK INTO R0
      007006  000207                  RTS     PC                              ;RETURN
      007010  176543          $HINUM: .WORD   176543
      007012  123456          $LONUM: .WORD   123456
                              ;;*****************************************************************

                              .SBTTL          TRAP DECODER

                              ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                              ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                              ;*OF THE DESIRED ROUTINE, THEN USING THE ADDRESS OBTAINED IT WILL
                              ;*GO TO THAT ROUTINE.

      007014  010046          $TRAP:  MOV     R0,-(SP)                        ;SAVE R0
```

M12

```
007016  016600  000002                  MOV     2(SP),R0                ;GET TRAP ADDRESS
007022  005740                          TST     -(R0)                   ;BACKUP BY 2
007024  111000                          MOVB    (R0),R0                 ;GET RIGHT BYTE OF TRAP
007026  016000  007034                  MOV     $TRPAD(R0),R0           ;INDEX TO TABLE
007032  000200                          RTS     R0                      ;GO TO ROUTINE


                                .SBTTL          TRAP TABLE

                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.

                                ;       ROUTINE
                                ;       -------
007034  001110                  $TRPAD: $TYPE           ;CALL=TYPE      TRAP+0(104400)  TTY TYPEOUT ROUTINE
007036  006462                          $TYPOCT         ;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING
007040  006436                          $TYPOS          ;CALL=TYPOS     TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZE
007042  006476                          $TYPON          ;CALL=TYPON     TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST C
                                ;;*************************************************************

                                .SBTTL          POWER DOWN AND UP ROUTINES

                                ;POWER DOWN ROUTINE
007044  012737  007172  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC        ;SET FOR FAST UP
007052  012737  000340  000026          MOV     #340,@#PWRVEC+2         ;PRIO:7
007060  010046                          MOV     R0,-(SP)                ;PUSH R0 ON STACK
007062  010146                          MOV     R1,-(SP)                ;PUSH R1 ON STACK
007064  010246                          MOV     R2,-(SP)                ;PUSH R2 ON STACK
007066  010346                          MOV     R3,-(SP)                ;PUSH R3 ON STACK
007070  010446                          MOV     R4,-(SP)                ;PUSH R4 ON STACK
007072  010546                          MOV     R5,-(SP)                ;PUSH R5 ON STACK
007074  010637  007176                  MOV     SP,$SAVR6               ;SAVE SP
007100  012737  007112  000024          MOV     #$PWRUP,@#PWRVEC        ;SET UP VECTOR
007106  000000                          HALT
007110  000776                          BR      .-2                     ;HANG UP

                                ;POWER UP ROUTINE
007112  013706  007176          $PWRUP: MOV     $SAVR6,SP               ;GET SP
007116  005037  007176                  CLR     $SAVR6                  ;WAIT LOOP FOR THE TTY
007122  005237  007176          1$:     INC     $SAVR6                  ;WAIT FOR THE INC
007126  001375                          BNE     1$                      ;OF  WORD
007130  012605                          MOV     (SP)+,R5                ;POP STACK INTO R5
007132  012604                          MOV     (SP)+,R4                ;POP STACK INTO R4
007134  012603                          MOV     (SP)+,R3                ;POP STACK INTO R3
007136  012602                          MOV     (SP)+,R2                ;POP STACK INTO R2
007140  012601                          MOV     (SP)+,R1                ;POP STACK INTO R1
007142  012600                          MOV     (SP)+,R0                ;POP STACK INTO R0
007144  012737  007044  000024          MOV     #$PWRDN,@#PWRVEC        ;SET UP THE POWER DOWN VECTOR
007152  012737  000340  000026          MOV     #340,@#PWRVEC+2         ;PRIO:7
007160  104400  011464                  TYPE    ,POWER                  ;POWER FAIL MESSAGE
007164  012716  005542                  MOV     #STAR,(SP)              ;RESTART AT STAR
007170  000002                          RTI
007172  000000                  $ILLUP: HALT                            ;THE POWER UP SEQUENCE WAS STARTED
007174  000776                          BR      .-2                     ; BEFORE THE POWER DOWN WAS COMPLETE
```

```
007176  000000                         $SAVR6: 0                                   ;PUT THE SP HERE


                                                        ;
                                                        ;NO TERMINAL HANDLER
                                                        ;ENTER HERE FROM START AT 210
                                                        ;

007200  012706  001100          NTH:    MOV     #1100,  SP              ;SET UP STACK POINTER
007204  112737  000001  001107          MOVB    #1,     $TPFLG          ;REMEMBER WE HAVE NO TERMINAL
007212  000000                          HALT                    ;WAITE FOR DIRECTIVE
007214  113700  177570                  MOVB    SWR,    R0      ;SAVE DIRECTIVE FOR LIST
007220  001002                          BNE     1$              ;IF HE WANTED TO START TESTING
007222  000137  001446                  JMP     START           ;DON'T HALT AGAIN
007226. 000000                  1$:     HALT                    ;WAIT HERE FOR ADDRESS
007230  013737  177570  004500          MOV     SWR,    NINP    ;STORE ADDRESS
007236  042700  177761                  BIC     #177761,        R0      ;MAKE SURE HE DIN'T GIVE ME BAD DIRECTIVE
007242  062700  007254                  ADD     #NTHFP, R0      ;GET READY TO GO TO A PLACE THAT
007246  000170  000000                  JMP     @(0)            ;THAT WILL TAKE CARE OF DIRECTIVE

007252  000000                  NTHF:   000000                  ;DIRECTIVE POINTER
007254  001446                  NTHFP:  START                   ;SWR=0  START TESTING
007256  004510                          NTF0                    ;SWR=2  INPUT MODULE ADR.
007260  004576                          NTF1                    ;SWR=4  ENTER OUTPUT MODULE ADR.
007262  007274                          NTF2                    ;SWR=6  ENTER IST GROUP VECTER ADR.
007264  007304                          NTF3                    ;SWR=10 ENTER 2ND GROUP VECTER ADR.
007266  005100                          INUNC                   ;SWR=12 UNCONNECT MODULES
007270  005036                          INNOR                   ;SWR=14 MAKE NORMAL CONNECTIONS
007272  007316                          NTF4                    ;SWR=16 CONNECT TWO MODULES (SOFT)
                                                                ;NOTE:  INPUT MODULE # IN SWR 0-2
                                                                ;       OUTPUT MODULE # IN SWR 3-6


                                                        ;
                                                        ;THIS ROUTINE HANLDES INPUTING VECTORS
                                                        ;
007274  012737  000001  005024  NTF2:   MOV     #1,     INVETT  ;ENTER VECTOR FIRST GROUP
007302  000403                          BR      NTF3A           ;BRANCH AHEAD
007304  012737  000002  005024  NTF3:   MOV     #2,     INVETT  ;ENTER VECTOR SECOUND GROUP
007312  000137  004714          NTF3A:  JMP     NTF3B           ;NOW GO TO NORMAL ROUTINE THAT TAKES CARE OF IT
                                                        ;
                                                        ;THIS ROUTINE HANDLES CONNECTING 2 MODULES WITH NO
                                                        ;TERMINAL BY SETTING UP INPUT BY SWR TO LOOK
                                                        ;LIKE INPUT FROM A TTY THEN TRANSFERRING TO NORMAL
                                                        ;ROUTINE
                                                        ;
007316  113737  177570  001270  NTF4:   MOVB    SWR,    $TMP0   ;GET SWR
007324  042737  177770  001270          BIC     #177770,$TMP0   ;FORM INPUT MODULE #
007332  113737  001270  011540          MOVB    $TMP0,  TTYINB  ;FUDGE IT TO LOOK LIKE IT
007340  112737  000074  011541          MOVB    #74,    TTYINB+1 ;CAME FROM TTY INPUT
007346  113737  177570  001270          MOVB    SWR,    $TMP0   ;GET SWR
007354  006037  001270                  ROR     $TMP0           ;FORM OUTPUT MODULE #
007360  006037  001270                  ROR     $TMP0
007364  006037  001270                  ROR     $TMP0
007370  042737  177760  001270          BIC     #177760,$TMP0   ;MAKE IT LOOK LIKE
```

RY

```
  1539  007376  113737  001270  011542          MOVB    $IMPO,  TTYINB+2 ;TTY INPUT - THEN
```

C4

```
007404  012737  011540  011534                MOV     #TTYINB,TTYINP  ;GO TO ROUTINE THAT HANDLES TTY
007412  000137  005130                        JMP     SINCO           ;INPUT FOR SINGLE CONNECTIONS


                                                      ;ERROR HANDLER DOES OUTPUT TERMINAL EXIST?

007416  005037  177776                EEDNH:  CLR     PSW
007422  105737  001107                        TSTB    $TPFLG          ;DOES TTY EXIST?
007426  001001                                BNE     2$
007430  000207                        1$:     RTS PC                  ;YES-EXIT
007432  032737  020000  177570        2$:     BIT     #SW13,@#SWR
007440  001373                                BNE     1$

007442  011637  001240                        MOV     (SP),   $ERRAD  ;GET ADDRESS OF ERROR CALL
007446  162737  000002  001240                SUB     #2,     $ERRAD
007454  117737  171560  001236                MOVB    @$ERRAD,$ITEMB  ;GET NUMBER OF ERROR
007462  005237  001230                        INC     $ERTTL          ;INCERROR COUNT

007466  000000                                HALT                    ;AN ERROR HAS OCCURED AND
007470  000240                                NOP                     ;NO OUPUT TERMINAL EXISTS
007472  062716  000004                        ADD     #4,(SP)         ;FOLLOW THE PROCEEDURE IN SECTION 6.1
007476  000002                                RTI                     ;DOCUMENTATION TO SEE WHAT ERROR OCCURED


                                                      ;
                                                      ;THIS ROUTINE PRINTS "END PASS" IF ENABLED
                                                      ;

007500  105737  001107                EOPT:   TSTB    $TPFLG          ;SEE IF WE HAVE OUTPUT TERMINAL
007504  001006                                BNE     1$              ;IF NOT DON'T PRINT END OF PASS
007506  032737  002000  177570                BIT     #2000,SWR       ;SEE IF HE INHIBITED END PASS TYPEOUT
007514  001002                                BNE     1$              ;BY SETTING BIT 10 IN SWR
007516  104400                                TYPE
007520  011423                                MEOP
007522  000207                        1$:     RTS     PC              ;EXIT
```

```

 007524  001240  001242  001244  DT1:    .WORD   $ERRAD,$GDADR,$BDADR,$GDDAT,$BDDAT,0
 007532  001246  001250  000000
 007540  001240  001242  001244  DT2:    .WORD   $ERRAD,$GDADR,$BDADR,$TMP0,0
 007546  001270  000000
 007552  001240  001270  001242  DT3:    .WORD   $ERRAD,$TMP0,$GDADR,$BDADR,0
 007560  001244  000000
 007564  001240  001254  001256  DT4:    .WORD   $ERRAD,$REG0,$REG1,$REG2,$REG3,0
 007572  001260  001262  000000
 007600  001240  001270  000000  DT5:    .WORD   $ERRAD,$TMP0,0
         000000                  DT6=0
         000000                  DF1=0

                                                 ;ASCII MESSAGES

 007606  005015  042523  042116  EM1:    .ASCIZ  <15><12>/SEND-RECIEVE DATA ERROR/
 007614  051055  041505  042511
 007622  042526  042040  052101
 007630  020101  051105  047522
 007636  000122
 007640  005015  047111  052520  EM2:    .ASCIZ  <15><12>/INPUT MODULE FAILED TO INTERRUPT/
 007646  020124  047515  052504
 007654  042514  043040  044501
 007662  042514  020104  047524
 007670  044440  052116  051105
 007676  052522  052120     000
 007703     015  044412  050116  EM3:    .ASCIZ  <15><12>/INPUT MODULE INTERRUPTED AT WRONG PRIORITY/
 007710  052125  046440  042117
 007716  046125  020105  047111
 007724  042524  051122  050125
 007732  042524  020104  052101
 007740  053440  047522  043516
 007746  050040  044522  051117
 007754  052111  000131
 007760  005015  054523  052123  EM4:    .ASCIZ  <15><12>/SYSTEM INITIALIZE FAILED TO CLEAR INPUT MODULE/
 007766  046505  044440  044516
 007774  044524  046101  055111
 010002  020105  040506  046111
 010010  042105  052040  020117
 010016  046103  040505  020122
 010024  047111  052520  020124
 010032  047515  052504  042514
 010040     000
 010041     015  042012  040525  EM5:    .ASCIZ  <15><12>/DUAL ADDRESS ERROR/
 010046  020114  042101  051104
 010054  051505  020123  051105
 010062  047522  000122
 010066  005015  047111  042524  EM6:    .ASCII  <15><12>/INTERRUPTS OUT OF ORDER, SHOULD BE:/
 010074  051122  050125  051524
 010102  047440  052125  047440
 010110  020106  051117  042504
 010116  026122  051440  047510
 010124  046125  020104  042502
 010132     072
```

```
010133     015 044412 052116          .ASCIZ  <15><12>/INTER3>INTER4>INTER1>INTER2/
010140  051105 037063 047111
010146  042524 032122 044476
010154  052116 051105 037061
010162  047111 042524 031122
010170     000
010171     015 044412 044516  EM7:    .ASCIZ  <15><12>/INITIALIZE FAILED TO CLEAR INTR, ENABLE BIT/
010176  044524 046101 055111
010204  020105 040506 046111
010212  042105 052040 020117
010220  046103 040505 020122
010226  047111 051124 020056
010234  047105 041101 042514
010242  041040 052111    000
010247     015 047012 020117  EM11:   .ASCIZ  <15><12>/NO CONNECTIONS MADE/
010254  047503 047116 041505
010262  044524 047117 020123
010270  040515 042504    000
010275     015 047012 020117  EM12:   .ASCIZ  <15><12>/NO INPUT MODULE ADDR, ENTERED/
010302  047111 052520 020124
010310  047515 052504 042514
010316  040440 042104 027122
010324  042440 052116 051105
010332  042105    000
010335     015 047012 020117  EM13:   .ASCIZ  <15><12>/NO OUTPUT MODULE ADDR, ENTERED/
010342  052517 050124 052125
010350  046440 042117 046125
010356  020105 042101 051104
010364  020056 047105 042524
010372  042522 000104
010376  005015 042526 052103  EM14:   .ASCIZ  <15><12>/VECTOR ADDR, NOT ENTERED FOR INPUT MODULE(S) IN CONNECTION(S)/
010404  051117 040440 042104
010412  027122 047040 052117
010420  042440 052116 051105
010426  042105 043040 051117
010434  044440 050116 052125
010442  046440 042117 046125
010450  024105 024523 044440
010456  020116 047503 047116
010464  041505 044524 047117
010472  051450 000051
010476  005015 051105 047522  DH1:    .ASCII  <15><12>/ERROR   ADDR    ADDR    DATA    DATA/
010504  020122 020040 042101
010512  051104 020040 020040
010520  042101 051104 020040
010526  020040 040504 040524
010534  020040 020040 040504
010542  040524
```

E4

```
010544  005015  041520  020040           .ASCIZ  <15><12>/PC      OUT     IN      EXP'D   IN/
010552  020040  020040  052517
010560  020124  020040  020040
010566  047111  020040  020040
010574  020040  054105  023520
010602  020104  020040  047111
010610     000
010611     015  042412  051122  DH2:     .ASCII  <15><12>/ERROR   ADDR    ADDR    PROS/
010616  051117  020040  040440
010624  042104  020122  020040
010632  040440  042104  020122
010640  020040  050040  047522
010646     123
010647     015  050012  020103           .ASCIZ  <15><12>/PC      OUT     IN      STAT/
010654  020040  020040  047440
010662  052125  020040  020040
010670  044440  020116  020040
010676  020040  051440  040524
010704  000124
010706  005015  051105  047522  DH3:     .ASCII  <15><12>/ERROR   ADDR    ADDR    ADDR/
010714  020122  020040  042101
010722  051104  020040  020040
010730  042101  051104  020040
010736  020040  042101  051104
010744  005015  041520  020040           .ASCIZ  <15><12>/PC      OUT     IN      DUAL/
010752  020040  020040  052517
010760  020124  020040  020040
010766  047111  020040  020040
010774  020040  052504  046101
011002     000
011003     015  042412  051122  DH4:     .ASCII  <15><12>/ERROR   INTER1  INTER2  INTER3/
011010  051117  020040  044440
011016  052116  051105  020061
011024  044440  052116  051105
011032  020062  044440  052116
011040  051105     063
011043     015  050012  020103           .ASCIZ  <15><12>/PC      ADDR    ADDR    ADDR/
011050  020040  020040  040440
011056  042104  020122  020040
011064  040440  042104  020122
011072  020040  040440  042104
011100  000122
011102  005015  051105  047522  DH5:     .ASCII  <15><12>/ERROR   ADDR/
011110  020122  020040  042101
011116  051104
011120  005015  041520  020040           .ASCIZ  <15><12>/PC      INTR/
011126  020040  020040  047111
011134  051124     000
011137     015  050012  047522  DH7:     .ASCIZ  <15><12>/PROGRAM NOT RUNNING  /
011144  051107  046501  047040
011152  052117  051040  047125
011160  044516  043516  020040
011166     000
011167     040  057040  000103  MCONC:   .ASCIZ  /  ^C/
```

G4

```
011174  000134                          MBS:    .ASCIZ  /\/
011176  005015     000                  MCRLF:  .ASCIZ  <15><12>//
011201     015  047412  042104          MODADR: .ASCIZ  <15><12>/ODD ADDRESS-NOT ACCEPTED/
011206  040440  042104  042522
011214  051523  047055  052117
011222  040440  041503  050105
011230  042524  000104
011234  005015  047515  052504          MMHD:   .ASCIZ  <15><12>/MODULE NO  ADDRESS  FUNCTION  VECTOR  CONNECTED TO/
011242  042514  047040  020117
011250  040440  042104  042522
011256  051523  020040  052506
011264  041516  044524  047117
011272  020040  042526  052103
011300  051117  020040  047503
011306  047116  041505  042524
011314  020104  047524     000
011321     015  020012  020040          MBSP:   .ASCIZ  <15><12>/        /
011326  020040  020040  000040
011334  020040  000040                  M3SP:   .ASCIZ          /   /
011340  020040  047111  052520          MFUN:   .ASCIZ          /  INPUT     /
011346  020124  020040  020040
011354     000
011355     040  047440  052125          MFUNC2: .ASCIZ          !  OUTPUT      N/A   !
011362  052520  020124  020040
011370  020040  047040  040457
011376  020040  000040
011402  005015  000052                  MSTAR:  .ASCIZ  <15><12>/*/
011406  005015  052522  047116          MRUN:   .ASCIZ  <15><12>/RUNNING.../
011414  047111  027107  027056
011422     000
011423     015  042412  042116          MEOP:   .ASCIZ  <15><12>/END PASS  /
011430  050040  051501  020123
011436  000040
011440  005015  040520  051523          MSUM:   .ASCIZ  <15><12>/PASSES   ERRORS/<15><12>
011446  051505  020040  042440
011454  051122  051117  006523
011462  000012
011464  005015  042522  052524          POWER:  .ASCIZ  <15><12>/RETURN TO MONITOR FROM POWER FAILURE/
011472  047122  052040  020117
011500  047515  044516  047524
011506  020122  051106  046517
011514  050040  053517  051105
011522  043040  044501  052514
011530  042522     000
        011534                          .EVEN
011534  000000                          TTYINP: 000000
011536  005015                          UNKINW: .ASCII  <15><12>//
011540  000000                          TTYINB: 000000
        011642                                  .=.+100
        000001                          .END
```

114

```
AINT       003214       AINTER     003444       BITO     = 000001       BIT00    = 000001
BIT01    = 000002       BIT02    = 000004       BIT03    = 000010       BIT04    = 000020
BIT05    = 000040       BIT06    = 000100       BIT07    = 000200       BIT08    = 000400
BIT09    = 001000       BIT1     = 000002       BIT10    = 002000       BIT11    = 004000
BIT12    = 010000       BIT13    = 020000       BIT14    = 040000       BIT15    = 100000
BIT2     = 000004       BIT3     = 000010       BIT4     = 000020       BIT5     = 000040
BIT6     = 000100       BIT7     = 000200       BIT8     = 000400       BIT9     = 001000
BPTVEC   = 000014       CFLG       004330       DATAR      002042       DATASC     001702
DATA0      002114       DATA1      001764       DESIFR     004030       DF1      = 000000
DH1        010476       DH2        010611       DH3        010706       DH4        011003
DH5        011102       DH7        011137       DISPLA   = 177570       DT1        007524
DT2        007540       DT3        007552       DT4        007564       DT5        007600
DT6      = 000000       DUAL       003072       DUALT      003126       DUAL1      003064
EEDNH      007416       EMTVEC   = 000030       EM1        007606       EM11       010247
EM12       010275       EM13       010335       EM14       010376       EM2        007640
EM3        007703       EM4        007760       EM5        010041       EM6        010066
EM7        010171       EOPT       007500       ERRVEC   = 000004       INADR      004332
INADRI     004504       INADRO     004572       INIT       002730       INITR      003044
INMAP      005334       INMAP1     005362       INNOR      005036       INTPO      004304
INTPO1     004316       INTRS1     003476       INTRS2     003510       INTRS3     003522
INTRS4     003534       INUMB      004372       INUNC      005100       INVET      004660
INVETT     005024       INVET2     005026       IOTVEC   = 000020       MAP1       005474
MBS        011174       MCONC      011167       MCRLF      011176       MEOP       011423
MFUN       011340       MFUNC2     011355       MMHD       011234       MODADR     011201
MOD1A      004240       MOD1V      004260       MOD2A      004242       MOD2V      004262
MOD3A      004244       MOD3V      004264       MOD4A      004246       MOD4V      004266
MOD5A      004250       MOD5C      004270       MOD6A      004252       MOD6C      004272
MOD7A      004254       MOD7C      004274       MOD8A      004256       MOD8C      004276
MRUN       011406       MSTAR      011402       MSUM       011440       M3SP       011334
M8SP       011321       NINP       004500       NINPT      004502       NTF0       004510
NTF1       004576       NTF2       007274       NTF3       007304       NTF3A      007312
NTF3B      004714       NTF4       007316       NTH        007200       NTHF       007252
NTHFP      007254       NUMBER     004000       ODDADR     004226       PC       =%000007
POPSP2   = 022626       POWER      011464       PS       = 177776       PSW      = 177776
PWRVEC   = 000024       RESVEC   = 000010       RUBF       003776       RUBH       003724
R0       =%000000       R1       =%000001       R2       =%000002       R3       =%000003
R4       =%000004       R5       =%000005       R6       =%000006       R7       =%000007
SFIVE      004302       SINCO      005130       SINGLE     001632       SINGLF     001676
SINT       002230       SINTR      002310       SINT4      002320       SINT4R     002402
SINT5      002412       SINT5R     002532       SINT6      002552       SINT6R     002632
SINT7      002644       SINT7R     002716       SP       =%000006       STACK    = 001100
STAR       005542       START      001446       START1     001476       START2     001512
START3     001526       START4     001550       START5     001572       SUM        005476
SWR      = 177570       SW0      = 000001       SW00     = 000001       SW01     = 000002
SW02     = 000004       SW03     = 000010       SW04     = 000020       SW05     = 000040
SW06     = 000100       SW07     = 000200       SW08     = 000400       SW09     = 001000
SW1      = 000002       SW10     = 002000       SW11     = 004000       SW12     = 010000
SW13     = 020000       SW14     = 040000       SW15     = 100000       SW2      = 000004
SW3      = 000010       SW4      = 000020       SW5      = 000040       SW6      = 000100
SW7      = 000200       SW8      = 000400       SW9      = 001000       S15        005030
S16        005032       S17        005034       TBITVE   = 000014       TRAPVE   = 000034
TRTVEC   = 000014       TTYIN      003546       TTYINB     011540       TTYINP     011534
TYPE     = 104400       TYPOC    = 104402       TYPON    = 104406       TYPOS    = 104404
UNKINP     004164       UNKINW     011536       VECTOR     004300       $BDADR     001244
```

Ta

```
MAINDEC-11-DZKHA-A       MACY11,624  28-JAN-74  10:08  PAGE 39
DZKHA,SRC         SYMBOL TABLE

$BDDAT    001250          $CM1    = 000006          $CM2    = 000014          $CM3    = 000006
$CM4    = 000004          $CRLF     001303          $DOAGN    006030          $ENDAD    006020
$ENDCT    006004          $EOP      005762          $ERFLG    001221          $ERRAD    001240
$ERROR    006212          $ERRTB    001306          $ERRTY    006302          $ERTTL    001230
$FILLC    001106          $FILLS    001105          $GDADR    001242          $GDDAT    001246
$GET42    006012          $HD     = 000003          $HINUM    007010          $ICNT     001222
$ILLUP    007172          $ITEMB    001236          $LF       001304          $LONUM    007012
$LPADR    001224          $LPERR    001226          $MXCNT    006210          $NULL     001104
$OCNT     006660          $OMODE    006662          $OVER     006174          $PASS     001216
$PWRDN    007044          $PWRUP    007112          $QUES     001302          $RAND     006664
$REGAD    001252          $REG0     001254          $REG1     001256          $REG2     001260
$REG3     001262          $REG4     001264          $REG5     001266          $SAVR6    007176
$SCOPE    006034          $SETUP  = 000017          $SS     = 000001          $STUP   = 177777
$SVLAD    006164          $SWR    = 164000          $TIMES    001300          $TKB      003722
$TKS      003720          $TMP0     001270          $TMP1     001272          $TMP2     001274
$TMP3     001276          $TN     = 000001          $TPB      001102          $TPFLG    001107
$TPS      001100          $TRAP     007014          $TRP    = 000010          $TRPAD    007034
$TSTNM    001220          $TYPE     001110          $TYPOC    006462          $TYPON    006476
$TYPOS    006436          $XTSTR    006042          $OFILL    006661          .       = 011642


 ERRORS DETECTED:  0
 *DZKHA,DZKHA/SOL_DZKHA,SRC
 RUN-TIME: 21 13 0 SECONDS
 CORE USED:  12K
```

K4