IP11/IP300

PROCESS CONTROL SUBSYSTEMS
MD-11-DVPCA-A

EP-DVPCA-A-DL-A

FICHE 1 OF 1

DEC 1977
digital
MADE IN USA

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DVPCA-A-D

PRODUCT NAME: PROCESS CONTROL SUBSYSTEMS - FIELD TEST

PRODUCT DATE: NOVEMBER 1977

MAINTAINER: DIAGNOSTIC ENGINEERING

. MAIN. MACY11 30(1046) 25-OCT-77 10:10
DVPCAA.P11 25-OCT-77 10:02 TABLE OF CONTENTS SEQ 0002

```
                        .SBTTL  DOCUMENT
                        ;;*****************************************************************
                        .REM  -
                        1.0             ABSTRACT
                        *****************************************************

                                THIS PROGRAM IS A DIAGNOSTIC TOOL FOR TESTING THE
                                ENTIRE FAMILY OF PCS MODULES.  THE PROGRAM IS INTENDED TO BE
                                USE BY FIELD ENGINEERING AND MANUFACTURING .
                                BY USING THIS PROGRAM AN OPERATOR IS ABLE TO CHECK THE 10CM,
                                AND ANY DIGITAL OR ANALOG MODULE.  SINCE THE ASSUMPTION
                                IS MADE THAT ALL OUTPUT MODULES ARE CONNECTED TO THE
                                CUSTOMERS WIRING DURING THE TEST OF OUTPUT MODULES THE
                                DISABLE BIT IS SET.  THIS DIAGNOSTIC WILL NOT CHECK THE
                                PART OF HARDWARE THAT IS INTERFACING DIGITAL OR ANALOG
                                MODULES WITH CUSTOMERS WIRING (DRIVING TRANSISTOR, ISOLATING
                                TRANSFORMERS, PHOTO-COUPLERS ECT).  THIS DIAGNOSTIC DOES
                                NOT REQUIRE ANY EXTERNAL DEVICES OR SPECIAL CONNECTIONS.

                        2.0             HARDWARE REQUIREMENTS
                        *****************************************************

                                1.  LSI 11 WITH 16K OF MEMORY OR ANY PDP11 CPU WITH
                                    UNIBUS BRIDGE INTERFACE.
                                2.  CONSOLE TERMINAL
                                3.  FLOPPY DISC OR SOME OTHER INPUT DEVICE
                                4.  10CM (M7458)

                        3.0             PROGRAM CONSIDERATION
                        *****************************************************

                        3.1             CPU COMPATIBILITY
                                THIS PROGRAM CAN BE USED BY THE LSI 11 OR BY THE PDP-11
                                /04,05,10,20,34,35,40,45,50 & 70 IF UNIBUS BRIDGE MODULE
                                IS USED.

                        3.2             XXDP

                                THIS PROGRAM CAN BE CHAINED BY XXDP & WILL NOT OVERLAY
                                THE LOADER

                                CHAIN MODE OPERATION

                                1.  THE INPUT DIALOGUE WITH AN OPERATOR IS BYPASSED
                                2.  THE SUBSYSTEM IS MAPPED IN MEMORY
                                3.  THE SYSTEM TEST IS AUTOMATICALLY EXECUTED

                                NORMAL OPERATION
                                1.  START AT LOC 204
                                2.  SELECT TEST
                                3.  PROGRAM RUNS AND GOES BACK TO MONITOR

                        3.3     ACT/APT

                                THIS PROGRAM IS ACT COMPATIBLE TO THE EXTENT THAT APT HOOKS
```

```
                        WILL BE IN THE PROGRAM & WILL WORK THRU THE "UPTON INTERFACE"



3.4     EXECUTION TIME

        MOST INDIVIDUAL TESTS TAKE LESS THEN 1 SEC. THE ONESHOT MODULE TEST
        TAKES 15 SEC.  THE SYSTEM TEST EXECUTION TIME DEPENDS
        ON THE NUMBER  AND TYPE OF I/O MODULES.

3.5     DEFAULT ADDRESSES & VECTORS

        THE FOLLOWING IS A LIST OF THE DEFAULT ADDRESSES & VECTORS
        OF ALL HARDWARE TO BE USED AND THEIR CORRESPONDING MEMORY LOCATIONS.
        IN THE CASE THAT THE IOCM IS SET TO AN ADDRESS OTHER THAN
        171000 THESE LOCATIONS MUST BE CHANGED.
                BASE:    .WORD   171000  ;FIRST ADDRESS OF I/OADDRESS BLOCK
                CSR:     .WORD   171377  ;ADDRESS OF IOCM CSR REGISTER
                IAR:     .WORD   171376  ;      "          IAR REGISTER
                VECTO:   .WORD   234     ;INTERRUPT VECTOR OF IOCM
                VECTOA:  .WORD   236     ;INTERRUPT VECTOR+2 OF IOCM

4.0     OPERATING PROCEDURE
********************************************************************

4.1     PROGRAM LOADING

        THE PROGRAM CAN BE LOADED FROM PAPER TAPE OR FLOPPY
        DISK USING STANDARD PROCEDURES.

4.2     PROGRAM STARTING

        LOCATION        200 - STARTING ADDRESS TO RUN THE DIAGNOSTICS MONITOR.
                              THE PROGRAM WILL PRINT OPTIONS AVAILABLE TO
                              THE OPERATOR.

        LOCATION        204 - STARTING ADDRESS OF MONITOR WITHOUT PRINTING
                              OPERATOR'S OPTIONS.

4.3     INPUT DIALOGUE

        IF AN OPERATOR STARTS THE PROGRAM AT LOCATION 204  HE CAN
        SELECT ONE OF FOLLOWING OPTIONS:


4.3.1   S-SYSTEM TEST

        THIS OPTION OF THE PROGRAM WILL MAP ALL THE ANALOG
        AND DIGITAL MODULES CONNECTED TO THE IOCM AND BUILD
        A TABLE OF THEM IN MEMORY.  THEN IT WILL PICK UP
        EACH ONE AND RUN THE APPROPRIATE TESTS.  IF SWITCH 14 IN
        THE SOFTWARE SWITCH REGISTER IS SET , IT WILL LOOP ON
        CURRENT TEST UNTIL OPERATOR TYPES CONTROL C.  AFTER THE  SELECTED # OF
        PASSES, THE PROGRAM WILL PRINT PASS COUNT UNLESS SWITCH 13 IS SET.

4.3.2   M - MAP OF DBUS
```

THIS OPTION ALLOWS AN OPERATOR TO CHECK WHICH I/O MODULE
ARE CONNECTED TO THE IOCM. IT CHECKS ALL ADDRESSES BETWEEN
171000 AND 171375. FOR ADDRESSES THAT ANSWER , IT CHECKS THE
GENERIC CODE AND TYPES THE ADDRESS AND INTERFACE TYPE.

4.3.3 D - TEST DIGITAL MODULE

THIS TEST WILL EXERCISE THE DIGITAL MODULE AT THE ADDRESS
SPECIFIED BY THE OPERATOR. FOLLOWING IS A DESCRIPTION OF
THE DIGITAL MODULE TESTS FOR EACH MODULE TYPE..
EACH TEST IS RUN AS MANY TIMES AS SELECTED BY THE L OPTION
AND THE PASS COUNT IS PRINTED.

M5010:

1. SET D BIT
2. CHECK THAT ALL BITS ARE ZERO
3. SET T BIT
4. CHECK THAT ALL BITS ARE ONES
5. SET C BIT

M5011:

1. SET DBIT AND TBIT
2. CHECK IF INPUTS ARE ALL ONES
3. CLEAR TBIT
4. CHECK IF INPUTS ARE ALL ZEROS
5. CLEAR ALL INTERRUPTS
6. SET TBIT
7. ENABLE INTERRUPT
8. CHECK IF INTERRUPT OCCURRED
9. CHECK IF INPUTS ARE ALL ONES
10. CHECK IF COS REGISTERS ARE ALL ONES
11. SET RIF BIT
12. CLEAR ALL COS REGISTERS
13. CHECK IF THEY ARE CLEAR
14. CLEAR T BIT
15. CHECK IF INTERRUPT OCCURED
16. CHECK IF ALL INPUTS ARE ZERO
17. RUN STEP 10 TO 13
18. CLEAR ALL INTERRUPTS

M5012:

1. SET DBIT
2. CHECK IF ALL BITS ARE ZERO
3. SET TBIT
4. CHECK IF ALL BITS ARE ONE
5. CLEAR T BIT
6. CLEAR ALL INTERRUPTS
7. SET T BIT
8. ENABLE INTERRUPT
9. CHECK IF INTERRUPT OCCURRED

```
                                        10. SET  RIF BIT
                                        11. CLEAR INTERRUPT
                                        12. CHECK IF INTERRUPTS ARE CLEAR
                                        13. CLEAR T BIT
                                        14. RUN STEP 9 TO 12
                                        15. CLEAR ALL INTERRUPTS


                                5013:

                                        1.  SET DBIT
                                        2.  CHECK IF ALL BITS ARE ZERO
                                        3.  SET TBIT
                                        4.  TEST IF ALL BITS ARE ONE
                                        5.  CLEAR T BIT
                                        6.  CLEAR ALL INTERRUPTS
                                        7.  SET T BIT
                                        8.  ENABLE INTERRUPT
                                        9.  CHECK IF INTERRUPT OCCURED
                                        10. SET RIF BIT
                                        11. CLEAR INTERRUPT
                                        12. CHECK IF INTERRUPT IS CLEAR
                                        13. CLEAR T BIT
                                        14. RUN STEP 9 TO 12
                                        15. CLEAR ALL INTERRUPTS

                                M6010:

                                        1.  SET D BIT
                                        2.  CLEAR ALL 4 BYTES OF I/O REGISTER
                                        3.  SET DATA PATTERN 125 IN FIRST BYTE
                                        4.  CHECK IF IT IS SET
                                        5.  CHECK IF OTHER BYTES ARE ZERO
                                        6.  DO THE SAME WITH DATA 252 , 377 , 000
                                        7.  DO THE SAME WITH OTHER BYTES

                                M6011:
                                        THE LINE CLOCK MUST BE ENABLE

                                        1.      SET D BIT
                                        2.      SET DATA PATTERN 252 AT MUT
                                        3.      CHECK IF IT IS SET
                                        4.      WAIT UP TO 10 SEC
                                        5.      DATA SHOULD BE CLEAR
                                        6.      REPEAT STEP 2 TO 5 WITH DATA EQUAL TO 125

                                M6012:
                                M6013:

                                        1.  SET D BIT
                                        2.  SET OUTPUT REGISTER TO FOLLOWING DATA PATTERN
                                            0,377,252,125
                                        3.  CHECK IF IT IS SET
```

4.3.4 I - TEST IOCM

THIS TEST WILL EXERCISE ALL THE FEATURES OF THE IOCM. THE FOLLOWING
IS A DESCRIPTION OF ALL THE TESTS.
THIS TEST IS RUN AS MANY TIMES AS SELECTED AND THE PASS COUNT IS TYPED

TST1: CHECKS IF EACH BIT OF THE IOCM IS CLEARED BY THE CBIT

TST2: CHECKS IF THE INTERRUPT ENABLE BIT (EBIT) CAN BE SET AND CLEARED

TST3: CHECKS IF THE MAINTENANCE BIT (MBIT) CAN BE SET AND CLEARED

TST4: CHECKS IF THE DISABLE BIT (DBIT) CAN BE SET AND CLEARED
CHECKS IF DBIT GENERATES CLEAR

TST5: CHECKS IF THE TEST BIT (TBIT) CAN BE SET AND CLEARED

TST6: CHECKS IF THE GENERIC CODE BIT (GBIT) CAN BE SET AND CLEARED

TST7: CHECKS IF THE RIF BIT (RBIT) CAN BE SET AND CLEARED

TST10: CHECKS DBUS DATA PATHS IN A MAINTENANCE
MODE. IF THE MBIT IS SET AND THE CPU ADDRESSES ANY LOCATION
BETWEEN 171000 AND 171375 IT SHOULD READ BACK THE LOWER BYTE
OF THE MODULE ADDRESS.

TST11: CHECKS MAINTENANCE INTERRUPT. IF THE MBIT & EBIT
ARE SET, THE IOCM WILL GENERATE AN INTERRUPT WITH VECTOR 234.
THE IAR (171376) HAS THE LOWER BYTE OF CSR ADDRESS (377).

4.3.5 W -WRAP AROUND TEST

THIS OPTION ALLOWS THE FIELD ENGINEER TO CONNECT MODULE M6010 TO
EITHER M5010 OR M5011. THEN IT WRAPS AROUND A SET OF DATA PATTERNS(125,252,377,0
IF SWITCH 14 IN THE SOFTWARE SWITCH REGISTER IS SET , IT WILL
LOOP ON THIS TEST UNTIL THE OPERATOR TYPES CONTROL C. EVERY SELECTED # OF
PASSES IT WILL PRINT PASS COUNT UNLESS SWITCH 13 IS SET.
ERRORS WILL BE PRINTED INDICATING GOOD DATA, BAD DATA AND THE PASS # AT WHICH
IT OCCURRED.

4.3.6 F -FIELD TEST

THE PURPOSE OF THIS TEST IS:
TO OUTPUT ANY SELECTED DATA PATTERN TO AN OUTPUT
MODULE SPECIFIED BY OPERATOR.
B. TO MONITOR DATA FROM AN INPUT MODULE SPECIFIED BY OPERATOR.

WARNING: THE FIELD ENGINEER MUST REALIZE THAT THIS IS THE ONLY TEST
THAT PERMITS HIM TO OUTPUT DATA TO THE CUSTOMER'S WIRING.
THERE WILL BE VOLTAGE APPLIED TO CUSTOMER EQUIPMENT CONNECTED
TO THE OUTPUT MODULE UNDER TEST.
THEREFORE PRECAUTIONS MUST BE TAKEN THAT AN ERRONEOUS OUTPUT WILL
NOT CAUSE DAMAGE TO THE FIELD EQUIPMENT AND THAT ALL TESTING IS
CONDUCTED WITH THE CUSOMERS KNOWLADGE

```
                                PROCEDURE:
                                        USER SELECTS THE ADDRESS OF THE MODULE UNDER TEST.
                                        IF IT IS AN OUTPUT MODULE THEN:

                                        USER SELECTS AND OUTPUTS A DATA PATTERN, ONE BYTE AT A TIME,
                                        TO THE MODULE. TYPE CONTROL/C TO ABORT TEST.
                                        AFTER ALL BYTES OF MODULE HAS BEEN SELECTED, USER MUST TYPE CONTROL/C
                                        TO GO BACK TO MONITOR.

                                EXEMPLE:

                                TYPE ADDRESS OF MUT   171XXX
                                TYPE ONE BYTE DATA PATTERN   YYY
                                (WHERE YYY IS A 8 BIT DATA PATTERN TO OUTPUT.
                                 ALSO, THE REQUEST FOR DATA FOR OUTPUT WILL BE REPEATED UP TO
                                 4 TIMES DEPENDENT UPON MODULE TYPE)

                                 IF IT IS AN INPUT MODULE THEN:

                                 THE TEST MONITORS AND PRINTS DATA FROM MODULE INPUTS.
                                 IT PRINTS EVERY BYTE ADDRESS OF THE MODULE WITH ITS
                                 CONTENT DURING THE FIRST PASS AND IT CONTINUES TO
                                MONITOR THE DATA WITHOUT PRINTING IT UNLESS A CHANGE
                                IN THE DATA OCCURED. MUST TYPE CONTROL/C TO RETURN TO MONITOR..

                                EXEMPLE:

                                TYPE ADDRESS OF MUT   171XXX
                                171XXX :   YYY
                                (WHERE YYY IS THE 8 BITS OF DATA READ FROM THE MODULE.
                                 THE SECOND LINE OF THE MESSAGE WILL BE REPEATED
                                 UP TO 4 TIMES, WITH INCREMENTING ADDRESS,
                                 DEPENDING UPON MODULE TYPE.

                        4.3.7   L - SET PASS COUNT

                                TYPE THE OCTAL NUMBER OF ITERATIONS FOR ALL TESTS.
                                DEFAULT NUMBER IS 1.
                                MAXIMUM NUMBER IS 177777

                        4.3.7   T - SET SOFTWARE SWITCH REGISTER

                                TYPE THE OCTAL NUMBER TO BE LOADED TO SWREG

                                100000 - HALT ON ERROR
                                40000  - LOOP ON TEST
                                20000  - INHIBIT ERROR AND END OF PASS PRINT
                                10000  - LOOP ON ERROR


                        4.3.8   CONTROL C

                                TYPING CONTROL C CAUSES THE PRESENT TEST TO BE ABORTED
                                AND PROGRAM RETURNS TO THE MONITOR.
                                IN SOME CASES IT MAY TAKE UP TO 20 SECONDS FOR THIS TO
                                HAPPEN.
```

```
                                        .SBTTL  TRAP CATCHER
                                
                                                .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ". +2, HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                                                .=174
000174  000000                  DISPREG:  .WORD  0              ;;SOFTWARE DISPLAY REGISTER
000176  000000                  SWREG:    .WORD  0              ;;SOFTWARE SWITCH REGISTER
                                        .SBTTL  STARTING ADDRESS(ES)
000200  000137  002362                  JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
                                .=100
000100  012350                  NOCLK
                                .=102
000102  000340                  #PR7
                                .=42
000042  000000                          HALT
                                .=204
000204  000137  002640                  JMP     SETCLK
                                .=1100
                                .SBTTL  BASIC DEFINITIONS

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

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

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

                                ;*PRIORITY LEVEL DEFINITIONS
000000                          PR0=    0               ;;PRIORITY LEVEL 0
```

```
                        PR1=    40              ;;PRIORITY LEVEL 1
                        PR2=    100             ;;PRIORITY LEVEL 2
                        PR3=    140             ;;PRIORITY LEVEL 3
                        PR4=    200             ;;PRIORITY LEVEL 4
                        PR5=    240             ;;PRIORITY LEVEL 5
                        PR6=    300             ;;PRIORITY LEVEL 6
                        PR7=    340             ;;PRIORITY LEVEL 7

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

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
```

```
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        .SBTTL  BIT DEFINITIONS
000200                  FBIT=BIT7                       ;FLAG BIT
000100                  EBIT=BIT6                       ;INTERRUPT ENABLE
000040                  MBIT=BIT5                       ;MAINTENANCE INTERRUPT
000020                  DBIT=BIT4                       ;I/O DISABLE BIT
000010                  TBIT=BIT3                       ;TEST BIT, INVERTS I/O BITS
000004                  GBIT=BIT2                       ;GENERIC CODE ENEBLE
000002                  CBIT=BIT1                       ;CLEAR I/O
000001                  RBIT=BIT0                       ;RIF BIT, CLEARS I/O INTERRUPT
```

```
                        .SBTTL  COMMON TAGS

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

            001100              .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102     000          $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103     000          $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114     000          $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115     001          $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134     000          $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135     000          $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154     000          $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155     002          $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156     012          $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157     000          $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001162  000000          $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000          $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001166  000000          $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001170  000000          $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001172  000000          $ESCAPE: 0                      ;;ESCAPE ON ERROR ADDRESS
001174     077          $QUES:  .ASCII  /?/             ;;QUESTION MARK
001175     015          $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001176  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                        ;;**********************************************************************
001200  000120          ATABL:  .BLKW   80.             ;TABLE A=ADDRESSES OF I/O MODULES
001440  000120          BTABL:  .BLKW   80.             ;TABLE B=GENERIC CODES OF THIS MODULES
001700  000141          GENER:  .WORD   141             ;GENERIC CODE FOR NONISOLATED DC 32 BITS IN
001702  000121                  .WORD   121             ;GENERIC CODE FOR NONISOLATED DC 16 BITS IN
001704  000122                  .WORD   122             ;GENERIC CODE FOR ISOLATED DC 16 BITS IN
001706  000101                  .WORD   101             ;GENERIC CODE FOR AC 8 BITS IN
001710  000041                  .WORD   41              ;GENERIC CODE FOR NONISOLATED DC 32 BITS OUT
001712  000021                  .WORD   21              ;GENERIC CODE FOR ONESHOT 16 BITS OUT
```

```
   537  001714  000001                          .WORD   1               ;GENERIC CODE FOR ISOLATED DC 8 BITS OUT
   538  001716  000002                          .WORD   2               ;GENERIC CODE FOR AC 8 BITS OUT
   539  001720  000000                          .WORD   0
   540  001722  005010                  MUT:    .WORD   5010
   541  001724  005011                          .WORD   5011
   542  001726  005012                          .WORD   5012
   543  001730  005013                          .WORD   5013
   544  001732  006010                          .WORD   6010
   545  001734  006011                          .WORD   6011
   546  001736  006012                          .WORD   6012
   547  001740  006013                          .WORD   6013
   548  001742  006672                  MODUL:  .WORD   M5010           ;THIS TABLE HAS ADDRESSES OF I/O SUBRUTINE
   549  001744  010322                          .WORD   M5011
   550  001746  011152                          .WORD   M5012
   551  001750  006774                          .WORD   M5013
   552  001752  007440                          .WORD   M6010
   553  001754  010024                          .WORD   M6011
   554  001756  007632                          .WORD   M6012
   555  001760  007632                          .WORD   M6013
   556  001762  000000                          .WORD   0
   557  001764  000000                  ANSW:   .WORD   0
   558  001766  000000                  YLOOP:  .WORD   0
   559  001770  000000                  DBUFF:  .WORD   0
   560  001772  000000                          .WORD   0
   561  001774     125                  PATT:   .BYTE   125
   562  001775     252                          .BYTE   252
   563  001776     377                          .BYTE   377
   564  001777     000                          .BYTE   0
   565  002000  000042                  XXDP:   .WORD   42
   566  002002  000000                  $MUT:   .WORD   0
   567  002004  000000                  BYTNUM: .WORD   0               ;NUMBER OF BITES OF I/O
   568  002006  000000                  TADDR:  .WORD   0               ;ADDRESS OF MUT
   569  002010  000000                  TADDR1: .WORD   0
   570  002012  000000                  TBADDR: .WORD   0
   571  002014  000000                  COSADR: .WORD   0               ;SECEND ADDRESS OF COS MODULE
   572  002016  000000                  DMUT:   .WORD   0               ;DIGITAL MODULE UNDER TEST
   573  002020  000000                  LONGIN: .WORD   0
   574  002022  000000                  TNUMB:  .WORD   0
   575  002024  177546                  CLKADR: .WORD   177546
   576  002026  171377                  CSR:    .WORD   171377
   577  002030  171376                  IAR:    .WORD   171376
   578  002032  000234                  VECTO:  .WORD   234
   579  002034  000236                  VECTOA: .WORD   236
   580  002036  000000                  VECT1:  .WORD   0
   581  002040  000000                  VECT1A: .WORD   0
   582  002042  171000                  BASE:   .WORD   171000          ;FIRST I/O
   583  002044  000000                  CLK:    .WORD   0
   584  002046  000000                  TEMP:   .WORD   0
   585  002050  000000                  TEMP1:  .WORD   0
   586  002052  000000                  TEMP3:  .WORD   0
   587  002054  000000                  TEMP4:  .WORD   0
   588  002056  000000                  PASS:   .WORD   0
   589  002060  000001                  PASCNT: .WORD   1
   590  002062  000100                  CLKVC:  .WORD   100
   591  002064  000102                  CLKVCA: .WORD   102
   592  002066  011640                  MOD:    .WORD   F5010
```

```
  593  002070  011640                          .WORD   F5011
  594  002072  011652                          .WORD   F5012
  595  002074  011664                          .WORD   F5013
  596  002076  011676                          .WORD   F6010
  597  002100  011732                          .WORD   F6011
  598  002102  011716                          .WORD   F6012
  599  002104  011716                          .WORD   F6013
  600  002106  000000                          .WORD   0
  601  002110  000060                KBVEC:    .WORD   60
```

```
                                .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 $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

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


002112                          $ERRTB:
002112  016230                          EM1
002114  016340                          DH1
002116  022024                          DT1
002120  000000                          0
002122  016257                          EM2
002124  016340                          DH1
002126  022024                          DT1
002130  000000                          0
002132  016371                          EM3
002134  000000                          0
002136  000000                          0
002140  000000                          0
002142  016422                          EM4
002144  016340                          DH1
002146  022024                          DT1
002150  000000                          0
002152  016450                          EM5
002154  016470                          DH5
002156  022040                          DT5
002160  000000                          0
002162  016523                          EM6
002164  016340                          DH1
002166  022024                          DT1
002170  000000                          0
002172  016572                          EM7
002174  016607                          DH7
002176  022004                          DT7
002200  000000                          0
002202  016657                          EM10
002204  000000                          0
002206  000000                          0
002210  000000                          0
002212  016712                          EM11
002214  016607                          DH7
002216  022004                          DT7
002220  000000                          0
002222  016750                          EM12
002224  016470                          DH5
002226  022040                          DT5
002230  000000                          0
002232  017000                          EM13
```

```
  658  002234  000000                    0
  659  002236  000000                    0
  660  002240  000000                    0
  661  002242  017046                    EM14
  662  002244  000000                    0
  663  002246  000000                    0
  664  002250  000000                    0
  665  002252  017111                    EM15
  666  002254  000000                    0
  667  002256  000000                    0
  668  002260  000000                    0
  669  002262  017154                    EM16
  670  002264  000000                    0
  671  002266  000000                    0
  672  002270  000000                    0
  673  002272  016572                    EM17
  674  002274  016607                    DH7
  675  002276  021734                    DT17
  676  002300  000000                    0
  677  002302  017217                    EM20
  678  002304  016607                    DH7
  679  002306  021754                    DT20
  680  002310  000000                    0
  681  002312  017251                    EM21
  682  002314  017313                    DH21
  683  002316  022054                    DT21
  684  002320  000000                    0
  685  002322  017337                    EM22
  686  002324  000000                    0
  687  002326  000000                    0
  688  002330  000000                    0
  689  002332  017413                    EM23
  690  002334  017450                    DH23
  691  002336  021774                    DT23
  692  002340  000000                    0
  693  002342  017541                    EM24
  694  002344  000000                    0
  695  002346  000000                    0
  696  002350  000000                    0
  697  002352  020356                    EM25
  698  002354  000000                    0
  699  002356  000000                    0
  700  002360  000000                    0
```

```
                                        START:
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
002362  012706  001100                          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
002366  005026                                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
002370  022706  001140                          CMP     #SWR,R6 ;;DONE?
002374  001374                                  BNE     .-6             ;;LOOP BACK IF NO
002376  012706  001100                          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
002402  012737  013670  000020                  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
002410  012737  000340  000022                  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
002416  012737  014514  000030                  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
002424  012737  000340  000032                  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
002432  012737  015770  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002440  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
002446  013737  004264  004256                  MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
002454  005037  001172                          CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002460  112737  000001  001115                  MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
002466  012737  002466  001110                  MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER.  IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
002474  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002500  012737  002534  000004                  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002506  012737  177570  001140                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002514  012737  177570  001142                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002522  022777  177777  176410                  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002530  001012                                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
002532  000403                                  BR      65$             ;;BRANCH IF NO TIMEOUT
002534  012716  002542          64$:            MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002540  000002                                  RTI
002542  012737  000176  001140  65$:            MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002550  012737  000174  001142                  MOV     #DISPREG,DISPLAY
002556  012637  000004          66$:            MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
```

```
                                                .SBTTL  DIAGNOSTICS MONITOR
                                                .SBTTL  TYPE PROGRAM NAME
                                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
002562  005227  177777                                  INC     #-1                     ;;FIRST TIME?
002566  001024                                        BNE     67$                     ;;BRANCH IF NO
002570  022737  004300  000042                        CMP     #$ENDAD,@#42            ;;ACT-11?
002576  001420                                        BEQ     67$                     ;;BRANCH IF YES
002600  104401  002606                                TYPE    ,68$                    ;;TYPE ASCIZ STRING
002604  000415                                        BR      67$                     ;;GET OVER THE ASCIZ
                                                ;;68$:  .ASCIZ  <CRLF>/DIGITAL IO SYSTEM TEST/<CRLF>
002640                                          67$:
002640                                          SETCLK:
002640  013746  000004                                MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
002644  012737  002662  000004                        MOV     #1$,ERRVEC
002652  052777  000100  177144                        BIS     #BIT6,@CLKADR   ;ENABLE LINE CLOCK INTERRUPT FOR UNIBUS COMPUTERS
002660  000401                                        BR      2$
002662  022626                                  1$:   CMP     (SP)+,(SP)+     ;TRAP IF LSI11
002664                                          2$:
002664  012637  000004                                MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
002670  012706  001100                                MOV     #1100,R6
002674  005777  177100                                TST     @XXDP
002700  001402                                        BEQ     4$
002702  000137  003612                                JMP     AUTO
002706                                          4$:
002706  012777  012550  177174                        MOV     #KBINT,@KBVEC
002714  152777  000100  176222                        BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
002722  012746  000000                                MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
002726  012746  002734                                MOV     #64$,-(SP)
002732  000002                                        RTI
002734  000240                                  64$:  NOP
002736  104401  017607                                TYPE    ,M13            ;TEST OPTIONS
002742  104401  017631                                TYPE    ,M14            ;S - SYSTEM TEST
002746  104401  017652                                TYPE    ,M15            ;D - DIGITAL MODULE TEST
002752  104401  017721                                TYPE    ,M17            ;M - MAP OF DBUS DEVICES
002756  104401  017774                                TYPE    ,M19            ;I - IOCM TEST
002762  104401  020022                                TYPE    ,M21            ;T - SET SWREG
002766  104401  020230                                TYPE    ,M24            ;F - FIELD TEST
002772  104401  020250                                TYPE    ,M25            ;W - WRAP AROUND TEST
002776  104401  020267                                TYPE    ,M26            ;L - LOAD LOOP COUNT
```

```
  776  003002  012706  001100          MONIT:  MOV     #1100,R6        ;SET STACK
  777  003006  142777  000100  176130          BICB    #BIT6,@$TKS     ;DISABLE KB INTERRUPT
  778  003014  104401  021114                  TYPE    ,MASS12         ;TYPE TEST OPTION TO RUN
  779  003020  104406                          RDCHR
  780  003022  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
  781  003024  110037  021315                  MOVB    R0,MASS18
  782  003030  104401  021315                  TYPE    ,MASS18         ;ECHO CHARACTER
  783  003034  022700  000123                  CMP     #123,R0         ;IS IT S?
  784  003040  001002                          BNE     1$              ;STAR AUTO TEST
  785  003042  000137  003612                  JMP     AUTO
  786  003046  022700  000104          1$:     CMP     #104,R0         ;D ?
  787  003052  001002                          BNE     2$
  788  003054  000137  003222                  JMP     DIGIT           ;START DIGITAL MODULE TEST
  789  003060  022700  000101          2$:     CMP     #101,R0         ;A ?
  790  003064  001002                          BNE     3$
  791  003066  000137  011150                  JMP     ANALOG          ;START ANALOG MODULE TEST
  792  003072  022700  000115          3$:     CMP     #115,R0         ;M ?
  793  003076  001002                          BNE     4$
  794  003100  000137  005054                  JMP     MAPE            ;MAP THE SYSTEM
  795  003104  022700  000130          4$:     CMP     #130,R0         ;X ?
  796  003110  001002                          BNE     5$
  797  003112  000137  011634                  JMP     EXERC           ;START EXERCISER
  798  003116  022700  000111          5$:     CMP     #111,R0         ;I ?
  799  003122  001002                          BNE     6$
  800  003124  000137  005302                  JMP     IOCM            ;TEST IOCM
  801  003130  022700  000124          6$:     CMP     #124,R0         ;T ?
  802  003134  001002                          BNE     7$
  803  003136  000137  003564                  JMP     SWREGS          ;SET SOFTWARE SWITCH REG.
  804  003142  022700  000106          7$:     CMP     #106,R0         ;F ?
  805  003146  001002                          BNE     8$
  806  003150  000137  004610                  JMP     FIELD           ;START FIELD TEST
  807  003154  022700  000127          8$:     CMP     #127,R0         ;W ?
  808  003160  001002                          BNE     9$
  809  003162  000137  004314                  JMP     LOOP            ;WRAP AROUND  TEST
  810  003166  022700  000114          9$:     CMP     #114,R0         ;LOAD LOOP COUNT
  811  003172  001007                          BNE     10$
  812  003174  104401  020321                  TYPE    ,M27            ;NUMBER OF ITERATIONS =
  813  003200  104410                          RDOCT
  814  003202  012637  002060                  MOV     (SP)+,PASCNT    ;;POP STACK INTO PASCNT
  815  003206  000137  003002                  JMP     MONIT
  816  003212  104401  020013          10$:    TYPE    ,M20            ;?
  817  003216  000137  003002                  JMP     MONIT           ;OPERATOR TYPED WRONG CHARACTER
```

```
                                        .SBTTL  DIGITAL MODULE MONITOR
003222                                  DIGIT:
003222  012777  012550  176660                  MOV     #KBINT,@KBVEC
003230  152777  000100  175706                  BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
003236  104401  021144                          TYPE    ,MASS13         ;TYPE ADDRESS OF MUT
003242  104410                                  RDOCT
003244  012637  002006                          MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
003250  013746  000004                          MOV     ERRVEC,-(SP)    ;SAVE ERROR VECTOR
003254  012737  003550  000004                  MOV     #5$,ERRVEC      ;SET LOC 4
003262  105777  176540                          TSTB    @CSR            ;MAKE SURE IOCM IS OK
003266  001404                                  BEQ     17$
003270  104010                                  ERROR 10                ;TEST ABORTED, IOCM ERROR
003272  012637  000004                          MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
003276  000207                                  RTS     PC
003300  112777  000004  176520          17$:    MOVB    #GBIT,@CSR      ;SET GENERIC BIT
003306  117700  176474                          MOVB    @TADDR,R0       ;R0=GENERIC CODE OF MUT
003312  012701  001742                          MOV     #MODUL,R1       ;ADDRESS OT TEST
003316  012703  001722                          MOV     #MUT,R3
003322  012702  001700                          MOV     #GENER,R2       ;TABLE OF GENERIC CODES
003326  020022                          3$:     CMP     R0,(R2)+
003330  001067                                  BNE     1$
003332  011337  002002                          MOV     (R3),$MUT
003336  152777  000002  176462                  BISB    #CBIT,@CSR      ;SET C BIT
003344  005037  001766                          CLR     YLOOP           ;WAIT
003350  005237  001766                  64$:    INC     YLOOP
003354  023727  001766  000007                  CMP     YLOOP,#7
003362  001372                                  BNE     64$
003364  152777  000002  176434                  BISB    #CBIT,@CSR      ;DO IT AGAIN
003372  005037  001766                          CLR     YLOOP           ;WAIT
003376  005237  001766                  65$:    INC     YLOOP
003402  023727  001766  000007                  CMP     YLOOP,#7
003410  001372                                  BNE     65$
003412  011137  002016                          MOV     (R1),DMUT
003416  005037  002056                  9$:     CLR     PASS
003422  004777  176370                  4$:     JSR     PC,@DMUT        ;TEST MODULE
003426  004737  012446                          JSR     PC,CNTRC        ;CONTROL C?
003432  005237  002056                  2$:     INC     PASS
003436  023737  002060  002056                  CMP     PASCNT,PASS
003444  001366                                  BNE     4$
003446  032737  020000  000176                  BIT     #BIT13,SWREG    ;INHIBIT PRINT ?
003454  001005                                  BNE     7$
003456  013746  002060                          MOV     PASCNT,-(SP)    ;;SAVE PASCNT FOR TYPEOUT
003462  104402                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
003464  104401  017515                          TYPE    ,M11            ;# OF  PASSES
003470  032737  040000  000176          7$:     BIT     #BIT14,SWREG    ;LOOP ?
003476  001347                                  BNE     9$
003500  104401  021221                          TYPE    ,MASS15         ;END OF MODULE TEST
003504  000137  003002                          JMP     MONIT
003510  062701  000002                  1$:     ADD     #2,R1
003514  062703  000002                          ADD     #2,R3
003520  005712                                  TST     (R2)
003522  001301                                  BNE     3$
003524  104401  021060                          TYPE    ,MASS11         ;UNKNOWN GENERIC CODE
003530  010046                                  MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
003532  104402                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
003534  104401  020417                          TYPE    ,MASS0          ;CR,LF
```

```
   874  003540  012637  000004                MOV     (SP)+,ERRVEC    ;RESTORE ERRVEC
   875  003544  000137  003002                JMP     MONIT
   876  003550  022626              5$:       CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
   877  003552  104024                        ERROR 24                ;MODULE NOT RESPONDING
   878  003554  012637  000004                MOV     (SP)+,ERRVEC
   879  003560  000137  003002                JMP     MONIT
```

```
                                        .SBTTL  SET SOFTWARE SWITCH REGISTER
                                
003564  104401  020041          SWREGS: TYPE    ,M22            ;SWICHES OPTIONS
003570  104401  020216                  TYPE    ,M23            ;SW REG =
003574  104410                          RDOCT
003576  012637  000176                  MOV     (SP)+,SWREG     ;;POP STACK INTO SWREG
003602  104401  020417                  TYPE    ,MASSO          ;CR,LF
003606  000137  003002                  JMP     MONIT
```

```
                        .SBTTL  AUTO TEST MONITOR

                        ;;**********************************************************************
                        ;THIS PART OF A PROGRAM IS A MONITOR FOR AUTOMATIC TEST.
                        ;FIRST IT TEST 10CM.  THEN IT GENERATES A TABLE AND B TABLE.
                        ;THEN IT TESTS EACH INDIVIDUAL I/O MODULE CONNECTED TO DBUS.
                        ;REGISTERS R0 & R1 SERVE AS THE POINTERS FOR MUT

                        ;;**********************************************************************
003612  005037  002056    AUTO:   CLR     PASS
003616  012777  012550  176264        MOV     #KBINT,@KBVEC
003624  152777  000100  175312        BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
003632  004737  005414        JSR     PC,T10CM
003636  105777  176164        TSTB    @CSR            ;MAKE SURE 10CM IS OK
003642  001402        BEQ     7$
003644  104010        ERROR 10                        ;TEST ABORTED
003646  000207        RTS     PC
003650  013746  000004    7$:     MOV     @#ERRVEC,-(SP)  ;SAVE ERROR VECTOR
003654  012737  003766  000004        MOV     #1$,ERRVEC      ;SET ERROR VECTOR
003662  013700  002042        MOV     BASE,R0         ;R0=171000
003666  112777  000004  176132        MOVB    #GBIT,@CSR      ;SET GENERIC CODE ENABLE
003674  012702  001200        MOV     #ATABL,R2
003700  012705  001440        MOV     #BTABL,R5
003704  111001            5$:     MOVB    (R0),R1         ;R1=GENERIC CODE,READ GENERIC CODE
003706  042701  177400        BIC     #177400,R1
003712  004737  012200        JSR     PC,TABLE        ;FIND WHAT I/O IT IS
003716  000164  003722        JMP     3$(R4)
003722  000410            3$:     BR      10$
003724  000407        BR      10$
003726  000410        BR      12$
003730  000410        BR      13$
003732  000404        BR      10$
003734  000405        BR      12$
003736  000405        BR      13$
003740  000404        BR      13$
003742  000403        BR      13$
003744  005200            10$:    INC     R0
003746  005200            11$:    INC     R0
003750  005200            12$:    INC     R0
003752  005200            13$:    INC     R0
003754  020037  002030        CMP     R0,IAR
003760  001351        BNE     5$
003762  005012        CLR     (R2)
003764  000406        BR      6$
003766  022626            1$:     CMP     (SP)+,(SP)+
003770  005200        INC     R0
003772  020037  002030        CMP     R0,IAR
003776  001342        BNE     5$
004000  005015        CLR     (R5)
004002  012637  000004    6$:     MOV     (SP)+,@#ERRVEC  ;START TESTING INDIVIDUAL I/O
004006  142777  000004  176012        BICB    #GBIT,@CSR      ;CLEAR G BIT
004014  004737  005414            25$:    JSR     PC,T10CM
004020  005000        CLR     R0
004022  005001            22$:    CLR     R1
004024  026061  001440  001700    23$:    CMP     BTABL(R0),GENER(R1)
```

```
   944  004032  001416                          BEQ     21$
   945  004034  062701  000002                  ADD     #2,R1
   946  004040  005761  001700                  TST     GENER(R1)
   947  004044  001367                          BNE     23$
   948  004046  104401  021060                  TYPE    ,MASS11          ;UNKNOWN GENERIC CODE
   949  004052  016046  001440                  MOV     BTABL(R0),-(SP) ;;SAVE BTABL(R0) FOR TYPEOUT
   950  004056  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
   951  004060  104401  020417                  TYPE    ,MASS0
   952  004064  000137  003002                  JMP     MONIT
   953  004070  016137  001742  002016  21$:    MOV     MODUL(R1),DMUT
   954  004076  016037  001200  002006          MOV     ATABL(R0),TADDR
   955  004104  016137  001722  002002          MOV     MUT(R1),$MUT
   956  004112  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
   957  004114  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
   958  004116  004777  175674                  JSR     PC,@DMUT        ;TEST THIS MODULE
   959  004122  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
   960  004124  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
   961  004126  062700  000002                  ADD     #2,R0
   962  004132  005760  001440                  TST     BTABL(R0)
   963  004136  001331                          BNE     22$
   964  004140  005237  002056                  INC     PASS            ;INC PASS COUNT
   965  004144  004737  012446                  JSR     PC,CNTRC        ;CONTROL C ?
   966  004150  023737  002060  002056          CMP     PASCNT,PASS     ;IS IT 1000 PASSES
   967  004156  001316                          BNE     25$
   968  004160  005037  002056                  CLR     PASS
   969  004164  032737  020000  000176          BIT     #BIT13,SWREG    ;INHIBIT PRINT
   970  004172  001005                          BNE     20$
   971  004174  013746  002060                  MOV     PASCNT,-(SP)    ;;SAVE PASCNT FOR TYPEOUT
   972  004200  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
   973  004202  104401  017515                  TYPE    ,M11            ;# OF  PASSES COMPLITED
   974  004206  032737  040000  000176  20$:    BIT     #BIT14,SWREG    ;DO WE LOOP ON TESTS
   975  004214  001277                          BNE     25$
   976  004216  005777  175556          26$:    TST     @XXDP
   977  004222  001004                          BNE     27$
   978  004224  104401  021271          24$:    TYPE    ,MASS17         ;END OF SYSTEM TEST
   979  004230  000137  003002                  JMP     MONIT
   980  004234                          27$:
   981                                  .SBTTL  END OF PASS ROUTINE
   982
   983                                  ;;**********************************************************************
   984                                  ;*INCREMENT THE PASS NUMBER ($PASS)
   985                                  ;*IF THERES A MONITOR GO TO IT
   986                                  ;*IF THERE ISN'T JUMP TO 200
   987
   988  004234                          $EOP:
   989  004234  000004                          SCOPE
   990  004236  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
   991  004242  005237  001100                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
   992  004246  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG.  NUMBER
   993  004254  005327                          DEC     (PC)+           ;;LOOP?
   994  004256  000001                  $EOPCT: .WORD   1
   995  004260  003013                          BGT     $DOAGN          ;;YES
   996  004262  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
   997  004264  000001                  $ENDCT: .WORD   1
   998  004266  004256                          $EOPCT
   999  004270  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
```

```
. MAIN.  MACY11 30(1046)  25-OCT-77  10:10  PAGE 23
DVPCAA.P11    25-OCT-77 10:02          END OF PASS ROUTINE                                    SEQ 0025

  1000  004274  001405                        BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
  1001  004276  000005                        RESET                   ;;CLEAR THE WORLD
  1002  004300  004710                $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
  1003  004302  000240                        NOP                     ;;SAVE ROOM
  1004  004304  000240                        NOP                     ;;FOR
  1005  004306  000240                        NOP                     ;;ACT11
  1006  004310                        $DOAGN:
  1007  004310  000137  000200                JMP     @#200           ;;RETURN
```

```
1008                                        .SBTTL  LOOP TEST
1009
1010                                        ;THIS TEST ENABLE FIELD ENGINEER TO CONNECT OUTPUT MODULE TO INPUT MODULE
1011                                        ;OUTPUT TEST PATTERNS AND READ THEM BACK
1012                                        ;
1013                                        ;
1014  004314  104401  021515          LOOP: TYPE    ,MASS23           ;CONNECT OUTPUT TO INPUT MODULE
1015  004320  012777  012550  175562        MOV     #KBINT,@KBVEC
1016  004326  152777  000100  174610        BISB    #BIT6,@$TKS       ;ENABLE KB INTERRUPT
1017  004334  104401  021430                TYPE    ,MASS21           ;TYPE ADDRESS OF OUTPUT MUT
1018  004340  104410                        RDOCT
1019  004342  012637  002006                MOV     (SP)+,TADDR       ;;POP STACK INTO TADDR
1020  004346  104401  021463                TYPE    ,MASS22           ;TYPE ADDRESS OF INPUT MUT
1021  004352  104410                        RDOCT
1022  004354  012637  002050                MOV     (SP)+,TEMP1       ;;POP STACK INTO TEMP1
1023  004360  112777  000004  175440        MOVB    #GBIT,@CSR        ;SET GENERIC BIT
1024  004366  117700  175414                MOVB    @TADDR,R0         ;GENERIC CODE OF OUTPUT MUT
1025  004372  020027  000041                CMP     R0,#41            ;IS IT M6010?
1026  004376  001047                        BNE     1$
1027  004400  117701  175444                MOVB    @TEMP1,R1         ;GENERIC CODE OF INPUT MUT
1028  004404  020127  000141                CMP     R1,#141           ;IS IT M5010?
1029  004410  001403                        BEQ     2$
1030  004412  020127  000121                CMP     R1,#121           ;IS IT M5011?
1031  004416  001042                        BNE     6$
1032  004420  005037  002056          2$:   CLR     PASS
1033  004424  004737  013042          3$:   JSR     PC,LOPTST         ;TEST SUBROUTINE
1034  004430  004737  012360                JSR     PC,CLRINT         ;CLEAR ALL INTERRUPTS
1035  004434  004737  012446                JSR     PC,CNTRC          ;CONTROL-C?
1036  004440  005237  002056                INC     PASS
1037  004444  023737  002060  002056        CMP     PASCNT,PASS       ;
1038  004452  001364                        BNE     3$
1039  004454  032737  020000  000176        BIT     #BIT13,SWREG      ;INHIBIT PRINTOUT
1040  004462  001005                        BNE     5$
1041  004464  013746  002060                MOV     PASCNT,-(SP)      ;;SAVE PASCNT FOR TYPEOUT
1042  004470  104402                        TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1043  004472  104401  017515                TYPE    ,M11              ;PASSES COMPLETED
1044  004476  032737  040000  000176  5$:   BIT     #BIT14,SWREG      ;LOOP
1045  004504  001345                        BNE     2$
1046  004506  104401  021556                TYPE    ,MASS25           ;END OF LOOP TEST
1047  004512  000137  003002                JMP     MONIT
1048  004516  104401  021601          1$:   TYPE    ,MASS26           ;WRONG MODULE -OUTPUT MUST BE M6010
1049  004522  000402                        BR      4$
1050  004524  104401  021650          6$:   TYPE    ,MASS28           ;WRONG MODULE- INPUT MUST BE EITHER M5010 OR M5011
1051  004530                          4$:
1052  004530  152777  000002  175270        BISB    #CBIT,@CSR        ;SET C BIT
1053  004536  005037  001766                CLR     YLOOP             ;WAIT
1054  004542  005237  001766          64$:  INC     YLOOP
1055  004546  023727  001766  000007        CMP     YLOOP,#7
1056  004554  001372                        BNE     64$
1057  004556  152777  000002  175242        BISB    #CBIT,@CSR        ;DO IT AGAIN
1058  004564  005037  001766                CLR     YLOOP             ;WAIT
1059  004570  005237  001766          65$:  INC     YLOOP
1060  004574  023727  001766  000007        CMP     YLOOP,#7
1061  004602  001372                        BNE     65$
1062  004604  000137  003002                JMP     MONIT
```

```
1063                                    .SBTTL  MODULE TEST
1064
1065                                    ;THIS TEST ENABLE FIELD ENGINEER TO SELECT AND OUTPUT ANY DATA
1066                                    ;PATTERN TO OUTPUT MODULES,MONITOR AND PRINT DATA FROM INPUT MODULES
1067                                    ;
1068                                    ;
1069  004610  104401  021144      FIELD:  TYPE    ,MASS13         ;TYPE ADDRESS OF MUT
1070  004614  012777  012550  175266        MOV     #KBINT,@KBVEC
1071  004622  152777  000100  174314        BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
1072  004630  104410                        RDOCT
1073  004632  012637  002006              MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
1074  004636  013746  000004              MOV     ERRVEC, -(SP)   ;SAVE ERROR VECTOR
1075  004642  012737  005040  000004      MOV     #5$,ERRVEC      ;SET LOC 4
1076  004650  112777  000004  175150      MOVB    #GBIT,@CSR      ;SET GENERIC BIT
1077  004656  117700  175124              MOVB    @TADDR,R0       ;GENERIC CODE OF MUT
1078  004662  042700  177400              BIC     #177400,R0
1079  004666  012701  002066              MOV     #MOD,R1         ;ADDRESS OF TEST
1080  004672  012702  001700              MOV     #GENER,R2       ;TABLE OF GENERIC CODES
1081  004676  020022                3$:     CMP     R0,(R2)+
1082  004700  001037                        BNE     2$
1083  004702  152777  000002  175116      BISB    #CBIT,@CSR      ;SET C BIT
1084  004710  005037  001766              CLR     YLOOP           ;WAIT
1085  004714  005237  001766        64$:    INC     YLOOP
1086  004720  023727  001766  000007      CMP     YLOOP,#7
1087  004726  001372                        BNE     64$
1088  004730  152777  000002  175070      BISB    #CBIT,@CSR      ;DO IT AGAIN
1089  004736  005037  001766              CLR     YLOOP           ;WAIT
1090  004742  005237  001766        65$:    INC     YLOOP
1091  004746  023727  001766  000007      CMP     YLOOP,#7
1092  004754  001372                        BNE     65$
1093  004756  011137  002016              MOV     (R1),DMUT
1094  004762  004777  175030              JSR     PC,@DMUT        ;TEST MODULE
1095  004766  104401  021362              TYPE    ,MASS20         ;TYPE CONTROL-C TO RETURN TO MONITOR
1096  004772  004737  012446        1$:     JSR     PC,CNTRC        ;CONTROL-C ?
1097  004776  000775                        BR      1$
1098  005000  062701  000002        2$:     ADD     #2,R1
1099  005004  062703  000002              ADD     #2,R3
1100  005010  005712                        TST     (R2)
1101  005012  001331                        BNE     3$
1102  005014  104401  021060              TYPE    ,MASS11         ;UNKNOWN GENERIC CODE
1103  005020  010046                        MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
1104  005022  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1105  005024  104401  020417              TYPE    ,MASS0          ;CR,LF
1106  005030  012637  000004              MOV     (SP)+,ERRVEC    ;RESTORE ERRVEC
1107  005034  000137  003002              JMP     MONIT
1108  005040  022626                5$:     CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
1109  005042  104024                        ERROR 24                ;MODULE NOT RESPONDING
1110  005044  012637  000004              MOV     (SP)+,ERRVEC
1111  005050  000137  003002              JMP     MONIT
```

```
                        .SBTTL  MAP OF DBUS

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

                                ;THIS TEST WILL LIST ALL I/O INTERFACES CONNECTED TO IOCM
                                ;IT WILL ALSO LIST QBUS ADDRESSES

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


005054  013746  000004         MAPE:   MOV     @#ERRVEC,-(SP)  ;SAVE ERROR VECTOR
005060  012737  005260  000004          MOV     #1$,ERRVEC      ;SET ERROR VECTOR
005066  012777  012550  175014          MOV     #KBINT,@KBVEC
005074  152777  000100  174042          BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
005102  013700  002042                  MOV     BASE,R0         ;R0=171000
005106  112777  000004  174712          MOVB    #GBIT,@CSR      ;SET GENERIC CODE ENABLE
005114  111001                  5$:     MOVB    (R0),R1         ;R1=GENERIC CODE,READ GENERIC CODE
005116  042701  177400                  BIC     #177400,R1
005122  004737  012154                  JSR     PC,GENCOD       ;FIND WHAT I/O IT IS
005126  104401  020422                  TYPE    ,MASS2          ;ADDRESS
005132  010046                          MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
005134  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
005136  000164  005142                  JMP     3$(R4)
005142  104401  020434          3$:     TYPE    ,MASS3          ;M5010
005146  000434                          BR      10$
005150  104401  020503                  TYPE    ,MASS4          ;M5011
005154  000431                          BR      10$
005156  104401  020552                  TYPE    ,MASS5          ;M5012
005162  000430                          BR      12$
005164  104401  020616                  TYPE    ,MASS6          ;M5013
005170  000426                          BR      13$
005172  104401  020650                  TYPE    ,MASS7          ;M6010
005176  000420                          BR      10$
005200  104401  020720                  TYPE    ,MASS8          ;M6011
005204  000417                          BR      12$
005206  104401  020761                  TYPE    ,MASS9          ;M6012
005212  000415                          BR      13$
005214  104401  021025                  TYPE    ,MASS10         ;M6013
005220  000412                          BR      13$
005222  104401  021060                  TYPE    ,MASS11         ;UNKNOWN GENERIC CODE
005226  010146                          MOV     R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
005230  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
005232  104401  020417                  TYPE    ,MASS0          ;CR,LF
005236  000403                          BR      13$
005240  005200                  10$:    INC     R0              ;INC ADDRESS BY APPROPRIATE NUMBER OF BYTES
005242  005200                  11$:    INC     R0
005244  005200                  12$:    INC     R0
005246  005200                  13$:    INC     R0
005250  020037  002030                  CMP     R0,IAR          ;LAST ADDRESS ?
005254  001317                          BNE     5$              ;NO, DO IT AGAIN
005256  000405                          BR      6$
005260  022626                  1$:     CMP     (SP)+,(SP)+     ;HERE IF ADDRESS IS NOT RESPONDING
005262  005200                          INC     R0              ;INC ADDRESS
```

```
 1168  005264  020037  002030            CMP     R0,IAR            ;LAST ONE ?
 1169  005270  001311                    BNE     5$
 1170  005272  012637  000004      6$:   MOV     (SP)+,@#ERRVEC    ;RESTORE ERROR VECTOR
 1171  005276  000137  003002            JMP     MONIT
```

```
1172                                    .SBTTL  10CM TEST
1173
1174                                    ;;**********************************************************************
1175
1176                                            ;THIS PART OF DIAGNOSTIC TEST THE 10CM
1177                                            ;IT HAS 9 TESTS.  IT CHECKS IF ALL THE BITS OF THE 10CM
1178                                            ;CAN BE SET AND CLEAR , CHECKS MAINTENANCE
1179                                            ;INTERRUPT AND CHECKS ALL ADDRESSES IN MAINTENANCE MODE
1180
```

```
 1181                                   ;;**************************************************************************
 1182  005302  005037  002056           IOCM:   CLR     PASS
 1183  005306  013746  000004                   MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
 1184  005312  012777  012550  174570           MOV     #KBINT,@KBVEC
 1185  005320  152777  000100  173616           BISB    #BIT6,@$TKS     ;ENABLE KB INTERRUPT
 1186  005326  004737  005414           1$:     JSR     PC,TIOCM        ;TEST IOCM
 1187  005332  005237  002056                   INC     PASS
 1188  005336  023737  002060  002056           CMP     PASCNT,PASS     ;1000 PASSES DONE ?
 1189  005344  001370                           BNE     1$
 1190  005346  032737  020000  000176           BIT     #BIT13,SWREG    ;INHIBIT PRINT
 1191  005354  001005                           BNE     2$
 1192  005356  013746  002060                   MOV     PASCNT,-(SP)    ;;SAVE PASCNT FOR TYPEOUT
 1193  005362  104402                           TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1194  005364  104401  017515                   TYPE    ,M11            ;# OF  PASSES
 1195  005370  032737  040000  000176   2$:     BIT     #BIT14,SWREG    ;LOOP ON TEST?
 1196  005376  001353                           BNE     1$
 1197  005400  104401  021246                   TYPE    ,MASS16         ;IOCM TEST PASSED
 1198  005404  012637  000004                   MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
```

```
. MAIN.   MACY11 30(1046)  25-OCT-77  10:10  PAGE 30
DVPCAA.P11     25-OCT-77 10:02          10CM TEST                                SEQ 0032

  1199  005410  000137  003002                    JMP     MONIT
```

```
1200                                            ;THIS TEST CHECKS IF EACH BIT OF CSR
1201                                            ;IS CLEAR BY CBIT
1202  005414                                    TIOCM:
1203                                            ;;**********************************************************************
1204  005414  000004                           TST1:   SCOPE
1205  005416  112737  000001  002022                    MOVB    #1,TNUMB
1206  005424  012737  005540  000004                    MOV     #10$,ERRVEC
1207  005432  112777  000074  174366                    MOVB    #74,@CSR        ;SET ALL BITS
1208  005440  152777  000002  174360                    BISB    #CBIT,@CSR      ;SET C BIT
1209  005446  005037  001766                           CLR     YLOOP           ;WAIT
1210  005452  005237  001766                  64$:     INC     YLOOP
1211  005456  023727  001766  000007                    CMP     YLOOP,#7
1212  005464  001372                                    BNE     64$
1213  005466  152777  000002  174332                    BISB    #CBIT,@CSR      ;DO IT AGAIN
1214  005474  005037  001766                           CLR     YLOOP           ;WAIT
1215  005500  005237  001766                  65$:     INC     YLOOP
1216  005504  023727  001766  000007                    CMP     YLOOP,#7
1217  005512  001372                                    BNE     65$
1218  005514  117737  174306  001126                    MOVB    @CSR,$BDDAT     ;STORE CONTENTS OF CSR IN BDDAT
1219  005522  005037  001124                           CLR     $GDDAT
1220  005526  005737  001126                           TST     $BDDAT          ;IS CSR CLEAR
1221  005532  001405                                    BEQ     2$              ;YES
1222  005534  104001                                    ERROR 1                 ;CBIT NOT CLEARING IOCM
1223  005536  000207                                    RTS     PC              ;FATAL ERROR RETURN TO MONITOR
1224  005540  104003                  10$:     ERROR 3                 ;IOCM NOT RESPONDING
1225  005542  022626                                    CMP     (SP)+,(SP)+     ;RESTORE STACK
1226  005544  000207                                    RTS     PC
1227  005546                           2$:
```

```
                                                ;THIS TEST CHECKS E BIT
                                                ;;*****************************************************************
005546  000004                                TST2:   SCOPE
005550  112737  000002  002022                        MOVB    #2,TNUMB
005556  012737  000100  001124                        MOV     #EBIT,$GDDAT    ;SET TESTED BIT
005564  113737  017512  016326                        MOVB    M7,EM2X         ;SET EBIT IN ERROR MESSAGE
005572  004737  012230                              JSR     PC,BITSET
005576  104002                                      ERROR 2                 ;BIT IS NOT SETTING
005600  005037  001124                              CLR     $GDDAT
005604  004737  012230                              JSR     PC,BITSET
005610  104002                                      ERROR 2

                                                ;THIS TEST CHECKS MBIT
                                                ;;*****************************************************************
005612  000004                                TST3:   SCOPE
005614  112737  000003  002022                        MOVB    #3,TNUMB
005622  012737  000240  001124                        MOV     #MBIT!#FBIT,$GDDAT;SET MAITENANCE BIT
005630  113737  017507  016326                        MOVB    M6,EM2X         ;SET ERROR MESSAGE
005636  004737  012230                              JSR     PC,BITSET
005642  104002                                      ERROR 2                 ;BIT NOT SETTING
005644  005037  001124                              CLR     $GDDAT
005650  004737  012230                              JSR     PC,BITSET,
005654  104002                                      ERROR 2                 ;BIT NOT CLEARING


                                                ;THIS TEST CHECKS DBIT
                                                ;;*****************************************************************
005656  000004                                TST4:   SCOPE
005660  112737  000004  002022                        MOVB    #4,TNUMB
005666  012737  000020  001124                        MOV     #DBIT,$GDDAT
005674  113737  017504  016326                        MOVB    M5,EM2X         ;FIX ERROR MESSAGE
005702  004737  012230                              JSR     PC,BITSET       ;SET D BIT
005706  104002                                      ERROR 2                 ;BIT NOT SETTING
005710  005037  001124                              CLR     $GDDAT
005714  004737  012230                              JSR     PC,BITSET
005720  104002                                      ERROR 2                 ;BIT NOT CLEARING
005722  112777  000074  174076                        MOVB    #74,@CSR        ;SET FEW BITS AT CSR
005730  142777  000020  174070                        BICB    #DBIT,@CSR      ;CLEAR DBIT
005736  005037  001766                              CLR     YLOOP           ;WAIT
005742  005237  001766                64$:    INC     YLOOP
005746  023727  001766  000007                      CMP     YLOOP,#7
005754  001372                                      BNE     64$
005756  117737  174044  001124                      MOVB    @CSR,$GDDAT
005764  132737  000177  001124                      BITB    #177,$GDDAT
005772  001401                                      BEQ     1$              ;CHECK IF DBIT CLEARING CLEARS CSR
005774  104016                                      ERROR 16                ;CLEARING DBIT DOES NOT CLEAR CSR
005776                                          1$:
005776  152777  000002  174022                      BISB    #CBIT,@CSR      ;SET C BIT
006004  005037  001766                              CLR     YLOOP           ;WAIT
006010  005237  001766                65$:    INC     YLOOP
006014  023727  001766  000007                      CMP     YLOOP,#7
006022  001372                                      BNE     65$
006024  152777  000002  173774                      BISB    #CBIT,@CSR      ;DO IT AGAIN
006032  005037  001766                              CLR     YLOOP           ;WAIT
006036  005237  001766                66$:    INC     YLOOP
006042  023727  001766  000007                      CMP     YLOOP,#7
```

```
1284  006050  001372                          BNE     66$
1285
1286
1287                                          ;THIS TEST CHECKS TBIT
1288                                        ;;**************************************************************
1289  006052  000004                        TST5:   SCOPE
1290  006054  112737  000005  002022              MOVB    #5,TNUMB
1291  006062  012737  000010  001124              MOV     #TBIT,$GDDAT
1292  006070  113737  017501  016326              MOVB    M4,EM2X         ;SET ERROR MESSAGE
1293  006076  004737  012230                      JSR     PC,BITSET       ;SET AND CHECK T BIT
1294  006102  104002                              ERROR 2                 ;BIT IS NOT SETTING
1295  006104  005037  001124                      CLR     $GDDAT
1296  006110  004737  012230                      JSR     PC,BITSET       ;CLEAR T BIT
1297  006114  104002                              ERROR 2                 ;BIT NOT CLEAR
1298  006116  004737  012360                      JSR     PC,CLRINT       ;CLEAR ALL INTERRUPT CAUSED BY T BIT
1299
1300                                          ;THIS TEST CHECKS GBIT
1301                                        ;;**************************************************************
1302  006122  000004                        TST6:   SCOPE
1303  006124  112737  000006  002022              MOVB    #6,TNUMB
1304  006132  012737  000004  001124              MOV     #GBIT,$GDDAT
1305  006140  113737  017476  016326              MOVB    M3,EM2X         ;FIX ERROR MESSAGE
1306  006146  004737  012230                      JSR     PC,BITSET       ;SET AND CHECK G BIT
1307  006152  104002                              ERROR 2                 ;BIT NOT SETTING
1308  006154  005037  001124                      CLR     $GDDAT
1309  006160  004737  012230                      JSR     PC,BITSET       ;CLEAR G BIT
1310  006164  104002                              ERROR 2                 ;BIT'S NOT CLEARING
1311
1312
1313                                          ;THIS TEST CHECKS RBIT
1314                                        ;;**************************************************************
1315  006166  000004                        TST7:   SCOPE
1316  006170  112737  000007  002022              MOVB    #7,TNUMB
1317  006176  012737  000001  001124              MOV     #RBIT,$GDDAT
1318  006204  113737  017474  016326              MOVB    M2,EM2X         ;SET ERROR MESSAGE
1319  006212  004737  012230                      JSR     PC,BITSET       ;SET BIT
1320  006216  104002                              ERROR 2                 ;BIT IS NOT SETTING
1321  006220  005037  001124                      CLR     $GDDAT
1322  006224  004737  012230                      JSR     PC,BITSET       ;CLEAR BIT
1323  006230  104002                              ERROR 2                 ;BIT IS NOT CLEAR
```

```
1324
1325                                    ;;**********************************************************************
1326
1327                                            ;THIS TEST CHECKS ALL BITS OF DBUS IN A MAINTENANCE MODE.
1328                                            ;IF MBIT IS SET AND CPU ADDRESSES ANY LOCATION BETWEEN
1329                                            ;171000 AND 171375 IT SHOULD READ BACK A LOWER
1330                                            ;BYTE OF AN ADDRESS.
1331
1332                                    ;;**********************************************************************
1333
1334                                    ;;**********************************************************************
1335  006232  000004                    TST10:  SCOPE
1336  006234  112737  000010  002022            MOVB    #10,TNUMB
1337  006242  112777  000040  173556            MOVB    #MBIT,@CSR      ;SET MAINTENANCE MODE
1338  006250  013700  002042                    MOV     BASE,R0
1339  006254  005001                            CLR     R1
1340  006256  005037  001124                    CLR     $GDDAT
1341  006262  005037  001126                    CLR     $BDDAT
1342  006266  111001                    1$:     MOVB    (R0),R1         ;READ FIRST ADDRESS
1343  006270  042701  177400                    BIC     #177400,R1
1344  006274  120001                            CMPB    R0,R1           ;CHECK IF DBUS=ADDRESS
1345  006276  001405                            BEQ     2$
1346  006300  110037  001124                    MOVB    R0,$GDDAT
1347  006304  110137  001126                    MOVB    R1,$BDDAT
1348  006310  104004                            ERROR 4                 ;DBUS BIT STACK
1349  006312  005200                    2$:     INC     R0
1350  006314  122700  000376                    CMPB    #376,R0         ;IS IT THE LAST ADDRESS
1351  006320  001362                            BNE     1$
1352  006322  152777  000002  173476            BISB    #CBIT,@CSR      ;SET C BIT
1353  006330  005037  001766                    CLR     YLOOP           ;WAIT
1354  006334  005237  001766            64$:    INC     YLOOP
1355  006340  023727  001766  000007            CMP     YLOOP,#7
1356  006346  001372                            BNE     64$
1357  006350  152777  000002  173450            BISB    #CBIT,@CSR      ;DO IT AGAIN
1358  006356  005037  001766                    CLR     YLOOP           ;WAIT
1359  006362  005237  001766            65$:    INC     YLOOP
1360  006366  023727  001766  000007            CMP     YLOOP,#7
1361  006374  001372                            BNE     65$
```

```
 1362
 1363                                   ;;**********************************************************************
 1364
 1365                                           ;THIS TEST WILL CHECK MAINTENANCE INTERRUPT
 1366                                           ;IF MBIT & EBIT ARE SET IOCM WILL GENERATE AN INTERRUPT
 1367                                           ;AT LOCATION 234 AND IAR WILL HAVE LOWER BYTE
 1368                                           ;OF CSR ADDRESS
 1369
 1370                                   ;;**********************************************************************
 1371
 1372                                   ;;**********************************************************************
 1373  006376  000004                  TST11:  SCOPE
 1374  006400  112737  000011  002022          MOVB    #11,TNUMB
 1375  006406  012777  006502  173416          MOV     #5$,@VECTO      ;SET VECTOR ADDRESS
 1376  006414  012777  000340  173412          MOV     #PR7,@VECTOA    ;SET VECTOR+2 ADDRESS
 1377  006422  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
 1378  006426  012737  006664  000004          MOV     #8$,ERRVEC
 1379  006434  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
 1380  006440  012746  006446                  MOV     #64$,-(SP)
 1381  006444  000002                          RTI
 1382  006446  000240                  64$:    NOP
 1383  006450  112777  000140  173350  1$:     MOVB    #EBIT!#MBIT,@CSR;START INTERRUPT
 1384  006456  005037  001766                  CLR     YLOOP           ;WAIT
 1385  006462  005237  001766          65$:    INC     YLOOP
 1386  006466  023727  001766  177777          CMP     YLOOP,#-1
 1387  006474  001372                          BNE     65$
 1388  006476  104014                          ERROR 14                ;NO INTERRUPT
 1389  006500  000431                          BR      6$
 1390  006502  022626                  5$:     CMP     (SP)+,(SP)+     ;INTERRUPT WORKED
 1391  006504  117737  173320  002046          MOVB    @IAR,TEMP       ;READ ADDRESS OF INTERRUPTING MODULE
 1392  006512  122737  000377  002046          CMPB    #377,TEMP       ;IAR SHOULD HAVE ALL ONES
 1393  006520  001407                          BEQ     7$
 1394  006522  012737  000377  001124          MOV     #377,$GDDAT
 1395  006530  013737  002046  001126          MOV     TEMP,$BDDAT
 1396  006536  104006                          ERROR 6                 ;WRONG PATTERN IN IAR
 1397  006540  152777  000001  173260  7$:     BISB    #RBIT,@CSR
 1398  006546  105777  173254                  TSTB    @CSR            ;CLEAR INTERRUPT
 1399  006552  132777  000200  173246          BITB    #FBIT,@CSR      ;CHECK IF INTERRUPT CLEAR
 1400  006560  001401                          BEQ     6$
 1401  006562  104015                          ERROR 15                ;RIF BIT NOT CLEARING INTERRUPT
 1402  006564                          6$:
 1403  006564  152777  000002  173234          BISB    #CBIT,@CSR      ;SET C BIT
 1404  006572  005037  001766                  CLR     YLOOP           ;WAIT
 1405  006576  005237  001766          66$:    INC     YLOOP
 1406  006602  023727  001766  000007          CMP     YLOOP,#7
 1407  006610  001372                          BNE     66$
 1408  006612  152777  000002  173206          BISB    #CBIT,@CSR      ;DO IT AGAIN
 1409  006620  005037  001766                  CLR     YLOOP           ;WAIT
 1410  006624  005237  001766          67$:    INC     YLOOP
 1411  006630  023727  001766  000007          CMP     YLOOP,#7
 1412  006636  001372                          BNE     67$
 1413  006640  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
 1414  006644  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
 1415  006650  012746  006656                  MOV     #68$,-(SP)
 1416  006654  000002                          RTI
 1417  006656  000240                  68$:    NOP
```

```
 1418  006660  000004                          SCOPE
 1419  006662  000207                          RTS     PC
 1420
 1421  006664  022626                  8$:     CMP     (SP)+,(SP)+
 1422  006666  104022                          ERROR 22                ;ADDRESS TEST WITH MBIT SET NOT WORKING
 1423  006670  000735                          BR      6$
 1424
```

```
1425
1426                                    .SBTTL  DIGITAL MODUL TEST ,M5010
1427                                    ;;**********************************************************************
1428
1429                                            ;THIS TEST WILL CHECK M5010 MODULE
1430                                            ;32 BIT NONISOLATED DC SENSE
1431
1432                                    ;;**********************************************************************
1433
1434
1435  006672                            M5010:
1436                                    ;;**********************************************************************
1437  006672  000004                    TST12:  SCOPE
1438  006674  112737  000012  002022            MOVB    #12,TNUMB
1439  006702  152777  000020  173116            BISB    #DBIT,@CSR      ;SET DISABLE BIT
1440  006710  112737  000000  001124            MOVB    #0,$GDDAT       ;SET WHAT DATA SHOULD BE
1441  006716  012737  000004  002004            MOV     #4,BYTNUM       ;SET NUMBER OF BYTES
1442  006724  004737  011746                    JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
1443
1444  006730  152777  000010  173070            BISB    #TBIT,@CSR
1445  006736  112737  000377  001124            MOVB    #377,$GDDAT     ;SET WHAT DATA SHOULD BE
1446  006744  012737  000004  002004            MOV     #4,BYTNUM       ;SET NUMBER OF BYTES
1447  006752  004737  011746                    JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
1448  006756  142777  000010  173042            BICB    #TBIT,@CSR
1449  006764  004737  012360                    JSR     PC,CLRINT
1450  006770  000004                            SCOPE
1451  006772  000207                            RTS     PC
```

```
1452
1453                                    .SBTTL  DIGITAL MODULE TEST ,M5013
1454
1455                                    ;;**********************************************************************
1456
1457                                            ;THIS TEST CHECKS M5013 MODULE
1458                                            ;8 BIT AC SENSE
1459
1460                                    ;;**********************************************************************
1461
1462
1463  006774                            M5013:
1464                                    ;;**********************************************************************
1465  006774  000004                    TST13:  SCOPE
1466  006776  112737  000013  002022            MOVB    #13,TNUMB
1467  007004  152777  000020  173014            BISB    #DBIT,@CSR      ;SET DISABLE BIT
1468  007012  112737  000000  001124            MOVB    #0,$GDDAT       ;SET WHAT DATA SHOULD BE
1469  007020  012737  000001  002004            MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
1470  007026  004737  011746                    JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
1471
1472  007032  152777  000010  172766            BISB    #TBIT,@CSR      ;SET COMPLIM BIT
1473  007040  004737  012360                    JSR     PC,CLRINT
1474  007044  112737  000377  001124            MOVB    #377,$GDDAT     ;SET WHAT DATA SHOULD BE
1475  007052  012737  000001  002004            MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
1476  007060  004737  011746                    JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
1477  007064  142777  000010  172734            BICB    #TBIT,@CSR
1478  007072  004737  012360                    JSR     PC,CLRINT
1479
1480
1481  007076  012746  000000                    MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
1482  007102  012746  007110                    MOV     #64$,-(SP)
1483  007106  000002                            RTI
1484  007110  000240                    64$:    NOP
1485  007112  012777  007210  172712            MOV     #5$,@VECTO      ;SET INTERRUPTVECTOR ADDRESS
1486  007120  012777  000340  172706            MOV     #PR7,@VECTOA
1487  007126  152777  000010  172672            BISB    #TBIT,@CSR      ;START INTERRUPT
1488  007134  152777  000100  172664            BISB    #EBIT,@CSR      ;ENABLE INTERRUPT
1489  007142  005037  001766                    CLR     YLOOP           ;WAIT
1490  007146  005237  001766            65$:    INC     YLOOP
1491  007152  023727  001766  177777            CMP     YLOOP,#-1
1492  007160  001372                            BNE     65$
1493  007162  104005                            ERROR 5                 ;NO INTERRUPT
1494  007164  142777  000100  172634    7$:     BICB    #EBIT,@CSR      ;CLEAR CSR
1495  007172  142777  000010  172626            BICB    #TBIT,@CSR
1496  007200  004737  012360                    JSR     PC,CLRINT       ;CLEAR ALL INTERRUPT
1497  007204  000004                            SCOPE
1498  007206  000207                            RTS     PC              ;RETURN TO MONITOR
1499
1500  007210  022626                    5$:     CMP     (SP)+,(SP)+     ;RESET STACK
1501  007212  005037  002050                    CLR     TEMP1
1502  007216  013704  000004                    MOV     ERRVEC,R4
1503  007222  012737  007430  000004            MOV     #4$,ERRVEC
1504  007230  117737  172574  002046    6$:     MOVB    @IAR,TEMP       ;CHECK WHICH I/O INTERRUPTED
1505  007236  152777  000001  172562            BISB    #RBIT,@CSR      ;HERE IF INTERRUPT
1506  007244  123737  002046  002006            CMPB    TEMP,TADDR      ;IS IT MUT
1507  007252  001414                            BEQ     2$
```

```
1508  007254  053737  002042  002046          BIS     BASE,TEMP       ;ASSEMBLE ADDRESS  OF INTERRUPTING MODULE
1509  007262  105777  172560                  TSTB    @TEMP
1510  007266  005237  002050                  INC     TEMP1           ;LOOP NO MORE THEN 400 TIMES
1511  007272  022737  000400  002050          CMP     #400,TEMP1
1512  007300  001353                          BNE     6$
1513  007302  104005                          ERROR 5                 ;NO INTERRUPT
1514  007304  105777  172476          2$:     TSTB    @TADDR          ;CLEAR INTERRUPT
1515  007310  132777  000200  172510          BITB    #FBIT,@CSR      ;CHECK IF INTERRUPT CLEAR
1516  007316  001413                          BEQ     8$
1517  007320  117737  172504  002046          MOVB    @IAR,TEMP
1518  007326  023737  002046  002006          CMP     TEMP,TADDR
1519  007334  001002                          BNE     9$
1520  007336  104015                          ERROR 15                ;RIF BIT NOT CLEARING INTERRUPT
1521  007340  000430                          BR      3$
1522  007342  004737  012360          9$:     JSR     PC,CLRINT
1523  007346  132777  000010  172452  8$:     BITB    #TBIT,@CSR
1524  007354  001422                          BEQ     3$
1525  007356  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
1526  007362  012746  007370                  MOV     #66$,-(SP)
1527  007366  000002                          RTI
1528  007370  000240          66$:    NOP
1529  007372  142777  000010  172426          BICB    #TBIT,@CSR      ;CHECK IF CLEARING TBIT CAUSE INTERRUPT
1530  007400  005037  001766                  CLR     YLOOP           ;WAIT
1531  007404  005237  001766          67$:    INC     YLOOP
1532  007410  023727  001766  177777          CMP     YLOOP,#-1
1533  007416  001372                          BNE     67$
1534  007420  104005                          ERROR 5                 ;NO INTERRUPT
1535  007422  010437  000004          3$:     MOV     R4,ERRVEC
1536  007426  000656                          BR      7$
1537  007430  022626                  4$:     CMP     (SP)+,(SP)+
1538  007432  104005                          ERROR 5                 ;NO INTERRUPT
1539  007434  000137  007304                  JMP     2$
1540
```

DIGITAL MODULE TEST ,M6010 SEQ 0042

```
                                        .SBTTL  DIGITAL MODULE TEST ,M6010

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

                                                ;THIS TEST CHECKS M6010 MODULE
                                                ;32 BIT NONISOLATED DC OUT

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

007440                                  M6010:
                                        ;;**************************************************************
007440  000004                  TST14:  SCOPE
007442  112737  000014  002022          MOVB    #14,TNUMB
007450  152777  000020  172350          BISB    #DBIT,@CSR      ;SET D BIT
007456  013700  002006                  MOV     TADDR,R0
007462  105020                          CLRB    (R0)+           ;CLEAR ALL FOUR BYTES OF I/O REGISTER
007464  105020                          CLRB    (R0)+
007466  105020                          CLRB    (R0)+
007470  105020                          CLRB    (R0)+
007472  005000                          CLR     R0
007474  005001                  4$:     CLR     R1
007476  005037  001770                  CLR     DBUFF           ;CLEAR IMIGE OF I/O REGISRERS
007502  005037  001772                  CLR     DBUFF+2
007506  116160  001774  001770  3$:     MOVB    PATT(R1),DBUFF(R0);SET DATA PATTERN IN IMIGE
007514  013737  002006  002046          MOV     TADDR,TEMP
007522  060037  002046                  ADD     R0,TEMP
007526  116177  001774  172312          MOVB    PATT(R1),@TEMP  ;SET DATA IN I/O REG
007534  005002                          CLR     R2
007536  013737  002006  002050  2$:     MOV     TADDR,TEMP1
007544  060237  002050                  ADD     R2,TEMP1
007550  117737  172274  001126          MOVB    @TEMP1,$BDDAT   ;READ I/O REGISTER
007556  123762  001126  001770          CMPB    $BDDAT,DBUFF(R2);IS DATA OK?
007564  001404                          BEQ     1$
007566  116237  001770  001124          MOVB    DBUFF(R2),$GDDAT
007574  104017                          ERROR 17                ;DATA ERROR
007576  005202                  1$:     INC     R2
007600  022702  000004                  CMP     #4,R2
007604  001354                          BNE     2$              ;TEST IF OTHER BYTES ARE OK
007606  005201                          INC     R1
007610  022701  000004                  CMP     #4,R1           ;LAST PATTERN?
007614  001334                          BNE     3$              ;TEST NEXT DATA PATTERN
007616  005200                          INC     R0
007620  022700  000004                  CMP     #4,R0           ;LAST BYTE?
007624  001323                          BNE     4$              ;NO. TEST NEXT BYTE
007626  000004                          SCOPE
007630  000207                          RTS     PC
```

```
                                        .SBTTL  DIGITAL MODULE TEST , M6012,M6013

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

                                                ;THIS TEST CHECKS MODULES M6012 AND M6013
                                                ;8 BIT AC AND DC OUT

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

007632                                  M6012:
007632                                  M6013:
                                        ;;**********************************************************************
007632  000004                          TST15:  SCOPE
007634  112737  000015  002022                MOVB    #15,TNUMB
007642  152777  000020  172156                BISB    #DBIT,@CSR
007650  012737  000125  001124                MOV     #125,$GDDAT     ;SET DATA PATTERN
007656  012737  000001  002004                MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
007664  004737  012112                        JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
007670  012737  000001  002004                MOV     #1,BYTNUM
007676  004737  011746                        JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY

007702  012737  000252  001124                MOV     #252,$GDDAT     ;SET DATA PATTERN
007710  012737  000001  002004                MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
007716  004737  012112                        JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
007722  012737  000001  002004                MOV     #1,BYTNUM
007730  004737  011746                        JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY

007734  012737  000377  001124        MOV     #377,$GDDAT     ;SET DATA PATTERN
007742  012737  000001  002004        MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
007750  004737  012112                JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
007754  012737  000001  002004        MOV     #1,BYTNUM
007762  004737  011746                JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY

007766  012737  000000  001124        MOV     #0,$GDDAT       ;SET DATA PATTERN
007774  012737  000001  002004        MOV     #1,BYTNUM       ;SET NUMBER OF BYTES
010002  004737  012112                JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
010006  012737  000001  002004        MOV     #1,BYTNUM
010014  004737  011746                JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY
010020  000004                        SCOPE
010022  000207                        RTS     PC
```

```
                                        .SBTTL  DIGITAL MODULE TEST ,M6011
                                        ;;*****************************************************************
                                        ;THIS TEST CHECKS MODULE M6011
                                        ;ONE SHOT DC OUT
                                        ;;*****************************************************************

010024                                  M6011:
                                        ;;*****************************************************************
010024  000004                          TST16:  SCOPE
010026  112737  000016  002022                  MOVB    #16,TNUMB
010034  013737  002006  002010                  MOV     TADDR,TADDR1
010042  005237  002010                          INC     TADDR1
010046  152777  000020  171752                  BISB    #DBIT,@CSR
010054  005037  002044                          CLR     CLK             ;CLEAR CLOCK COUNTER
010060  012777  012352  171774                  MOV     #COUNT,@CLKVC   ;SET LOC 100- CLOCK VECTOR
010066  012777  000340  171770                  MOV     #PR7,@CLKVCA    ;SET LOC 102
010074  012746  000000                          MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
010100  012746  010106                          MOV     #64$,-(SP)
010104  000002                                  RTI
010106  000240                          64$:    NOP
010110  012737  000252  001124                  MOV     #252,$GDDAT     ;SET DATA PATTERN
010116  012737  000002  002004                  MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
010124  004737  012112                          JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
010130  012737  000002  002004                  MOV     #2,BYTNUM
010136  004737  011746                          JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY
010142  005001                                  CLR     R1
010144  005201                          11$:    INC     R1              ;CHECK IF LINE CLOCK IS INTERRUPTING
010146  001376                                  BNE     11$
010150  005737  002044                          TST     CLK
010154  001010                                  BNE     1$
010156  104025                                  ERROR 25                ;LINE CLOCK ISN'T INTERRUPTING
010160  000453                                  BR      M6011B
010162  105777  171620                  2$:     TSTB    @TADDR
010166  001003                                  BNE     1$
010170  105777  171614                          TSTB    @TADDR1
010174  001410                                  BEQ     M6011A
010176  022737  001131  002044          1$:     CMP     #601.,CLK               ;WAIT 10SEC
010204  001366                                  BNE     2$
010206  005037  001124                          CLR     $GDDAT
010212  004737  012030                          JSR     PC,TSTONE       ;PATTERN SHOULD BE CLEAR
010216                                  M6011A:
010216  005037  002044                          CLR     CLK             ;CLEAR CLOCK COUNTER
010222  012737  000125  001124                  MOV     #125,$GDDAT     ;SET DATA PATTERN
010230  012737  000002  002004                  MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
010236  004737  012112                          JSR     PC,SETBYT       ;SET DATA IN I/O MODULE
010242  012737  000002  002004                  MOV     #2,BYTNUM
010250  004737  011746                          JSR     PC,TSTBYT       ;TEST IF DATA IS SET CORRECTLY
010254  105777  171526                  2$:     TSTB    @TADDR
010260  001003                                  BNE     1$
010262  105777  171522                          TSTB    @TADDR1
010266  001410                                  BEQ     M6011B
010270  022737  001131  002044          1$:     CMP     #601.,CLK       ;WAIT 10 SEC
010276  001366                                  BNE     2$
010300  005037  001124                          CLR     $GDDAT
```

```
 1688  010304  004737  012030                  JSR     PC,TSTONE       ;PATTERN SHOULD BE CLEAR
 1689  010310  000004            M6011B:       SCOPE
 1690  010312  012777  012350  171542          MOV     #NOCLK,@CLKVC   ;RESTORE CLOCK VECTOR
 1691  010320  000207                          RTS     PC
```

```
                                        .SBTTL  DIGITAL MODULE TEST ,M5011

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

                                                ;THIS TEST CHECKS M5011 MODULE
                                                ;16 BIT CHANGE OF STATE SENSE

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


010322                                  M5011:
                                        ;;**********************************************************************
010322  000004                          TST17:  SCOPE
010324  112737  000017  002022                  MOVB    #17,TNUMB
010332  013737  002006  002014                  MOV     TADDR,COSADR
010340  062737  000002  002014                  ADD     #2,COSADR
010346  152777  000030  171452                  BISB    #DBIT!#TBIT,@CSR;SET DISABLE BIT AND TBIT
010354  112737  000377  001124                  MOVB    #377,$GDDAT     ;SET WHAT DATA SHOULD BE
010362  012737  000002  002004                  MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
010370  004737  011746                          JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT

010374  142777  000010  171424                  BICB    #TBIT,@CSR      ;CLEAR COMPLIM BIT
010402  112737  000000  001124                  MOVB    #0,$GDDAT       ;SET WHAT DATA SHOULD BE
010410  012737  000002  002004                  MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
010416  004737  011746                          JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
010422  004737  012360                          JSR     PC,CLRINT
010426  012746  000000                          MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
010432  012746  010440                          MOV     #64$,-(SP)
010436  000002                                  RTI
010440  000240                          64$:    NOP
010442  012777  010540  171362                  MOV     #1$,@VECTO      ;SET INTERRUPT VECTOR
010450  012777  000340  171356                  MOV     #PR7,@VECTOA
010456  152777  000010  171342                  BISB    #TBIT,@CSR      ;START INTERRUPT
010464  152777  000100  171334                  BISB    #EBIT,@CSR      ;ENABLE INTERRUPT
010472  005037  001766                          CLR     YLOOP           ;WAIT
010476  005237  001766                  65$:    INC     YLOOP
010502  023727  001766  177777                  CMP     YLOOP,#-1
010510  001372                                  BNE     65$
010512  104005                                  ERROR 5                 ;NO INTERRUPT
010514  142777  000100  171304          2$:     BICB    #EBIT,@CSR      ;CLEAR CSR
010522  142777  000010  171276                  BICB    #TBIT,@CSR
010530  004737  012360                          JSR     PC,CLRINT       ;CLEAR ALL INTERRUPTS
010534  000004                                  SCOPE
010536  000207                                  RTS     PC              ;RETURN TO MONITOR

010540  022626                          1$:     CMP     (SP)+,(SP)+     ;ADJUST STOCK POINTER
010542  005037  002050                          CLR     TEMP1
010546  013704  000004                          MOV     ERRVEC,R4
010552  012737  011140  000004                  MOV     #10$,ERRVEC
010560  117737  171244  002046          7$:     MOVB    @IAR,TEMP       ;FIND WHICH I/O INTERRUPTED
010566  123737  002046  002006                  CMPB    TEMP,TADDR      ;IS IT MUT
010574  001421                                  BEQ     8$              ;YES
010576  053737  002042  002046                  BIS     BASE,TEMP       ;ASSEMBLE ADDRESS OF INTERRUPTING MODULE
010604  152777  000001  171214                  BISB    #RBIT,@CSR      ;CLEAR INTERRUPT
010612  105777  171230                          TSTB    @TEMP
010616  005237  002050                          INC     TEMP1
```

```
1748  010622  022737  000400  002050        CMP     #400,TEMP1
1749  010630  001353                        BNE     7$
1750  010632  104005                        ERROR 5                 ;NO INTERRUPT
1751  010634  000137  010514                JMP     2$
1752  010640  142777  000100  171160  8$:   BICB    #EBIT,@CSR
1753  010646  112737  000377  001124        MOVB    #377,$GDDAT
1754  010654  117737  171134  001126        MOVB    @COSADR,$BDDAT  ;TEST COS REGISTER
1755  010662  123737  001124  001126        CMPB    $GDDAT,$BDDAT   ;IS IT ALL ONES
1756  010670  001401                        BEQ     3$
1757  010672  104020                        ERROR 20                ;COS REGISTER ERROR
1758  010674  152777  000001  171124  3$:   BISB    #RBIT,@CSR
1759  010702  105777  171106                TSTB    @COSADR         ;CLEAR INTERRUPT
1760  010706  005037  001124                CLR     $GDDAT
1761  010712  117737  171076  001126        MOVB    @COSADR,$BDDAT
1762  010720  001401                        BEQ     4$
1763  010722  104013                        ERROR 13                ;RIF BIT NOT CLEARING FF
1764  010724  005237  002014        4$:     INC     COSADR
1765  010730  005237  002006                INC     TADDR           ;TEST NEXT BYTE
1766  010734  117737  171070  002046        MOVB    @IAR,TEMP
1767  010742  123737  002046  002006        CMPB    TEMP,TADDR      ;CHECK IF IT INTERRUPTED
1768  010750  001403                        BEQ     11$
1769  010752  104005                        ERROR 5                 ;NO INTERRUPT
1770  010754  000137  010514                JMP     2$
1771  010760  112737  000377  001124  11$:   MOVB    #377,$GDDAT
1772  010766  117737  171022  001126        MOVB    @COSADR,$BDDAT  ;CHECK IF COS REGISTER IS ALL ONES
1773  010774  123737  001124  001126        CMPB    $GDDAT,$BDDAT
1774  011002  001401                        BEQ     5$
1775  011004  104020                        ERROR 20                ;COS REGISTER ERROR
1776  011006  152777  000001  171012  5$:   BISB    #RBIT,@CSR
1777  011014  105777  170774                TSTB    @COSADR         ;CLEAR INTERRUPT
1778  011020  005037  001124                CLR     $GDDAT
1779  011024  117737  170764  001126        MOVB    @COSADR,$BDDAT
1780  011032  001401                        BEQ     6$
1781  011034  104013                        ERROR 13                ;RIF BIT NOT CLEARING FF
1782  011036  005337  002014        6$:     DEC     COSADR
1783  011042  005337  002006                DEC     TADDR           ;RESET ADDRESS
1784  011046  004737  012360                JSR     PC,CLRINT       ;CLEAR REMAINING INTERRUPTS
1785  011052  132777  000010  170746        BITB    #TBIT,@CSR      ;CHECK IF CLEARING T BIT CAUSE INTERRUPT
1786  011060  001615                        BEQ     2$
1787  011062  012746  000000                MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
1788  011066  012746  011074                MOV     #66$,-(SP)
1789  011072  000002                        RTI
1790  011074  000240        66$:    NOP
1791  011076  142777  000010  170722        BICB    #TBIT,@CSR      ;CLEAR INPUTS
1792  011104  152777  000100  170714        BISB    #EBIT,@CSR
1793  011112  005037  001766                CLR     YLOOP           ;WAIT
1794  011116  005237  001766        67$:    INC     YLOOP
1795  011122  023727  001766  177777        CMP     YLOOP,#-1
1796  011130  001372                        BNE     67$
1797  011132  104005                        ERROR 5                 ;NO INTERRUPT
1798  011134  000137  010514                JMP     2$
1799
1800  011140  022626        10$:    CMP     (SP)+,(SP)+     ;RESET STACK POINTER
1801  011142  104005                        ERROR 5                 ;NO INTERRUPT
1802  011144  000137  010514                JMP     2$
```

```
1803  011150  000240                    ANALOG: NOP
1804                                    .SBTTL  DIGITAL MODULE TEST ,M5012
1805
1806                                    ;;**********************************************************************
1807
1808                                            ;THIS TEST CHECKS M5012 MODULE
1809                                            ;ISOLATED 16 BIT DC INPUT
1810
1811                                    ;;**********************************************************************
1812
1813  011152                            M5012:
1814                                    ;;**********************************************************************
1815  011152  000004                    TST20:  SCOPE
1816  011154  012737  000020  002022            MOV     #20,TNUMB
1817  011162  152777  000020  170636            BISB    #DBIT,@CSR
1818  011170  112737  000000  001124            MOVB    #0,$GDDAT       ;SET WHAT DATA SHOULD BE
1819  011176  012737  000002  002004            MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
1820  011204  004737  011746                    JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
1821
1822  011210  152777  000010  170610            BISB    #TBIT,@CSR
1823  011216  112737  000377  001124            MOVB    #377,$GDDAT     ;SET WHAT DATA SHOULD BE
1824  011224  012737  000002  002004            MOV     #2,BYTNUM       ;SET NUMBER OF BYTES
1825  011232  004737  011746                    JSR     PC,TSTBYT       ;CHECK IF DATA IS CORRECT
1826  011236  142777  000010  170562            BICB    #TBIT,@CSR
1827  011244  004737  012360                    JSR     PC,CLRINT
1828
1829  011250  012746  000000                    MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
1830  011254  012746  011262                    MOV     #64$,-(SP)
1831  011260  000002                            RTI
1832  011262  000240                    64$:    NOP
1833  011264  012777  011362  170540            MOV     #6$,@VECT0      ;SET INTERRUPT VECTOR
1834  011272  012777  000340  170534            MOV     #PR7,@VECT0A
1835  011300  152777  000100  170520            BISB    #EBIT,@CSR      ;ENABLE INTERRUPT
1836  011306  152777  000010  170512            BISB    #TBIT,@CSR      ;START INTERRUPT
1837  011314  005037  001766                    CLR     YLOOP           ;WAIT
1838  011320  005237  001766            65$:    INC     YLOOP
1839  011324  023727  001766  177777            CMP     YLOOP,#-1
1840  011332  001372                            BNE     65$
1841  011334  104005                            ERROR 5                 ;NO INTERRUPT
1842  011336  142777  000100  170462    7$:     BICB    #EBIT,@CSR      ;CLEAR CSR
1843  011344  142777  000010  170454            BICB    #TBIT,@CSR
1844  011352  004737  012360                    JSR     PC,CLRINT       ;CLEAR ALL INTERRUPT
1845  011356  000004                            SCOPE
1846  011360  000207                            RTS     PC              ;RETURN TU MONITOR
1847
1848  011362  022626                    6$:     CMP     (SP)+,(SP)+     ;RESET STACK POINTER
1849  011364  005037  002050                    CLR     TEMP1
1850  011370  013704  000004                    MOV     ERRVEC,R4
1851  011374  012737  011626  000004            MOV     #4$,ERRVEC
1852  011402  117737  170422  002046    3$:     MOVB    @IAR,TEMP       ;FIND WHICH I/O INTERRUPTED
1853  011410  152777  000001  170410            BISB    #RBIT,@CSR
1854  011416  123737  002046  002006            CMPB    TEMP,TADDR      ;IS IT MUT
1855  011424  001415                            BEQ     2$              ;YES
1856  011426  053737  002042  002046            BIS     BASE,TEMP       ;ASSEMBLE ADDRESS OF INTERRUPTING I/O
1857  011434  105777  170406                    TSTB    @TEMP           ;CLEAR INTERRUPT
1858  011440  005237  002050                    INC     TEMP1
```

```
1859  011444  022737  000400  002050          CMP     #400,TEMP1
1860  011452  001353                          BNE     3$
1861  011454  104005                          ERROR 5                 ;NO INTERRUPT
1862  011456  000727                          BR      7$
1863  011460  105777  170322          2$:     TSTB    @TADDR          ;CLEAR INTERRUPT
1864  011464  005237  002006                  INC     TADDR
1865  011470  152777  000001  170330          BISB    #RBIT,@CSR
1866  011476  105777  170304                  TSTB    @TADDR          ;CLEAR INTERRUPT IN NEXT BYTE
1867  011502  005337  002006                  DEC     TADDR
1868  011506  132777  000200  170312          BITB    #FBIT,@CSR      ;CHECK IF INTERRUPT CLEAR
1869  011514  001413                          BEQ     8$
1870  011516  117737  170306  002046          MOVB    @IAR,TEMP
1871  011524  123737  002046  002006          CMPB    TEMP,TADDR
1872  011532  001002                          BNE     9$
1873  011534  104015                          ERROR   15              ;RIF BIT IS NOT CLEARING INTERRUPT
1874  011536  000430                          BR      5$
1875  011540  004737  012360          9$:     JSR     PC,CLRINT       ;CLEAR ALL REMINDING INTERRUPTS
1876  011544  132777  000010  170254  8$:     BITB    #TBIT,@CSR
1877  011552  001422                          BEQ     5$
1878  011554  012746  000000                  MOV     #PR0,-(SP)      ;SET PSW TO PRIORITY 0
1879  011560  012746  011566                  MOV     #66$,-(SP)
1880  011564  000002                          RTI
1881  011566  000240          66$:    NOP
1882  011570  142777  000010  170230          BICB    #TBIT,@CSR      ;CHECK IF CLEARING T BIT CAUSE INTERRUPT
1883  011576  005037  001766                  CLR     YLOOP           ;WAIT
1884  011602  005237  001766          67$:    INC     YLOOP
1885  011606  023727  001766  177777          CMP     YLOOP,#-1
1886  011614  001372                          BNE     67$
1887  011616  104005                          ERROR 5                 ;NO INTERRUPT
1888  011620  010437  000004          5$:     MOV     R4,ERRVEC
1889  011624  000644                          BR      7$
1890
1891  011626  022626          4$:     CMP     (SP)+,(SP)+     ;RESET STACK
1892  011630  104005                          ERROR 5                 ;NO INTERRUPT
1893  011632  000772                          BR      5$
```

```
  1894  011634  000137  003002          EXERC:   JMP     MONIT
```

```
1895  011640                          F5010:
1896  011640                          F5011:
1897  011640  012737  000004  002004          MOV     #4,BYTNUM       ;# OF BYTES PER MODULE
1898  011646  004737  012724                  JSR     PC,MONDAT       ;SUBROUTINE TO MONITOP DATA CONTINUOUSLY
1899
1900  011652                          F5012:
1901  011652  012737  000002  002004          MOV     #2,BYTNUM       ;# OF BYTES PER MODULE
1902  011660  004737  012724                  JSR     PC,MONDAT
1903
1904  011664                          F5013:
1905  011664  012737  000001  002004          MOV     #1,BYTNUM       ;# OF BYTES PER MODULE
1906  011672  004737  012724                  JSR     PC,MONDAT
1907
1908  011676                          F6010:
1909  011676  012737  000004  002004          MOV     #4,BYTNUM;MODULE IS 4 BYTE LONG
1910  011704  012700  000004                  MOV     #4,R0   ;SET A BYTE COUNTER
1911  011710  004737  012654                  JSR     PC,SETPTN
1912  011714  000207                          RTS     PC
1913
1914  011716                          F6012:
1915  011716                          F6013:
1916  011716  012737  000001  002004          MOV     #1,BYTNUM
1917  011724  004737  012654                  JSR     PC,SETPTN       ;ROUTINE TO OUTPUT ANY PATTERN TOOUT MODULE
1918  011730  000207                          RTS     PC
1919
1920  011732                          F6011:
1921  011732  012737  000002  002004          MOV     #2,BYTNUM       ;MODULE IS 2 BYTE LONG
1922  011740  004737  012654                  JSR     PC,SETPTN
1923  011744  000207                          RTS     PC
```

```
                                        .SBTTL  SUBROUTINES
                                                ;THIS SUBROUTINE IS USED TO CHECK IF
                                                ;A BITE OF DATA IN $GDDAT=$BDDAT.
                                                ;IT CHECKS AS MANY BYTES AS SPECIFIED
                                                ;BY     BYTNUM

011746                                  TSTBYT:
011746  013746  002004                          MOV     BYTNUM,-(SP)    ;;PUSH BYTNUM ON STACK
011752  013746  002006                          MOV     TADDR,-(SP)     ;;PUSH TADDR ON STACK
011756  117737  170024  001126          2$:     MOVB    @TADDR,$BDDAT   ;MOV DATA TO BDDAT
011764  123737  001124  001126                  CMPB    $GDDAT,$BDDAT   ;IS DATA OK
011772  001404                                  BEQ     1$
011774  013737  002006  002012                  MOV     TADDR,TBADDR
012002  104007                                  ERROR 7
012004  005237  002006                  1$:     INC     TADDR           ;NEXT BYTE
012010  005337  002004                          DEC     BYTNUM
012014  001360                                  BNE     2$
012016  012637  002006                          MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
012022  012637  002004                          MOV     (SP)+,BYTNUM    ;;POP STACK INTO BYTNUM
012026  000207                                  RTS     PC

012030                                  TSTONE:
012030  013746  002004                          MOV     BYTNUM,-(SP)    ;;PUSH BYTNUM ON STACK
012034  013746  002006                          MOV     TADDR,-(SP)     ;;PUSH TADDR ON STACK
012040  117737  167742  001126          2$:     MOVB    @TADDR,$BDDAT   ;MOV DATA TO BDDAT
012046  123737  001124  001126                  CMPB    $GDDAT,$BDDAT   ;IS DATA OK
012054  001404                                  BEQ     1$
012056  013737  002006  002012                  MOV     TADDR,TBADDR
012064  104011                                  ERROR 11
012066  005237  002006                  1$:     INC     TADDR           ;NEXT BYTE
012072  005337  002004                          DEC     BYTNUM
012076  001360                                  BNE     2$
012100  012637  002006                          MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
012104  012637  002004                          MOV     (SP)+,BYTNUM    ;;POP STACK INTO BYTNUM
012110  000207                                  RTS     PC
```

```
1959                                            ;THIS SUBROUTINE IS USED TO SET AS MANY
1960                                            ;BYTES AS SPECIFIED BY BYTNUM IN MUT
1961
1962  012112                            SETBYT:
1963  012112  013746  002004                   MOV     BYTNUM,-(SP)    ;;PUSH BYTNUM ON STACK
1964  012116  013746  002006                   MOV     TADDR,-(SP)     ;;PUSH TADDR ON STACK
1965  012122  113777  001124  167656   1$:     MOVB    $GDDAT,@TADDR   ;SET DATA IN MUT
1966  012130  005237  002006                   INC     TADDR
1967  012134  005337  002004                   DEC     BYTNUM
1968  012140  001370                           BNE     1$
1969  012142  012637  002006                   MOV     (SP)+,TADDR     ;;POP STACK INTO TADDR
1970  012146  012637  002004                   MOV     (SP)+,BYTNUM    ;;POP STACK INTO BYTNUM
1971  012152  000207                           RTS     PC
```

```
 1972
 1973                                        ;THESE TWO SUBROUTINES ARE USED IN MAPPING THE SYSTEM
 1974
 1975  012154  005004                GENCOD:  CLR     R4
 1976  012156  012703  001700                 MOV     #GENER,R3
 1977  012162  022301                3$:      CMP     (R3)+,R1
 1978  012164  001404                         BEQ     1$
 1979  012166  062704  000006                 ADD     #6,R4
 1980  012172  005713                         TST     (R3)
 1981  012174  001372                         BNE     3$
 1982  012176  000207                1$:      RTS     PC
 1983
 1984
 1985
 1986
 1987  012200  005004                TABLE:   CLR     R4
 1988  012202  012703  001700                 MOV     #GENER,R3
 1989  012206  022301                3$:      CMP     (R3)+,R1
 1990  012210  001404                         BEQ     1$
 1991  012212  062704  000002                 ADD     #2,R4
 1992  012216  005713                         TST     (R3)
 1993  012220  001372                         BNE     3$
 1994  012222  010022                1$:      MOV     R0,(R2)+
 1995  012224  010125                         MOV     R1,(R5)+
 1996  012226  000207                         RTS     PC
```

```
 1997                                                ;THIS SUBROUTINE CHECKS IF A BIT GET SET AND CLEAR IN CSR
 1998
 1999  012230  113777  001124  167570  BITSET:  MOVB    $GDDAT,@CSR     ;LOAD TESTED BIT
 2000  012236  005037  001126                   CLR     $BDDAT
 2001  012242  005037  001766                   CLR     YLOOP           ;WAIT
 2002  012246  005237  001766           64$:    INC     YLOOP
 2003  012252  023727  001766  000007         CMP     YLOOP,#7
 2004  012260  001372                         BNE     64$
 2005  012262  117737  167540  001126         MOVB    @CSR,$BDDAT     ;READ CSR
 2006  012270  023727  002022  000005         CMP     TNUMB,#5
 2007  012276  001003                         BNE     2$
 2008  012300  142737  000200  001126         BICB    #FBIT,$BDDAT
 2009  012306  123737  001124  001126  2$:    CMPB    $GDDAT,$BDDAT   ;COMPARE THEME
 2010  012314  001002                         BNE     1$
 2011  012316  062716  000002                 ADD     #2,(SP)         ;NO ERROR
 2012  012322  000337  001124           1$:    SWAB    $GDDAT
 2013  012326  000337  001126                  SWAB    $BDDAT
 2014  012332  042737  177400  001126         BIC     #177400,$BDDAT
 2015  012340  042737  177400  001124         BIC     #177400,$GDDAT
 2016  012346  000207                         RTS     PC
 2017
 2018                                                ;THIS IS INTERRUPT RUTINE FOR LINE CLOCK
 2019                                                ;WHEN DIAGNOSTIC DOES NOT USE IT
 2020
 2021  012350                           NOCLK:
 2022  012350  000002                         RTI
 2023
 2024                                                ;THIS IS INTERRUPT RUTINE FOR CLOCK TO COUNT TICKS
 2025
 2026  012352  005237  002044           COUNT:  INC     CLK
 2027  012356  000002                         RTI
 2028
```

```
2029                                                    ;THIS SUBROUTINE CLEARS UNEXPECTED INTERRUPTS FROM DBUS
2030  012360  005037  002052          CLRINT:  CLR     TEMP3
2031  012364  005037  002054                   CLR     TEMP4
2032  012370  132777  000200  167430  2$:      BITB    #FBIT,@CSR      ;IS INTERRUPT PENDING?
2033  012376  001422                           BEQ     1$              ;NO
2034  012400  117737  167424  002052           MOVB    @IAR,TEMP3
2035  012406  053737  002042  002052           BIS     BASE,TEMP3
2036  012414  152777  000001  167404           BISB    #RBIT,@CSR
2037  012422  105777  167424                   TSTB    @TEMP3
2038  012426  005237  002054                   INC     TEMP4
2039  012432  032737  000400  002054           BIT     #BIT8,TEMP4
2040  012440  001753                           BEQ     2$
2041  012442  104023                           ERROR   23              ;UNABLE TO CLEAR INTERRUPT
2042  012444  000207                 1$:       RTS     PC
2043
2044
2045                                                    ;THIS SUBROUTINE CHECKS FOR CONTROL C
2046
2047
2048  012446  117746  166474          CNTRC:   MOVB    @$TKB,-(SP)
2049  012452  042716  000200                   BIC     #BIT7,(SP)      ;CLEAR PARITY BIT
2050  012456  122716  000003                   CMPB    #3,(SP)         ;CONTROL C?
2051  012462  001030                           BNE     1$              ;NO
2052  012464  152777  000002  167334           BISB    #CBIT,@CSR      ;SET C BIT
2053  012472  005037  001766                   CLR     YLOOP           ;WAIT
2054  012476  005237  001766          64$:     INC     YLOOP
2055  012502  023727  001766  000007           CMP     YLOOP,#7
2056  012510  001372                           BNE     64$
2057  012512  152777  000002  167306           BISB    #CBIT,@CSR      ;DO IT AGAIN
2058  012520  005037  001766                   CLR     YLOOP           ;WAIT
2059  012524  005237  001766          65$:     INC     YLOOP
2060  012530  023727  001766  000007           CMP     YLOOP,#7
2061  012536  001372                           BNE     65$
2062  012540  000137  003002                   JMP     MONIT
2063  012544  005726                 1$:       TST     (SP)+
2064  012546  000207                           RTS     PC
2065
2066
2067  012550  004737  012446          KBINT:   JSR     PC,CNTRC
2068  012554  000002                           RTI
```

```
2069                                        ;THIS SUBROUTINE IS USED IN THE LOOP TEST
2070                                        ;AND IT IS SIMILAR TO THE TSTBYT ROUTINE
2071
2072  012556                          CHKBYT:
2073  012556  013746  002050                   MOV     TEMP1,-(SP)     ;;PUSH TEMP1 ON STACK
2074  012562  013746  002004                   MOV     BYTNUM,-(SP)    ;;PUSH BYTNUM ON STACK
2075  012566  117737  167256  001126  2$:      MOVB    @TEMP1,$BDDAT   ;READ DATA FROM INPUT MODULE
2076  012574  123737  001124  001126           CMPB    $GDDAT,$BDDAT
2077  012602  001410                          BEQ     1$
2078  012604  013737  002006  002012          MOV     TADDR,TBADDR
2079  012612  012637  002004                   MOV     (SP)+,BYTNUM    ;;POP STACK INTO BYTNUM
2080  012616  012637  002050                   MOV     (SP)+,TEMP1     ;;POP STACK INTO TEMP1
2081  012622  000207                          RTS     PC              ;ERROR
2082  012624  005237  002050          1$:      INC     TEMP1           ;GO TO NEXT BYTE
2083  012630  005337  002004                   DEC     BYTNUM
2084  012634  001354                          BNE     2$
2085  012636  012637  002004                   MOV     (SP)+,BYTNUM    ;;POP STACK INTO BYTNUM
2086  012642  012637  002050                   MOV     (SP)+,TEMP1     ;;POP STACK INTO TEMP1
2087  012646  062716  000002                   ADD     #2,(R6)         ;BYPASS ERROR IN MAINLINE CODE
2088  012652  000207                          RTS     PC
2089
```

```
                                        ;THIS SUBROUTINE ALLOWS FIELD ENGINEER TO SELECT ANY
                                        ;DATA PATTERN FOR OUTPUT MODULES
                                        
                                        
012654  142777  000020  167144  SETPTN: BICB    #DBIT,@CSR
012662  142777  000004  167136          BICB    #GBIT,@CSR
012670  104401  021323          1$:     TYPE    ,MASS19         ;SELECT A DATA PATTERN
012674  104410                          RDOCT
012676  012637  001124                  MOV     (SP)+,$GDDAT    ;;POP STACK INTO $GDDAT
012702  113777  001124  167076          MOVB    $GDDAT,@TADDR   ;OUTPUT PATTERN TO MODULE
012710  005237  002006                  INC     TADDR
012714  005337  002004                  DEC     BYTNUM
012720  001363                          BNE     1$
012722  000207                          RTS     PC
                                        
                                        ;THIS SUBROUTINE IS USED IN FIELD TEST TO CONTINUOUSLY
                                        ;MONITOR DATA FROM INPUT MODULE. DATA IS PRINTED ONLY
                                        ;DURING FIRST PASS UNLESS THERE IS A CHANGE IN THE DATA
012724  005001                  MONDAT: CLR     R1              ;CLEAR PASS REGISTER
012726  013700  002004          6$:     MOV     BYTNUM,R0       ;# OF BYTES PER MODULE
012732  005300                  5$:     DEC     R0
012734  117737  167046  001124  1$:     MOVB    @TADDR,$GDDAT   ;STORE DATA
012742  005701                          TST     R1              ;IS IT FIRST PASS?
012744  001016                          BNE     4$              ;NO
012746  117760  167034  001770  2$:     MOVB    @TADDR,DBUFF(R0)        ;STORE DATA IN TABLE
012754  013746  002006                  MOV     TADDR,-(SP)     ;;SAVE TADDR FOR TYPEOUT
012760  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012762  104401  021645                  TYPE    ,MASS27         ;COLON & TAB
012766  013746  001124                  MOV     $GDDAT,-(SP)    ;;SAVE $GDDAT FOR TYPEOUT
012772  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012774  104401  020417                  TYPE    ,MASSO          ;CR & LF
013000  000404                          BR      3$
013002  123760  001124  001770  4$:     CMPB    $GDDAT,DBUFF(R0)        ;ANY CHANGE IN DATA ?
013010  001356                          BNE     2$              ;YES
013012  005237  002006          3$:     INC     TADDR           ;GO TO NEXT BYTE IN MODULE
013016  005700                          TST     R0              ;LAST BYTE IN MODULE ?
013020  001344                          BNE     5$              ;NO
013022  163737  002004  002006          SUB     BYTNUM,TADDR    ;RESTORE ADDRESS OF MUT
013030  004737  012446                  JSR     PC,CNTRC        ;TYPE CONTROL C TO RETURN TO MONITOR
013034  012701  000001                  MOV     #1,R1           ;NEW PASS
013040  000732                          BR      6$
```

```
2131
2132                                        ;THIS SUBROUTINE IS USED TO WRAP AROUND DATA "LOOP TEST"
2133
2134
2135  013042                          LOPTST:
2136  013042  152777  000002  166756          BISB    #CBIT,@CSR      ;SET C BIT
2137  013050  005037  001766                  CLR     YLOOP           ;WAIT
2138  013054  005237  001766          64$:    INC     YLOOP
2139  013060  023727  001766  000007          CMP     YLOOP,#7
2140  013066  001372                          BNE     64$
2141  013070  152777  000002  166730          BISB    #CBIT,@CSR      ;DO IT AGAIN
2142  013076  005037  001766                  CLR     YLOOP           ;WAIT
2143  013102  005237  001766          65$:    INC     YLOOP
2144  013106  023727  001766  000007          CMP     YLOOP,#7
2145  013114  001372                          BNE     65$
2146  013116  020127  000141                  CMP     R1,#141 ;IS IT M5010?
2147  013122  001131                          BNE     1$              ;YES
2148  013124  012737  000125  001124          MOV     #125,$GDDAT
2149  013132  012737  000004  002004          MOV     #4,BYTNUM
2150  013140  004737  012112                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
2151  013144  005037  001766                  CLR     YLOOP           ;WAIT
2152  013150  005237  001766          66$:    INC     YLOOP
2153  013154  023727  001766  001777          CMP     YLOOP,#1777
2154  013162  001372                          BNE     66$
2155  013164  012737  000004  002004          MOV     #4,BYTNUM
2156  013172  004737  012556                  JSR     PC,CHKBYT       ;COMPARE DATA
2157  013176  104021                          ERROR   21
2158  013200  012737  000252  001124          MOV     #252,$GDDAT
2159  013206  012737  000004  002004          MOV     #4,BYTNUM
2160  013214  004737  012112                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
2161  013220  005037  001766                  CLR     YLOOP           ;WAIT
2162  013224  005237  001766          67$:    INC     YLOOP
2163  013230  023727  001766  001777          CMP     YLOOP,#1777
2164  013236  001372                          BNE     67$
2165  013240  012737  000004  002004          MOV     #4,BYTNUM
2166  013246  004737  012556                  JSR     PC,CHKBYT       ;COMPARE DATA
2167  013252  104021                          ERROR   21
2168  013254  012737  000377  001124          MOV     #377,$GDDAT
2169  013262  012737  000004  002004          MOV     #4,BYTNUM
2170  013270  004737  012112                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
2171  013274  005037  001766                  CLR     YLOOP           ;WAIT
2172  013300  005237  001766          68$:    INC     YLOOP
2173  013304  023727  001766  001777          CMP     YLOOP,#1777
2174  013312  001372                          BNE     68$
2175  013314  012737  000004  002004          MOV     #4,BYTNUM
2176  013322  004737  012556                  JSR     PC,CHKBYT       ;COMPARE DATA
2177  013326  104021                          ERROR   21
2178  013330  012737  000000  001124          MOV     #0,$GDDAT
2179  013336  012737  000004  002004          MOV     #4,BYTNUM
2180  013344  004737  012112                  JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
2181  013350  005037  001766                  CLR     YLOOP           ;WAIT
2182  013354  005237  001766          69$:    INC     YLOOP
2183  013360  023727  001766  001777          CMP     YLOOP,#1777
2184  013366  001372                          BNE     69$
2185  013370  012737  000004  002004          MOV     #4,BYTNUM
2186  013376  004737  012556                  JSR     PC,CHKBYT       ;COMPARE DATA
```

```
2187  013402  104021                        ERROR   21
2188  013404  000530                        BR      2$
2189  013406                          1$:
2190  013406  012737  000125  001124        MOV     #125,$GDDAT
2191  013414  012737  000002  002004        MOV     #2,BYTNUM
2192  013422  004737  012112                JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
2193  013426  005037  001766                CLR     YLOOP           ;WAIT
2194  013432  005237  001766          70$:  INC     YLOOP
2195  013436  023727  001766  001777        CMP     YLOOP,#1777
2196  013444  001372                        BNE     70$
2197  013446  012737  000002  002004        MOV     #2,BYTNUM
2198  013454  004737  012556                JSR     PC,CHKBYT       ;COMPARE DATA
2199  013460  104021                        ERROR   21
2200  013462  012737  000252  001124        MOV     #252,$GDDAT
2201  013470  012737  000002  002004        MOV     #2,BYTNUM
2202  013476  004737  012112                JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
2203  013502  005037  001766                CLR     YLOOP           ;WAIT
2204  013506  005237  001766          71$:  INC     YLOOP
2205  013512  023727  001766  001777        CMP     YLOOP,#1777
2206  013520  001372                        BNE     71$
2207  013522  012737  000002  002004        MOV     #2,BYTNUM
2208  013530  004737  012556                JSR     PC,CHKBYT       ;COMPARE DATA
2209  013534  104021                        ERROR   21
2210  013536  012737  000377  001124        MOV     #377,$GDDAT
2211  013544  012737  000002  002004        MOV     #2,BYTNUM
2212  013552  004737  012112                JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
2213  013556  005037  001766                CLR     YLOOP           ;WAIT
2214  013562  005237  001766          72$:  INC     YLOOP
2215  013566  023727  001766  001777        CMP     YLOOP,#1777
2216  013574  001372                        BNE     72$
2217  013576  012737  000002  002004        MOV     #2,BYTNUM
2218  013604  004737  012556                JSR     PC,CHKBYT       ;COMPARE DATA
2219  013610  104021                        ERROR   21
2220  013612  012737  000000  001124        MOV     #0,$GDDAT
2221  013620  012737  000002  002004        MOV     #2,BYTNUM
2222  013626  004737  012112                JSR     PC,SETBYT       ;OUTPUT DATAT PATTERN
2223  013632  005037  001766                CLR     YLOOP           ;WAIT
2224  013636  005237  001766          73$:  INC     YLOOP
2225  013642  023727  001766  001777        CMP     YLOOP,#1777
2226  013650  001372                        BNE     73$
2227  013652  012737  000002  002004        MOV     #2,BYTNUM
2228  013660  004737  012556                JSR     PC,CHKBYT       ;COMPARE DATA
2229  013664  104021                        ERROR   21
2230  013666  000207                    2$: RTS     PC
```

```
                                        .SBTTL  SCOPE HANDLER ROUTINE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS.  IT WILL INCREMENT
                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG. (DISPLAY<7:0>)
                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*CALL
                                        ;*      SCOPE             ;;SCOPE=IOT

013670                                  $SCOPE:
                                        ;#####START OF CODE FOR THE XOR TESTER#####
013670  000416                  $XTSTR: BR      6$                ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
013672  013746  000004                  MOV     @#ERRVEC,-(SP)    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
013676  012737  013716  000004          MOV     #5$,@#ERRVEC      ;;SET FOR TIMEOUT
013704  005737  177060                  TST     @#177060          ;;TIME OUT ON XOR?
013710  012637  000004                  MOV     (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
013714  000421                          BR      $SVLAD            ;;GO TO THE NEXT TEST
013716  022626                  5$:     CMP     (SP)+,(SP)+       ;;CLEAR THE STACK AFTER A TIME OUT
013720  012637  000004                  MOV     (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
013724  000407                          BR      7$                ;;LOOP ON THE PRESENT TEST
013726                          6$: ;#####END OF CODE FOR THE XOR TESTER#####
013726  105737  001103          2$:     TSTB    $ERFLG            ;;HAS AN ERROR OCCURRED?
013732  001412                          BEQ     $SVLAD            ;;BR IF NO
013734  032777  001000  165176          BIT     #BIT09,@SWR       ;;LOOP ON ERROR?
013742  001404                          BEQ     4$                ;;BR IF NO
013744  013737  001110  001106  7$:     MOV     $LPERR,$LPADR     ;;SET LOOP ADDRESS TO LAST SCOPE
013752  000415                          BR      $OVER
013754  105037  001103          4$:     CLRB    $ERFLG            ;;ZERO THE ERROR FLAG
013760  105237  001102          $SVLAD: INCB    $TSTNM            ;;COUNT TEST NUMBERS
013764  011637  001106                  MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
013770  011637  001110                  MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
013774  005037  001172                  CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
014000  112737  000001  001115          MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
014006  013777  001102  165126  $OVER:  MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
014014  013716  001106                  MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
014020  000002                          RTI                       ;;FIXES PS
014022                          $SW08TBL:
014022  005416                          .WORD   TST1+2            ;;STARTING ADDRESS OF TEST 1
014024  005550                          .WORD   TST2+2            ;;STARTING ADDRESS OF TEST 2
014026  005614                          .WORD   TST3+2            ;;STARTING ADDRESS OF TEST 3
014030  005660                          .WORD   TST4+2            ;;STARTING ADDRESS OF TEST 4
014032  006054                          .WORD   TST5+2            ;;STARTING ADDRESS OF TEST 5
014034  006124                          .WORD   TST6+2            ;;STARTING ADDRESS OF TEST 6
014036  006170                          .WORD   TST7+2            ;;STARTING ADDRESS OF TEST 7
014040  006234                          .WORD   TST10+2           ;;STARTING ADDRESS OF TEST 10
014042  006400                          .WORD   TST11+2           ;;STARTING ADDRESS OF TEST 11
014044  006674                          .WORD   TST12+2           ;;STARTING ADDRESS OF TEST 12
014046  006776                          .WORD   TST13+2           ;;STARTING ADDRESS OF TEST 13
014050  007442                          .WORD   TST14+2           ;;STARTING ADDRESS OF TEST 14
014052  007634                          .WORD   TST15+2           ;;STARTING ADDRESS OF TEST 15
014054  010026                          .WORD   TST16+2           ;;STARTING ADDRESS OF TEST 16
014056  010324                          .WORD   TST17+2           ;;STARTING ADDRESS OF TEST 17
014060  011154                          .WORD   TST20+2           ;;STARTING ADDRESS OF TEST 20
```

```
                                        .SBTTL  TTY INPUT ROUTINE
                                        ;;**********************************************************************
                                        .ENABL  LSB

                                        .DSABL  LSB


                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                        ;*CALL:
                                        ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                        ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                        ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                        ;

014062  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
014064  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
014072  105777  165046          1$:     TSTB    @$TKS           ;;WAIT FOR
014076  100375                          BPL     1$              ;;A CHARACTER
014100  117766  165042  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
014106  042766  177600  000004          BIC     #C<177>,4(SP)   ;;GET RID OF JUNK IF ANY
014114  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
014122  001013                          BNE     3$              ;;BRANCH IF NO
014124  105777  165014          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
014130  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
014132  117746  165010                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
014136  042716  177600                  BIC     #C177,(SP)      ;;MAKE IT 7-BIT ASCII
014142  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
014146  001366                          BNE     2$              ;;IF NOT DISCARD IT
014150  000750                          BR      1$              ;;YES, RESUME
014152  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
014160  002407                          BLT     4$              ;;BRANCH IF YES
014162  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
014170  003003                          BGT     4$              ;;BRANCH IF YES
014172  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
014200  000002                  4$:     RTI                     ;;GO BACK TO USER
                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                        ;*CALL:
                                        ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                        ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                        ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

014202  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
014204  012703  014310          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
014210  022703  014320          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
014214  101405                          BLOS    4$              ;;BR IF YES
014216  104406                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
014220  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
014222  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
014226  001003                          BNE     3$              ;;SKIP IF NOT
014230  104401  001174          4$:     TYPE    ,$QUES          ;;TYPE A '?'
014234  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
014236  111337  014306          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
014242  104401  014306                  TYPE    ,9$
```

```
2343  014246  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2344  014252  001356                          BNE     2$              ;;LOOP IF NOT RETURN
2345  014254  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2346  014260  104401  001176                  TYPE    ,$LF            ;;TYPE A LINE FEED
2347  014264  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2348  014266  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2349  014270  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2350  014276  012766  014310  000004          MOV     #$TTYIN,4(SP)
2351  014304  000002                          RTI                     ;;RETURN
2352  014306     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR.  TO TYPE
2353  014307     000                          .BYTE   0               ;;TERMINATOR
2354  014310  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
2355  014320  052536  005015     000  $CNTLU: .ASCIZ  / U/<15><12>    ;;CONTROL "U"
2356  014325     136  006507  000012  $CNTLG: .ASCIZ  / G/<15><12>    ;;CONTROL "G"
2357  014332  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
2358  014340  020075     000
2359  014343     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
2360  014350  036440  000040
2361                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
2362
2363                                  ;;**************************************************************
2364                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
2365                                  ;*CHANGE IT TO BINARY.
2366                                  ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
2367                                  ;*OCTAL DIGITS.  IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
2368                                  ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED.  THE COMPLETE NUMBER MUST
2369                                  ;*THEN BE RETYPED.  THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
2370                                  ;*CALL:
2371                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
2372                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
2373                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
2374
2375  014354  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
2376  014356  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
2377  014364  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2378  014366  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2379  014370  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2380  014372  104407                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
2381  014374  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
2382  014376  010037  014502                  MOV     R0,5$           ;;AND SAVE IT
2383  014402  005001                          CLR     R1              ;;CLEAR DATA WORD
2384  014404  005002                          CLR     R2
2385  014406  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
2386  014410  001420                          BEQ     3$              ;;IF ZERO GET OUT
2387  014412  122716  000060                  CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
2388  014416  003026                          BGT     4$              ;;IS AN OCTAL DIGIT
2389  014420  122716  000067                  CMPB    #'7,(SP)
2390  014424  002423                          BLT     4$
2391  014426  006301                          ASL     R1              ;;*2
2392  014430  006102                          ROL     R2
2393  014432  006301                          ASL     R1              ;;*4
2394  014434  006102                          ROL     R2
2395  014436  006301                          ASL     R1              ;;*8
2396  014440  006102                          ROL     R2
2397  014442  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
2398  014446  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
```

```
014450  000756                          BR      2$                      ;;LOOP
014452  005726                  3$:     TST     (SP)+                   ;;CLEAN TERMINATOR FROM STACK
014454  010166  000012                  MOV     R1,12(SP)               ;;SAVE THE RESULT
014460  010237  014512                  MOV     R2,$HIOCT
014464  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
014466  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
014470  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
014472  000002                          RTI                             ;;RETURN
014474  005726                  4$:     TST     (SP)+                   ;;CLEAN PARTIAL FROM STACK
014476  105010                          CLRB    (R0)                    ;;SET A TERMINATOR
014500  104401                          TYPE                            ;;TYPE UP THRU THE BAD CHAR.
014502  000000                  5$:     .WORD   0
014504  104401  001174                  TYPE    ,$QUES                  ;;"?" "CR" & "LF"
014510  000730                          BR      1$                      ;;TRY AGAIN
014512  000000                  $HIOCT: .WORD   0                       ;;HIGH ORDER BITS GO HERE
                                .SBTTL  ERROR HANDLER ROUTINE

                                ;;*************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO $ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1          HALT ON ERROR
                                ;*SW13=1          INHIBIT ERROR TYPEOUTS
                                ;*SW09=1          LOOP ON ERROR
                                ;*CALL
                                ;*       ERROR    N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

014514                          $ERROR:
014514  105237  001103          7$:     INCB    $ERFLG                  ;;SET THE ERROR FLAG
014520  001775                          BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
014522  013777  001102  164412          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
014530  005237  001112                  INC     $ERTTL                  ;;INC THE ERROR COUNT
014534  011637  001116                  MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
014540  162737  000002  001116          SUB     #2,$ERRPC
014546  117737  164344  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
014554  032777  020000  164356          BIT     #BIT13,@SWR             ;;SKIP TYPEOUT IF SET
014562  001004                          BNE     20$                     ;;SKIP TYPEOUTS
014564  004737  014646                  JSR     PC,$ERRTYP              ;;GO TO USER ERROR ROUTINE
014570  104401  001175                  TYPE    ,$CRLF
014574                          20$:
014574  005777  164340          2$:     TST     @SWR                    ;;HALT ON ERROR
014600  100001                          BPL     3$                      ;;SKIP IF CONTINUE
014602  000000                          HALT                            ;;HALT ON ERROR!
014604  032777  001000  164326  3$:     BIT     #BIT09,@SWR             ;;LOOP ON ERROR SWITCH SET?
014612  001402                          BEQ     4$                      ;;BR IF NO
014614  013716  001110                  MOV     $LPERR,(SP)             ;;FUDGE RETURN FOR LOOPING
014620  005737  001172          4$:     TST     $ESCAPE                 ;;CHECK FOR AN ESCAPE ADDRESS
014624  001402                          BEQ     5$                      ;;BR IF NONE
014626  013716  001172                  MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
014632                          5$:
014632  022737  004300  000042          CMP     #$ENDAD,@#42            ;;ACT-11 AUTO-ACCEPT?
014640  001001                          BNE     6$                      ;;BRANCH IF NO
014642  000000                          HALT                            ;;YES
014644                          6$:
014644  000002                          RTI                             ;;RETURN
```

```
                                .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.

014646                          $ERRTYP:
014646  104401  001175                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
014652  010046                        MOV     R0,-(SP)        ;;SAVE R0
014654  005000                        CLR     R0              ;;PICKUP THE ITEM INDEX
014656  153700  001114                BISB    @#$ITEMB,R0
014662  001004                        BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                ;;TYPE THE PC OF THE ERROR
014664  013746  001116                MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;ERROR ADDRESS
014670  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
014672  000426                        BR      6$              ;;GET OUT
014674  005300                1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
014676  006300                        ASL     R0              ;;      WORK FOR THE ERROR TABLE
014700  006300                        ASL     R0
014702  006300                        ASL     R0
014704  062700  002112                ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
014710  012037  014720                MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
014714  001404                        BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
014716  104401                        TYPE                    ;;TYPE THE "ERROR MESSAGE"
014720  000000                2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
014722  104401  001175                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
014726  012037  014736        3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
014732  001404                        BEQ     5$              ;;SKIP TYPEOUT IF 0
014734  104401                        TYPE                    ;;TYPE THE "DATA HEADER"
014736  000000                4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
014740  104401  001175                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
014744  011000                5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
014746  001004                        BNE     7$              ;;GO TYPE THE DATA
014750  012600                6$:     MOV     (SP)+,R0        ;;RESTORE R0
014752  104401  001175                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
014756  000207                        RTS     PC              ;;RETURN
014760                          7$:
014760  013046                        MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
014762  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
014764  005710                        TST     (R0)            ;;IS THERE ANOTHER NUMBER?
014766  001770                        BEQ     6$              ;;BR IF NO
014770  104401  014776                TYPE    ,8$             ;;TYPE TWO(2) SPACES
014774  000771                        BR      7$              ;;LOOP
014776  020040      000       8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
        015002                        .EVEN
                                .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.
                                ;*
```

```
2511                                    ;*CALL:
2512                                    ;*1) USING A TRAP INSTRUCTION
2513                                    ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2514                                    ;*OR
2515                                    ;*      TYPE
2516                                    ;*      MESADR
2517                                    ;*
2518
2519  015002  105737  001157           $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
2520  015006  100002                          BPL     1$              ;;BR IF YES
2521  015010  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
2522  015012  000407                          BR      3$              ;;LEAVE
2523  015014  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
2524  015016  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
2525  015022  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
2526  015024  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
2527  015026  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
2528  015030  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
2529  015032  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
2530  015036  000002                          RTI                     ;;RETURN
2531  015040  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
2532  015044  001430                          BEQ     8$
2533  015046  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
2534  015052  001006                          BNE     5$
2535  015054  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
2536  015056  104401                          TYPE                    ;;TYPE A CR AND LF
2537  015060  001175                          $CRLF
2538  015062  105037  015216                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
2539  015066  000755                          BR      2$              ;;GET NEXT CHARACTER
2540  015070  004737  015152          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
2541  015074  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
2542  015100  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
2543  015102  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
2544                                                                  ;;AND THE NULL CHAR.
2545  015106  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2546  015112  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2547  015114  004737  015152                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
2548  015120  105337  015216                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2549  015124  000770                          BR      7$              ;;LOOP
2550
2551                                    ;HORIZONTAL TAB PROCESSOR
2552
2553  015126  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2554  015132  004737  015152          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2555  015136  132737  000007  015216          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2556  015144  001372                          BNE     9$              ;;TAB STOP
2557  015146  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
2558  015150  000724                          BR      2$              ;;GET NEXT CHARACTER
2559  015152  105777  163772          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2560  015156  100375                          BPL     $TYPEC
2561  015160  116677  000002  163764          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2562  015166  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2563  015174  001003                          BNE     1$              ;;BRANCH IF NO
2564  015176  105037  015216                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2565  015202  000406                          BR      $TYPEX          ;;EXIT
2566  015204  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
```

```
 2567  015212  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 2568  015214  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 2569  015216  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 2570  015220  000207                  $TYPEX: RTS     PC
 2571
 2572                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
 2573
 2574                                  ;;**************************************************************
 2575                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
 2576                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
 2577                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
 2578                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
 2579                                  ;*REPLACED WITH SPACES.
 2580                                  ;*CALL:
 2581                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
 2582                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
 2583
 2584  015222                          $TYPDS:
 2585  015222  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 2586  015224  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 2587  015226  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 2588  015230  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 2589  015232  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 2590  015234  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 2591  015240  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 2592  015244  100004                          BPL     1$              ;;BR IF INPUT IS POS.
 2593  015246  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
 2594  015250  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
 2595  015256  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
 2596  015260  012703  015436                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
 2597  015264  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
 2598  015270  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
 2599  015272  016001  015426                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
 2600  015276  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
 2601  015300  002402                          BLT     4$              ;;BR IF DONE
 2602  015302  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
 2603  015304  000774                          BR      3$
 2604  015306  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
 2605  015310  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
 2606  015312  001002                          BNE     5$              ;;FALL THROUGH IF 0
 2607  015314  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
 2608  015316  100407                          BMI     7$              ;;BR IF YES
 2609  015320  106316                  5$:     ASLB    (SP)            ;;MSD?
 2610  015322  103003                          BCC     6$              ;;BR IF NO
 2611  015324  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
 2612  015332  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
 2613  015336  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 2614  015342  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 2615  015344  005720                          TST     (R0)+           ;;JUST !NCREMENTING
 2616  015346  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
 2617  015352  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
 2618  015354  003002                          BGT     8$              ;;GO TO EXIT
 2619  015356  010502                          MOV     R5,R2           ;;GET THE LSD
 2620  015360  000764                          BR      6$              ;;GO CHANGE TO ASCII
 2621  015362  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 2622  015364  100003                          BPL     9$              ;;BR IF NO
```

```
2623  015366  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
2624  015374  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
2625  015376  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
2626  015400  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
2627  015402  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2628  015404  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2629  015406  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2630  015410  104401  015436                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
2631  015414  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
2632  015422  012616                          MOV     (SP)+,(SP)
2633  015424  000002                          RTI                     ;;RETURN TO USER
2634  015426  023420                  $DTBL:  10000.
2635  015430  001750                          1000.
2636  015432  000144                          100.
2637  015434  000012                          10.
2638  015436  000004                  $DBLK:  .BLKW   4
2639                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
2640
2641                                  ;;**********************************************************************
2642                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2643                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
2644                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2645                                  ;*CALL:
2646                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2647                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
2648                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2649                                  ;*      .BYTE   M               ;;M=1 OR 0
2650                                  ;*                                      ;;1=TYPE LEADING ZEROS
2651                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
2652                                  ;*
2653                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2654                                  ;*$TYPOS OR $TYPOC
2655                                  ;*CALL:
2656                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2657                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
2658                                  ;*
2659                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
2660                                  ;*CALL:
2661                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2662                                  ;*      TYPOC                   ;;CALL FOR TYPEOUT
2663
2664  015446  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
2665  015452  116637  000001  015671          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
2666  015460  112637  015673                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
2667  015464  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
2668  015470  000406                          BR      $TYPON
2669  015472  112737  000001  015671  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
2670  015500  112737  000006  015673          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
2671  015506  112737  000005  015670  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
2672  015514  010346                          MOV     R3,-(SP)        ;;SAVE R3
2673  015516  010446                          MOV     R4,-(SP)        ;;SAVE R4
2674  015520  010546                          MOV     R5,-(SP)        ;;SAVE R5
2675  015522  113704  015673                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
2676  015526  005404                          NEG     R4
2677  015530  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
2678  015534  110437  015672                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
```

```
2679  015540  113704  015671              MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2680  015544  016605  000012              MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
2681  015550  005003                      CLR     R3              ;;CLEAR THE OUTPUT WORD
2682  015552  006105              1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2683  015554  000404                      BR      3$              ;;GO DO MSB
2684  015556  006105              2$:     ROL     R5              ;;FORM THIS DIGIT
2685  015560  006105                      ROL     R5
2686  015562  006105                      ROL     R5
2687  015564  010503                      MOV     R5,R3
2688  015566  006103              3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2689  015570  105337  015672              DECB    $OMODE          ;;TYPE THIS DIGIT?
2690  015574  100016                      BPL     7$              ;;BR IF NO
2691  015576  042703  177770              BIC     #177770,R3      ;;GET RID OF JUNK
2692  015602  001002                      BNE     4$              ;;TEST FOR 0
2693  015604  005704                      TST     R4              ;;SUPPRESS THIS 0?
2694  015606  001403                      BEQ     5$              ;;BR IF YES
2695  015610  005204              4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2696  015612  052703  000060              BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2697  015616  052703  000040      5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2698  015622  110337  015666              MOVB    R3,8$           ;;SAVE FOR TYPING
2699  015626  104401  015666              TYPE    ,8$             ;;GO TYPE THIS DIGIT
2700  015632  105337  015670      7$:     DECB    $OCNT           ;;COUNT BY 1
2701  015636  003347                      BGT     2$              ;;BR IF MORE TO DO
2702  015640  002402                      BLT     6$              ;;BR IF DONE
2703  015642  005204                      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2704  015644  000744                      BR      2$              ;;GO DO THE LAST DIGIT
2705  015646  012605              6$:     MOV     (SP)+,R5        ;;RESTORE R5
2706  015650  012604                      MOV     (SP)+,R4        ;;RESTORE R4
2707  015652  012603                      MOV     (SP)+,R3        ;;RESTORE R3
2708  015654  016666  000002  000004      MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2709  015662  012616                      MOV     (SP)+,(SP)
2710  015664  000002                      RTI                     ;;RETURN
2711  015666     000              8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2712  015667     000                      .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2713  015670     000              $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2714  015671     000              $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2715  015672  000000              $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2716                                      .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
2717
2718                                      ;;**************************************************************
2719                                      ;*SAVE R0-R5
2720                                      ;*CALL:
2721                                      ;*      SAVREG
2722                                      ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
2723                                      ;*
2724                                      ;*TOP---(+16)
2725                                      ;* +2---(+18)
2726                                      ;* +4---R5
2727                                      ;* +6---R4
2728                                      ;* +8---R3
2729                                      ;*+10---R2
2730                                      ;*+12---R1
2731                                      ;*+14---R0
2732
2733  015674                              $SAVREG:
2734  015674  010046                              MOV     R0,-(SP)        ;;PUSH R0 ON STACK
```

```
2735  015676  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2736  015700  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2737  015702  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2738  015704  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2739  015706  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2740  015710  016646  000022                MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
2741  015714  016646  000022                MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
2742  015720  016646  000022                MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
2743  015724  016646  000022                MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
2744  015730  000002                        RTI
2745
2746                                  ;*RESTORE R0-R5
2747                                  ;*CALL:
2748                                  ;*      RESREG
2749  015732                          $RESREG:
2750  015732  012666  000022                MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
2751  015736  012666  000022                MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
2752  015742  012666  000022                MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
2753  015746  012666  000022                MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
2754  015752  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
2755  015754  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
2756  015756  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
2757  015760  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
2758  015762  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
2759  015764  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
2760  015766  000002                        RTI
2761                                  .SBTTL  TRAP DECODER
2762
2763                                  ;;**************************************************************
2764                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2765                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2766                                  ;*OF THE DESIRED ROUTINE.  THEN USING THE ADDRESS OBTAINED IT WILL
2767                                  ;*GO TO THAT ROUTINE.
2768
2769  015770  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2770  015772  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
2771  015776  005740                        TST     -(R0)           ;;BACKUP BY 2
2772  016000  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2773  016002  006300                        ASL     R0              ;;POSITION FOR INDEXING
2774  016004  016000  016024                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2775  016010  000200                        RTS     R0              ;;GO TO ROUTINE
2776
2777
2778                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
2779
2780  016012  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
2781  016014  016666  000004  000002        MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
2782  016022  000002                        RTI                     ;;RESTORE THE PSW
2783
2784                                  .SBTTL  TRAP TABLE
2785
2786                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2787                                  ;*BY THE "TRAP" INSTRUCTION.
2788
2789                                  ;       ROUTINE
2790                                  ;       -------
```

```
                                        $TRPAD: .WORD   $TRAP2
                                                $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
                                                $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
                                                $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
                                                $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
                                                $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

                                                $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
                                                $RDLIN  ;;CALL=RDLIN    TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
                                                $RDOCT  ;;CALL=RDOCT    TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
                                                $SAVREG ;;CALL=SAVREG   TRAP+11(104411) SAVE R0-R5 ROUTINE
                                                $RESREG ;;CALL=RESREG   TRAP+12(104412) RESTORE R0-R5 ROUTINE
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;**********************************************************************
                                        ;POWER DOWN ROUTINE
016052  012737  016212  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
016060  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
016066  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
016070  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
016072  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
016074  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
016076  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
016100  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
016102  017746  163032                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
016106  010637  016216                  MOV     SP,$SAVR6       ;;SAVE SP
016112  012737  016124  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
016120  000000                          HALT
016122  000776                          BR      .-2             ;;HANG UP

                                        ;;**********************************************************************
                                        ;POWER UP ROUTINE
016124  012737  016212  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
016132  013706  016216                  MOV     $SAVR6,SP       ;;GET SP
016136  005037  016216                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
016142  005237  016216          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
016146  001375                          BNE     1$              ;;OF  WORD
016150  012677  162764                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
016154  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
016156  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
016160  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
016162  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
016164  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
016166  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
016170  012737  016052  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
016176  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
016204  104401                          TYPE                    ;REPORT THE POWER FAILURE
016206  016220          $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
016210  000002                  RTI
016212  000000          $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
016214  000776                  BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
016216  000000          $SAVR6: 0                       ;;PUT THE SP HERE
016220  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
016226  000122
                                        .EVEN
```

```
2847  016230  041103  052111  047040  EM1:    .ASCIZ  /CBIT NOT CLEARING IOCM/
2848  016236  052117  041440  042514
2849  016244  051101  047111  020107
2850  016252  047511  046503     000
2851  016257     111  041517  020115  EM2:    .ASCII  /IOCM TEST/<15><12>
2852  016264  042524  052123  005015
2853  016272  051503  020122  040504  EM2Y:   .ASCII  /CSR DATA ERROR WHEN TESTING /
2854  016300  040524  042440  051122
2855  016306  051117  053440  042510
2856  016314  020116  042524  052123
2857  016322  047111  020107
2858  016326  020040  044502  020124  EM2X:   .ASCIZ  /  BIT  /<15><12>
2859  016334  006440  000012
2860  016340  041520  052011  052123  DH1:    .ASCIZ  /PC     TST#    GDDAT   BDDAT   PASS/
2861  016346  004443  042107  040504
2862  016354  004524  042102  040504
2863  016362  004524  040520  051523
2864  016370     000
2865  016371     111  041517  020115  EM3:    .ASCIZ  /IOCM IS NOT RESPONDING/<15><12>
2866  016376  051511  047040  052117
2867  016404  051040  051505  047520
2868  016412  042116  047111  006507
2869  016420  000012
2870  016422  041104  051525  041040  EM4:    .ASCIZ  /DBUS BIT IS FAILING/<15><12>
2871  016430  052111  044440  020123
2872  016436  040506  046111  047111
2873  016444  006507  000012
2874  016450  047516  044440  052116  EM5:    .ASCIZ  /NO INTERRUPT /<15><12>
2875  016456  051105  052522  052120
2876  016464  006440  000012
2877  016470  041520  052011  052123  DH5:    .ASCIZ  /PC     TST#    MODULE  ADDR    PASS/
2878  016476  004443  047515  052504
2879  016504  042514  020040  040440
2880  016512  042104  004522  040520
2881  016520  051523     000
2882  016523     127  047522  043516  EM6:    .ASCIZ  /WRONG PATTERN IN IAR AFTER INTERRUPT/<15><12>
2883  016530  050040  052101  042524
2884  016536  047122  044440  020116
2885  016544  040511  020122  043101
2886  016552  042524  020122  047111
2887  016560  042524  051122  050125
2888  016566  006524  000012
2889  016572                          EM17:
2890  016572  040504  040524  042440  EM7:    .ASCIZ  /DATA ERROR/<15><12>
2891  016600  051122  051117  005015
2892  016606     000
2893  016607     120  004503  047515  DH7:    .ASCIZ  /PC     MODULE  ADDRESS TST#    GDDAT   BDDAT   PASS/
2894  016614  052504  042514  040411
2895  016622  042104  042522  051523
2896  016630  052011  052123  004443
2897  016636  042107  040504  004524
2898  016644  042102  040504  004524
2899  016652  040520  051523     000
2900  016657     124  051505  020124  EM10:   .ASCIZ  /TEST ABORTED, IOCM ERROR/<15><12>
2901  016664  041101  051117  042524
2902  016672  026104  044440  041517
```

```
 2903  016700  020115  051105  047522
 2904  016706  006522  000012
 2905  016712  040504  040524  047040  EM11:   .ASCIZ  /DATA NOT CLEAR AFTER 10 SEC/<15><12>
 2906  016720  052117  041440  042514
 2907  016726  051101  040440  052106
 2908  016734  051105  030440  020060
 2909  016742  042523  006503  000012
 2910  016750  047111  042524  051122  EM12:   .ASCIZ  /INTERRUPT TOO LATE   /<15><12>
 2911  016756  050125  020124  047524
 2912  016764  020117  040514  042524
 2913  016772  020040  006440  000012
 2914  017000  044522  020106  044502  EM13:   .ASCIZ  /RIF BIT NOT CLEARING COS REGISTERS /<15><12>
 2915  017006  020124  047516  020124
 2916  017014  046103  040505  044522
 2917  017022  043516  041440  051517
 2918  017030  051040  043505  051511
 2919  017036  042524  051522  006440
 2920  017044  000012
 2921  017046  047516  044440  052116  EM14:   .ASCIZ  /NO INTERRUPT IN MAINTENANCE MODE/<15><12>
 2922  017054  051105  052522  052120
 2923  017062  044440  020116  040515
 2924  017070  047111  042524  040516
 2925  017076  041516  020105  047515
 2926  017104  042504  005015     000
 2927  017111     122  043111  041040  EM15:   .ASCIZ  /RIF BIT ISN"T CLEARING INTERRUPT/<15><12>
 2928  017116  052111  044440  047123
 2929  017124  052042  041440  042514
 2930  017132  051101  047111  020107
 2931  017140  047111  042524  051122
 2932  017146  050125  006524  000012
 2933  017154  046103  040505  044522  EM16:   .ASCIZ  /CLEARING DBIT ISN"T CLEARING CSR/<15><12>
 2934  017162  043516  042040  044502
 2935  017170  020124  051511  021116
 2936  017176  020124  046103  040505
 2937  017204  044522  043516  041440
 2938  017212  051123  005015     000
 2939  017217     103  051517  051040  EM20:   .ASCIZ  /COS REGISTER DATA ERROR./<15><12>
 2940  017224  043505  051511  042524
 2941  017232  020122  040504  040524
 2942  017240  042440  051122  051117
 2943  017246  005015     000
 2944  017251     114  047517  020120  EM21:   .ASCIZ  /LOOP ERROR ON 6010-5010 OR 5011./<15><12>
 2945  017256  051105  047522  020122
 2946  017264  047117  033040  030460
 2947  017272  026460  030065  030061
 2948  017300  047440  020122  030065
 2949  017306  030461  005015     000
 2950  017313     120  004503  042107  DH21:   .ASCIZ  /PC     GDDAT   BDDAT   PASS/
 2951  017320  040504  004524  042102
 2952  017326  040504  004524  040520
 2953  017334  051523     000
 2954  017337     101  042104  042522  EM22:   .ASCIZ  /ADDRESS TEST WITH MBIT SET ISN'T WORKING /<15><12>
 2955  017344  051523  052040  051505
 2956  017352  020124  044527  044124
 2957  017360  046440  044502  020124
 2958  017366  042523  020124  051511
```

```
 2959  017374  023516  020124  047527
 2960  017402  045522  047111  020107
 2961  017410  005015     000
 2962  017413     125  040516  046102  EM23:   .ASCIZ  /UNABLE TO CLEAR INTERRUPT /<15><12>
 2963  017420  020105  047524  041440
 2964  017426  042514  051101  044440
 2965  017434  052116  051105  052522
 2966  017442  052120  006440  000012
 2967  017450  041520  052011  052123  DH23:   .ASCIZ  /PC     TST#    IAR/
 2968  017456  004443  040511  000122
 2969  017464  006516  000012          M1:     .ASCIZ  /N/<15><12>
 2970  017470  006531  000012          M0:     .ASCIZ  /Y/<15><12>
 2971  017474  000122                  M2:     .ASCIZ  /R/
 2972  017476  020107     000          M3:     .ASCIZ  /G /
 2973  017501     124  000040          M4:     .ASCIZ  /T /
 2974  017504  020104     000          M5:     .ASCIZ  /D /
 2975  017507     115  000040          M6:     .ASCIZ  /M /
 2976  017512  020105     000          M7:     .ASCIZ  /E /
 2977  017515     040  040520  051523  M11:    .ASCIZ  / PASSES COMPLETED/<15><12>
 2978  017522  051505  041440  046517
 2979  017530  046120  052105  042105
 2980  017536  005015     000
 2981  017541     115  042117  046125  EM24:   .ASCIZ  /MODULE UNDER TEST IS NOT RESPONDING./<15><12>
 2982  017546  020105  047125  042504
 2983  017554  020122  042524  052123
 2984  017562  044440  020123  047516
 2985  017570  020124  042522  050123
 2986  017576  047117  044504  043516
 2987  017604  005015     000
 2988  017607     015  052012  051505  M13:    .ASCIZ  <15><12>/TEST OPTIONS: /<15><12>
 2989  017614  020124  050117  044524
 2990  017622  047117  035123  005015
 2991  017630     000
 2992  017631     123  020040  054523  M14:    .ASCIZ  /S  SYSTEM TEST/<15><12>
 2993  017636  052123  046505  052040
 2994  017644  051505  006524  000012
 2995  017652  020104  042040  043511  M15:    .ASCIZ  /D  DIGITAL MODULE/<15><12>
 2996  017660  052111  046101  046440
 2997  017666  042117  046125  006505
 2998  017674  000012
 2999  017676  020101  040440  040516  M16:    .ASCIZ  /A  ANALOG MODULE./<15><12>
 3000  017704  047514  020107  047515
 3001  017712  052504  042514  005015
 3002  017720     000
 3003  017721     115  020040  040515  M17:    .ASCIZ  /M  MAP OF DBUS INTERFACES/<15><12>
 3004  017726  020120  043117  042040
 3005  017734  052502  020123  047111
 3006  017742  042524  043122  041501
 3007  017750  051505  005015     000
 3008  017755     130  020040  054105  M18:    .ASCIZ  /X  EXERCIZER/<15><12>
 3009  017762  051105  044503  042532
 3010  017770  006522  000012
 3011  017774  020111  044440  041517  M19:    .ASCIZ  /I  IOCM TEST/<15><12>
 3012  020002  020115  042524  052123
 3013  020010  005015     000
 3014  020013     040  020040  006477  M20:    .ASCIZ  /  ?/<15><12>
```

```
 3015  020020  000012
 3016  020022  020124  051440  052105  M21:    .ASCIZ  /T  SET SWREG/<15><12>
 3017  020030  051440  051127  043505
 3018  020036  005015     000
 3019  020041     061  030060  030060  M22:    .ASCII  /100000   HALT ON ERROR/<15><12>
 3020  020046  020060  020040  040510
 3021  020054  052114  047440  020116
 3022  020062  051105  047522  006522
 3023  020070     012
 3024  020071     064  030060  030060          .ASCII  /40000    LOOP ON TEST/<15><12>
 3025  020076  020040  020040  047514
 3026  020104  050117  047440  020116
 3027  020112  042524  052123  005015
 3028  020120  030062  030060  020060          .ASCIZ  /20000    INHIBIT ERROR & EOP PRINT/<15><12>
 3029  020126  020040  044440  044116
 3030  020134  041111  052111  042440
 3031  020142  051122  051117  023040
 3032  020150  042440  050117  050040
 3033  020156  044522  052116  005015
 3034  020164     000
 3035  020165     061  030060  030060          .ASCIZ  /10000    LOOP ON ERROR/<15><12>
 3036  020172  020040  020040  047514
 3037  020200  050117  047440  020116
 3038  020206  051105  047522  006522
 3039  020214  000012
 3040  020216  005015  053523  042522  M23:    .ASCIZ  <15><12>/SWREG =/
 3041  020224  020107  000075
 3042  020230  020106  043040  042511  M24:    .ASCIZ  /F  FIELD TEST/<15><12>
 3043  020236  042114  052040  051505
 3044  020244  006524  000012
 3045  020250  020127  046040  047517  M25:    .ASCIZ  /W  LOOP TEST/<15><12>
 3046  020256  020120  042524  052123
 3047  020264  005015     000
 3048  020267     114  020040  047514  M26:    .ASCIZ  /L  LOAD ITERATION COUNT/<15><12>
 3049  020274  042101  044440  042524
 3050  020302  040522  044524  047117
 3051  020310  041440  052517  052116
 3052  020316  005015     000
 3053  020321     116  046525  042502  M27:    .ASCIZ  /NUMBER OF ITERATIONS(OCT) = /
 3054  020326  020122  043117  044440
 3055  020334  042524  040522  044524
 3056  020342  047117  024123  041517
 3057  020350  024524  036440  000040
 3058  020356  044514  042516  041440  EM25:   .ASCIZ  /LINE CLOCK IS NOT INTERRUPTING/<15><12>
 3059  020364  047514  045503  044440
 3060  020372  020123  047516  020124
 3061  020400  047111  042524  051122
 3062  020406  050125  044524  043516
 3063  020414  005015     000
 3064  020417     015  000012          MASS0:  .ASCIZ  <15><12>
 3065  020422  042101  051104  051505  MASS2:  .ASCIZ  /ADDRESS  /
 3066  020430  020123  000040
 3067  020434  020040  046440  030065  MASS3:  .ASCIZ  /   M5010 NONISOLATED 32 BIT DC INPUT/<15><12>
 3068  020442  030061  047040  047117
 3069  020450  051511  046117  052101
 3070  020456  042105  031440  020062
```

```
3071  020464  044502  020124  041504
3072  020472  044440  050116  052125
3073  020500  005015     000
3074  020503     040  020040  032515  MASS4:  .ASCIZ  /  M5011 NONISOLATED 16 BIT DC INPUT/<15><12>
3075  020510  030460  020061  047516
3076  020516  044516  047523  040514
3077  020524  042524  020104  033061
3078  020532  041040  052111  042040
3079  020540  020103  047111  052520
3080  020546  006524  000012
3081  020552  020040  046440  030065  MASS5:  .ASCIZ  /  M5012 ISOLATED 16 BIT DC INPUT/<15><12>
3082  020560  031061  044440  047523
3083  020566  040514  042524  020104
3084  020574  033061  041040  052111
3085  020602  042040  020103  047111
3086  020610  052520  006524  000012
3087  020616  020040  046440  030065  MASS6:  .ASCIZ  /  M5013 8 BIT AC INPUT/<15><12>
3088  020624  031461  034040  041040
3089  020632  052111  040440  020103
3090  020640  047111  052520  006524
3091  020646  000012
3092  020650  020040  046440  030066  MASS7:  .ASCIZ  /  M6010 NONISOLATED 32 BIT DC OUTPUT/<15><12>
3093  020656  030061  047040  047117
3094  020664  051511  046117  052101
3095  020672  042105  031440  020062
3096  020700  044502  020124  041504
3097  020706  047440  052125  052520
3098  020714  006524  000012
3099  020720  020040  046440  030066  MASS8:  .ASCIZ  /  M6011 16 BIT ONESHOT OUTPUT/<15><12>
3100  020726  030461  030440  020066
3101  020734  044502  020124  047117
3102  020742  051505  047510  020124
3103  020750  052517  050124  052125
3104  020756  005015     000
3105  020761     040  020040  033115  MASS9:  .ASCIZ  /  M6012 ISOLATED 8 BIT DC OUTPUT/<15><12>
3106  020766  030460  020062  051511
3107  020774  046117  052101  042105
3108  021002  034040  041040  052111
3109  021010  042040  020103  052517
3110  021016  050124  052125  005015
3111  021024     000
3112  021025     040  020040  033115  MASS10: .ASCIZ  /  M6013 8 BIT AC OUTPUT/<15><12>
3113  021032  030460  020063  020070
3114  021040  044502  020124  041501
3115  021046  047440  052125  052520
3116  021054  006524  000012
3117  021060  005015  020040  052440  MASS11: .ASCIZ  <15><12>/  UNKNOWN GENERIC CODE  /
3118  021066  045516  047516  047127
3119  021074  043440  047105  051105
3120  021102  041511  041440  042117
3121  021110  020105  000040
3122  021114  005015  044127  041511  MASS12: .ASCIZ  <15><12>/WHICH TEST OPTION ?  /
3123  021122  020110  042524  052123
3124  021130  047440  052120  047511
3125  021136  020116  020077  000040
3126  021144  054524  042520  040440  MASS13: .ASCIZ  /TYPE ADDRESS OF MUT    /
```

```
3127  021152  042104  042522  051523
3128  021160  047440  020106  052515
3129  021166  020124  020040  000040
3130  021174  047111  042524  051122  MASS14: .ASCIZ  /INTERRUPT TOO LATE/<15><12>
3131  021202  050125  020124  047524
3132  021210  020117  040514  042524
3133  021216  005015     000
3134  021221     105  042116  047440  MASS15: .ASCIZ  /END OF MODULE TEST/<15><12>
3135  021226  020106  047515  052504
3136  021234  042514  052040  051505
3137  021242  006524  000012
3138  021246  047105  020104  043117  MASS16: .ASCIZ  /END OF 10CM TEST/<15><12>
3139  021254  044440  041517  020115
3140  021262  042524  052123  005015
3141  021270     000
3142  021271     105  042116  047440  MASS17: .ASCIZ  /END OF  AUTO TEST/<15><12>
3143  021276  020106  040440  052125
3144  021304  020117  042524  052123
3145  021312  005015     000
3146  021315     040  020040  005015  MASS18: .ASCIZ  /   /<15><12>
3147  021322     000
3148  021323     124  050131  020105  MASS19: .ASCIZ  /TYPE ONE BYTE DATA PATTERN    /
3149  021330  047117  020105  054502
3150  021336  042524  042040  052101
3151  021344  020101  040520  052124
3152  021352  051105  020116  020040
3153  021360  000040
3154  021362  054524  042520  041440  MASS20: .ASCIZ  /TYPE CONTROL-C TO RETURN TO MONITOR/<15><12>
3155  021370  047117  051124  046117
3156  021376  041455  052040  020117
3157  021404  042522  052524  047122
3158  021412  052040  020117  047515
3159  021420  044516  047524  006522
3160  021426  000012
3161  021430  054524  042520  040440  MASS21: .ASCIZ  /TYPE ADDRESS OF OUTPUT MUT/
3162  021436  042104  042522  051523
3163  021444  047440  020106  052517
3164  021452  050124  052125  046440
3165  021460  052125     000
3166  021463     124  050131  020105  MASS22: .ASCIZ  /TYPE ADDRESS OF  INPUT MUT/
3167  021470  042101  051104  051505
3168  021476  020123  043117  044440
3169  021504  050116  052125  046440
3170  021512  052125     000
3171  021515     103  047117  042516  MASS23: .ASCIZ  /CONNECT OUTPUT TO INPUT MODULE/<15><12>
3172  021522  052103  047440  052125
3173  021530  052520  020124  047524
3174  021536  044440  050116  052125
3175  021544  046440  042117  046125
3176  021552  006505  000012
3177  021556  047105  020104  043117  MASS25: .ASCIZ  /END OF LOOP TEST/<15><12>
3178  021564  046040  047517  020120
3179  021572  042524  052123  005015
3180  021600     000
3181  021601     127  047522  043516  MASS26: .ASCIZ  /WRONG MODULE-OUTPUT MUST BE M6010/<15><12>
3182  021606  046440  042117  046125
```

```
3183  021614  026505  052517  050124
3184  021622  052125  046440  051525
3185  021630  020124  042502  046440
3186  021636  030066  030061  005015
3187  021644     000
3188  021645     072  000011          MASS27: .ASCIZ  /:      /
3189  021650  051127  047117  020107  MASS28: .ASCIZ  /WRONG MODULE-INPUT MUST BE EITHER M5010 OR M5011/<15><12>
3190  021656  047515  052504  042514
3191  021664  044455  050116  052125
3192  021672  046440  051525  020124
3193  021700  042502  042440  052111
3194  021706  042510  020122  032515
3195  021714  030460  020060  051117
3196  021722  046440  030065  030461
3197  021730  005015     000
3198          021734                  .EVEN
3199  021734  001116  002002  002050  DT17:   .WORD   $ERRPC, $MUT,   TEMP1,  TNUMB,  $GDDAT, $BDDAT, PASS,
3200  021742  002022  001124  001126
3201  021750  002056  000000
3202  021754  001116  002002  002014  DT20:   .WORD   $ERRPC, $MUT,   COSADR, TNUMB,  $GDDAT, $BDDAT, PASS,
3203  021762  002022  001124  001126
3204  021770  002056  000000
3205  021774  001116  002022  002052  DT23:   .WORD   $ERRPC, TNUMB,  TEMP3,
3206  022002  000000
3207  022004  001116  002002  002012  DT7:    .WORD   $ERRPC, $MUT,   TBADDR, TNUMB,  $GDDAT, $BDDAT, PASS,
3208  022012  002022  001124  001126
3209  022020  002056  000000
3210  022024  001116  002022  001124  DT1:    .WORD   $ERRPC, TNUMB,  $GDDAT, $BDDAT, PASS,
3211  022032  001126  002056  000000
3212  022040  001116  002022  002002  DT5:    .WORD   $ERRPC, TNUMB,  $MUT,   'TADDR, PASS,
3213  022046  002006 .002056  000000
3214  022054  001116  001124  001126  DT21:   .WORD   $ERRPC, $GDDAT, $BDDAT, PASS.
3215  022062  002056  000000
3216          000001                  .END
```

```
ANALOG  011150            791    1803#
ANSW    001764            557#
ATABL   001200            529#    910     954
AUTO    003612            758     785     898#
BASE    002042            582#    908    1129    1338    1508    1744    1856    2035
BITSET  012230           1234    1237    1246    1249    1259    1262    1293    1296    1306    1309    1319    1322    1999#
BIT0  = 000001            455#    478
BIT00 = 000001            445#    455
BIT01 = 000002            444#    454
BIT02 = 000004            443#    453
BIT03 = 000010            442#    452
BIT04 = 000020            441#    451
BIT05 = 000040            440#    450
BIT06 = 000100            439#    449
BIT07 = 000200            438#    448
BIT08 = 000400            437#    447
BIT09 = 001000            436#    446    2257    2443
BIT1  = 000002            454#    477
BIT10 = 002000            435#
BIT11 = 004000            434#
BIT12 = 010000            433#
BIT13 = 020000            432#    857     969    1039    1190    2435
BIT14 = 040000            431#    862     974    1044    1195
BIT15 = 100000            430#
BIT2  = 000004            453#    476
BIT3  = 000010            452#    475
BIT4  = 000020            451#    474
BIT5  = 000040            450#    473
BIT6  = 000100            449#    472     750     761     777     821     900    1016    1071    1128    1185
BIT7  = 000200            448#    471    2049
BIT8  = 000400            447#   2039
BIT9  = 001000            446#
BPTVEC= 000014            462#
BTABL   001440            530#    911     943     949     962
BYTNUM  002004            567#   1441*   1446*   1469*   1475*   1606*   1608*   1613*   1615*   1619*   1621*   1626*   1628*
                         1655*   1657*   1677*   1679*   1710*   1715*   1819*   1824*   1897*   1901*   1905*   1909*   1916*
                         1921*   1931    1939*   1942*   1946    1954*   1957*   1963    1967*   1970*   2074    2079*   2083*
                         2085*   2101*   2109    2127    2149*   2155*   2159*   2165*   2169*   2175*   2179*   2185*   2191*
                         2197*   2201*   2207*   2211*   2217*   2221*   2227*
CBIT  = 000002            477#    840     845    1052    1057    1083    1088    1208    1213    1275    1280    1352    1357
                         1403    1408    2052    2057    2136    2141
CHKBYT  012556           2072#   2156    2166    2176    2186    2198    2208    2218    2228
CLK     002044            583#   1647*   1662    1670    1675*   1685    2026*
CLKADR  002024            575#    750*
CLKVC   002062            590#   1648*   1690*
CLKVCA  002064            591#   1649*
CLRINT  012360           1034    1298    1449    1473    1478    1496    1522    1717    1733    1784    1827    1844    1875
                         2030#
CNTRC   012446            853     965    1035    1096   2048#    2067    2128
COSADR  002014            571#   1706*   1707*   1754    1759    1761    1764*   1772    1777    1779    1782*   3202
COUNT   012352           1648    2026#
CR    = 000015            370#   2562    2572
CRLF  = 000200            371#    746    2533    2572
CSR     002026            576#    827     832*    840*    845*    902     909*    939*   1023*   1052*   1057*   1076*   1083*
                         1088*   1130*   1207*   1208*   1213*   1218    1264*   1265*   1270    1275*   1280*   1337*   1352*
                         1357*   1383*   1397*   1398    1399    1403*   1408*   1439*   1444*   1448*   1467*   1472*   1477*
```

```
                    1487*  1488*  1494*  1495*  1505*  1515   1523   1529*  1554*  1604*  1646*  1708*  1713*
                    1724*  1725*  1731*  1732*  1745*  1752*  1758*  1776*  1785   1791*  1792*  1817*  1822*
                    1826*  1835*  1836*  1842*  1843*  1853*  1865*  1868   1876   1882*  1999*  2005   2032
                    2036*  2052*  2057*  2094*  2095*  2136*  2141*
DBIT   = 000020      474#   1257   1265   1439   1467   1554   1604   1646   1708   1817   2094
DBUFF    001770      559#   1562*  1563*  1564*  1572   1574   2114*  2122
DDISP  = 177570      377#    509    725
DH1      016340      618    622    630    638    2860#
DH21     017313      682   2950#
DH23     017450      690   2967#
DH5      016470      634    654   2877#
DH7      016607      642    650    674    678    2893#
DIGIT    003222      788    819#
DISPLA   001142      509#   725*   733*  2267*  2430*
DISPRE   000174      347#   733
DMUT     002016      572#   850*   852    953*   958   1093*  1094
DSWR   = 177570      376#    508    724
DT1      022024      619    623    631    639    3210#
DT17     021734      675   3199#
DT20     021754      679   3202#
DT21     022054      683   3214#
DT23     021774      691   3205#
DT5      022040      635    655   3212#
DT7      022004      643    651   3207#
EBIT   = 000100      472#   1232   1383   1488   1494   1725   1731   1752   1792   1835   1842
EMTVEC= 000030      465#    712*   713*
EM1      016230      617   2847#
EM10     016657      645   2900#
EM11     016712      649   2905#
EM12     016750      653   2910#
EM13     017000      657   2914#
EM14     017046      661   2921#
EM15     017111      665   2927#
EM16     017154      669   2933#
EM17     016572      673   2889#
EM2      016257      621   2851#
EM2X     016326     1233*  1245*  1258*  1292*  1305*  1318*  2858#
EM2Y     016272     2853#
EM20     017217      677   2939#
EM21     017251      681   2944#
EM22     017337      685   2954#
EM23     017413      689   2962#
EM24     017541      693   2981#
EM25     020356      697   3058#
EM3      016371      625   2865#
EM4      016422      629   2870#
EM5      016450      633   2874#
EM6      016523      637   2882#
EM7      016572      641   2890#
ERRVEC= 000004       458#    722    723*   734*   748    749*   754*   825    826*   830*   874*   878*   906
                     907*   938*  1074   1075*  1106*  1110*  1125   1126*  1170*  1183   1198*  1206*  1377
                    1378*  1413*  1502   1503*  1535*  1739   1740*  1850   1851*  1888*  2246   2247*  2249*
                    2252*
EXERC    011634      797   1894#
FBIT   = 000200      471#   1244   1399   1515   1868   2008   2032
FIELD    004610      806   1069#
```

```
F5010   011640            592    1895#
F5011   011640            593    1896#
F5012   011652            594    1900#
F5013   011664            595    1904#
F6010   011676            596    1908#
F6011   011732            597    1920#
F6012   011716            598    1914#
F6013   011716            599    1915#
GBIT  = 000004            476#    832     909     939    1023    1076    1130    1304    2095
GENCOD  012154           1133    1975#
GENER   001700            531#    836     943     946    1080    1976    1988
GNS   = ****** U          346     745    2792    2793    2794    2795    2796    2799    2800    2801    2802    2803
HT    = 000011            368#   2531    2572
IAR     002030            577#    929     935    1163    1168    1391    1504    1517    1741    1766    1852    1870    2034
IOCM    005302            800    1182#
IOTVEC= 000020            463#    710*    711*
KBINT   012550            760     820     899    1015    1070    1127    1184    2067#
KBVEC   002110            601#    760*    820*    899*   1015*   1070*   1127*   1184*
LF    = 000012            369#   2566    2572
LONGIN  002020            573#
LOOP    004314            809    1014#
LOPTST  013042           1033    2135#
MAPE    005054            794    1125#
MASS0   020417            873     886     951    1105    1157    2120    3064#
MASS10  021025           1152    3112#
MASS11  021060            870     948    1102    1154    3117#
MASS12  021114            778    3122#
MASS13  021144            822    1069    3126#
MASS14  021174           3130#
MASS15  021221            864    3134#
MASS16  021246           1197    3138#
MASS17  021271            978    3142#
MASS18  021315            781*    782    3146#
MASS19  021323           2096    3148#
MASS2   020422           1134    3065#
MASS20  021362           1095    3154#
MASS21  021430           1017    3161#
MASS22  021463           1020    3166#
MASS23  021515           1014    3171#
MASS25  021556           1046    3177#
MASS26  021601           1048    3181#
MASS27  021645           2117    3188#
MASS28  021650           1050    3189#
MASS3   020434           1138    3067#
MASS4   020503           1140    3074#
MASS5   020552           1142    3081#
MASS6   020616           1144    3087#
MASS7   020650           1146    3092#
MASS8   020720           1148    3099#
MASS9   020761           1150    3105#
MBIT  = 000040            473#   1244    1337    1383
MOD     002066            592#   1079
MODUL   001742            548#    834     953
MONDAT  012724           1898    1902    1906    2108#
MONIT   003002            776#    815     817     865     875     879     887     952     979    1047    1062    1107    1111
                         1171    1199    1894    2062
```

```
MUT     001722            540#    835     955
M0      017470           2970#
M1      017464           2969#
M11     017515            861     973    1043    1194    2977#
M13     017607            766    2988#
M14     017631            767    2992#
M15     017652            768    2995#
M16     017676           2999#
M17     017721            769    3003#
M18     017755           3008#
M19     017774            770    3011#
M2      017474           1318    2971#
M20     020013            816    3014#
M21     020022            771    3016#
M22     020041            882    3019#
M23     020216            883    3040#
M24     020230            772    3042#
M25     020250            773    3045#
M26     020267            774    3048#
M27     020321            812    3053#
M3      017476           1305    2972#
M4      017501           1292    2973#
M5      017504           1258    2974#
M5010   006672            548    1435#
M5011   010322            549    1702#
M5012   011152            550    1813#
M5013   006774            551    1463#
M6      017507           1245    2975#
M6010   007440            552    1550#
M6011   010024            553    1640#
M6011A  010216           1669    1674#
M6011B  010310           1665    1684    1689#
M6012   007632            554    1599#
M6013   007632            555    1600#
M7      017512           1233    2976#
NOCLK   012350            352    1690    2021#
PASCNT  002060            589#    814*    855     859     966     971    1037    1041    1188    1192
PASS    002056            588#    851*    854*    855     898*    964*    966     968*   1032*   1036*   1037    1182*   1187*
                         1188    3199    3202    3207    3210    3212    3214
PATT    001774            561#   1564    1567
PIRQ  = 177772            375#
PIRQVE= 000240            469#
PR0   = 000000            392#    762    1379    1414    1481    1525    1650    1718    1787    1829    1878
PR1   = 000040            393#
PR2   = 000100            394#
PR3   = 000140            395#
PR4   = 000200            396#
PR5   = 000240            397#
PR6   = 000300            398#
PR7   = 000340            354     399#   1376    1486    1649    1723    1834
PS    = 177776            372#    373
PSW   = 177776            373#
PWRVEC= 000024            464#   2808*   2809*   2818*   2824*   2836*   2837*
RBIT  = 000001            478#   1317    1397    1505    1745    1758    1776    1853    1865    2036
RDCHR = 104406            779    2335    2799#
RDLIN = 104407           2380    2800#
```

```
RDOCT = 104410          813     823     884     1018    1021    1072    2097    2801#
RESREG= 104412         2803#
RESVEC= 000010          459#
SAVREG= 104411         2802#
SETBYT  012112         1607    1614    1620    1627    1656    1678    1962#   2150    2160    2170    2180    2192    2202
                       2212    2222
SETCLK  002640          358     747#
SETPTN  012654         1911    1917    1922    2094#
STACK = 001100          363#    708
START   002362          350     701#
STKLMT= 177774          374#
SWR     001140          508#    706     724*    726     732*    2257    2435    2440    2443    2816    2829*
SWREG   000176          348#    732     857     862     885*     969     974    1039    1044    1190    1195
SWREGS  003564          803     882#
SW0   = 000001          427#
SW00  = 000001          417#    427
SW01  = 000002          416#    426
SW02  = 000004          415#    425
SW03  = 000010          414#    424
SW04  = 000020          413#    423
SW05  = 000040          412#    422
SW06  = 000100          411#    421
SW07  = 000200          410#    420
SW08  = 000400          409#    419
SW09  = 001000          408#    418
SW1   = 000002          426#
SW10  = 002000          407#
SW11  = 004000          406#
SW12  = 010000          405#
SW13  = 020000          404#
SW14  = 040000          403#
SW15  = 100000          402#
SW2   = 000004          425#
SW3   = 000010          424#
SW4   = 000020          423#
SW5   = 000040          422#
SW6   = 000100          421#
SW7   = 000200          420#
SW8   = 000400          419#
SW9   = 001000          418#
TABLE   012200          914     1987#
TADDR   002006          568#     824*    833     954*    1019*   1024    1073*   1077    1506    1514    1518    1555    1565
                       1569    1644    1666    1681    1706    1742    1765*   1767    1783*   1854    1863    1864*   1866
                       1867*   1871    1932    1933    1936    1938*   1941*   1947    1948    1951    1953*   1956*   1964
                       1965*   1966*   1969*   2078    2099*   2100*   2111    2114    2115    2124*   2127*   3212
TADDR1  002010          569#    1644*   1645*   1668    1683
TBADDR  002012          570#    1936*   1951*   2078*   3207
TBIT  = 000010          475#    1291    1444    1448    1472    1477    1487    1495    1523    1529    1708    1713    1724
                       1732    1785    1791    1822    1826    1836    1843    1876    1882
TBITVE= 000014          460#
TEMP    002046          584#    1391*   1392    1395    1504*   1506    1508*   1509    1517*   1518    1565*   1566*   1567*
                       1741*   1742    1744*   1746    1766*   1767    1852*   1854    1856*   1857    1870*   1871
TEMP1   002050          585#    1022*   1027    1501*   1510*   1511    1569*   1570*   1571    1738*   1747*   1748    1849*
                       1858*   1859    2073    2075    2080*   2082*   2086*   3199
TEMP3   002052          586#    2030*   2034*   2035*   2037    3205
TEMP4   002054          587#    2031*   2038*   2039
```

```
T1OCM   005414   901    940    1186   1202#
TKVEC = 000060   467#
TNUMB   002022   574#   1205*  1231*  1243*  1256*  1290*  1303*  1316*  1336*  1374*  1438*  1466*  1553*
                 1603*  1643*  1705*  1816*  2006   3199   3202   3205   3207   3210   3212
TPVEC = 000064   468#
TRAPVE= 000034   466#   714*   715*
TRTVEC= 000014   461#
TSTBYT  011746   1442   1447   1470   1476   1609   1616   1622   1629   1658   1680   1711   1716   1820
                 1825   1930#
TSTONE  012030   1673   1688   1945#
TST1    005414   1204#  2271
TST10   006232   1335#  2278
TST11   006376   1373#  2279
TST12   006672   1437#  2280
TST13   006774   1465#  2281
TST14   007440   1552#  2282
TST15   007632   1602#  2283
TST16   010024   1642#  2284
TST17   010322   1704#  2285
TST2    005546   1230#  2272
TST20   011152   1815#  2286
TST3    005612   1242#  2273
TST4    005656   1255#  2274
TST5    006052   1289#  2275
TST6    006122   1302#  2276
TST7    006166   1315#  2277
TYPDS = 104405   2796#
TYPE  = 104401   743    766    767    768    769    770    771    772    773    774    778    782    812
                 816    822    861    864    870    873    882    883    886    948    951    973    978
                 1014   1017   1020   1043   1046   1048   1050   1069   1095   1102   1105   1134   1138
                 1140   1142   1144   1146   1148   1150   1152   1154   1157   1194   1197   2096   2117
                 2120   2339   2342   2346   2409   2411   2438   2463   2480   2482   2485   2487   2491
                 2498   2536   2630   2699   2792#  2838
TYPOC = 104402   860    872    950    972    1042   1104   1136   1156   1193   2116   2119   2471   2495
                 2793#
TYPON = 104404   2795#
TYPOS = 104403   2794#
VECTO   002032   578#   1375*  1485*  1722*  1833*
VECTOA  002034   579#   1376*  1486*  1723*  1834*
VECT1   002036   580#
VECT1A  002040   581#
XXDP    002000   565#   756    976
YLOOP   001766   558#   841*   842*   843    846*   847*   848    1053*  1054*  1055   1058*  1059*  1060
                 1084*  1085*  1086   1089*  1090*  1091   1209*  1210*  1211   1214*  1215*  1216   1266*
                 1267*  1268   1276*  1277*  1278   1281*  1282*  1283   1353*  1354*  1355   1358*  1359*
                 1360   1384*  1385*  1386   1404*  1405*  1406   1409*  1410*  1411   1489*  1490*  1491
                 1530*  1531*  1532   1726*  1727*  1728   1793*  1794*  1795   1837*  1838*  1839   1883*
                 1884*  1885   2001*  2002*  2003   2053*  2054*  2055   2058*  2059*  2060   2137*  2138*
                 2139   2142*  2143*  2144   2151*  2152*  2153   2161*  2162*  2163   2171*  2172*  2173
                 2181*  2182*  2183   2193*  2194*  2195   2203*  2204*  2205   2213*  2214*  2215   2223*
                 2224*  2225
$AUTOB  001134   505#
$BDADR  001122   500#
$BDDAT  001126   502#   1218*  1220   1341*  1347*  1395*  1571*  1572   1754*  1755   1761*  1772*  1773
                 1779*  1933*  1934   1948*  1949   2000*  2005*  2008*  2009   2013*  2014*  2075*  2076
                 3199   3202   3207   3210   3214
```

```
$CHARC  015216          2538*  2548*  2555   2564*  2569#
$CKSWR= ****** U        2799
$CMTAG  001100           488#   703    704    712    716    717
$CM1   = 000004          520#   521#   522#   523#   524#
$CM2   = 000010          520#   521#   522#   523#   524#
$CM3   = 000004          518#   520
$CNTLG  014325          2356#
$CNTLU  014320          2355#
$CRLF   001175           526#  2355   2414   2438   2455   2463   2482   2487   2491   2537   2572
$DBLK   015436          2596   2630   2638#
$DOAGN  004310           995   1000   1006#
$DTBL   015426          2599   2634#
$ENDAD  004300           741   1002#  2450
$ENDCT  004264           716    997#
$EOP    004234           988#
$EOPCT  004256           716*   994#   998
$ERFLG  001103           491#  2236   2255   2261*  2270   2428*  2455
$ERMAX  001115           497#   718*  2266*  2270
$ERROR  014514           712   2427#
$ERRPC  001116           498#  2432*  2433*  2434   2455   2469   3199   3202   3205   3207   3210   3212   3214
$ERRTB  002112           616#  2477
$ERRTY  014646          2437   2462#
$ERTTL  001112           495#  2431*  2455
$ESCAP  001172           524#   717*  2265*  2446   2448   2455
$FILLC  001156           516#  2541   2572
$FILLS  001155           515#  2572
$GDADR  001120           499#
$GDDAT  001124           501#  1219*  1232*  1236*  1244*  1248*  1257*  1261*  1270*  1271   1291*  1295*  1304*
                        1308*  1317*  1321*  1340*  1346*  1394*  1440*  1445*  1468*  1474*  1574*  1605*  1612*
                        1618*  1625*  1654*  1672*  1676*  1687*  1709*  1714*  1753*  1755   1760*  1771*  1773
                        1778*  1818*  1823*  1934   1949   1965   1999   2009   2012*  2015*  2076   2098*  2099
                        2111*  2118   2122   2148*  2158*  2168*  2178*  2190*  2200*  2210*  2220*  3199   3202
                        3207   3210   3214
$GET42  004270           999#
$GTSWR= ****** U        2798
$HIOCT  014512          2402*  2413#
$ICNT   001104           492#
$ILLUP  016212          2808   2824   2841#
$INTAG  001135           506#
$ITEMB  001114           496#  2434*  2455   2466
$LF     001176           527#  2346   2355   2414   2455   2572
$LPADR  001106           493#  2259*  2263*  2268   2270
$LPERR  001110           494#   719*  2259   2264*  2270   2445
$MAIL  = ****** U        736   2263   2440   2525
$MNEW   014343          2359#
$MSWR   014332          2357#
$MUT    002002           566#   839*   955*  3199   3202   3207   3212
$NULL   001154           514#  2543   2572
$NWTST= 000000          1203#  1229#  1241#  1254#  1288#  1301#  1314#  1334#  1372#  1436#  1464#  1551#  1601#
                        1641#  1703#  1814#
$OCNT   015670          2671*  2700*  2713#
$OMODE  015672          2666*  2670*  2675   2678*  2689*  2715#
$OVER   014006          2260   2267#
$PASS   001100           489#   991*   992*  1008
$POWER  016220          2839   2844#
$PWRDN  016052          2808#  2836
```

```
$PWRMG  016206            2839#
$PWRUP  016124            2818    2824#
$QUES   001174             525#   2339    2355    2411    2414    2455    2572
$RDCHR  014062            2303#   2799
$RDDEC= ****** U          2802
$RDLIN  014202            2331#   2800
$RDOCT  014354            2375#   2801
$RDSZ = 000010            2324#
$REGAD  001160             518#
$REGO   001162             520#
$REG1   001164             521#
$REG2   001166             522#
$REG3   001170             523#
$RESRE  015732            2749#   2803
$R2A  = ****** U          2804
$SAVRE  015674            2733#   2802
$SAVR6  016216            2817*   2825    2826*   2827*   2843#
$SCOPE  013670             710    2242#
$SETUP= 000027             701#    709     710     712     714     716     717     719     741     744     990    2243    2292
                          2361    2428    2443    2450
$STUP = 177777             701#
$SVLAD  013760            2250    2256    2262#
$SWR  = 121000               1#     524     525     717     719     985     991    1001    1007    1008    1205    1231    1243
                          1256    1290    1303    1316    1336    1374    1438    1466    1553    1603    1643    1705    1816
                          2237    2238    2239    2243    2253    2255    2256    2257    2262    2264    2267    2270    2420
                          2421    2422    2423    2431    2435    2440    2443    2455    2840
$SWRMK= 000000            2239
$SWO8T  014022            2270#
$TKB    001146             511#    2048    2290    2307    2313
$TKS    001144             510#     761*    777*    821*    900*   1016*   1071*   1128*   1185*   2290    2305    2311
$TN   = 000021               1#    1203    1205#   1229    1231#   1241    1243#   1254    1256#   1288    1290#   1301    1303#
                          1314    1316#   1334    1336#   1372    1374#   1436    1438#   1464    1466#   1551    1553#   1601
                          1603#   1641    1643#   1703    1705#   1814    1816#   2271
$TPB    001152             513#    2561*   2572
$TPFLG  001157             517#    2519    2572
$TPS    001150             512#    2559    2572
$TRAP   015770             714     2769#
$TRAP2  016012            2780#   2791
$TRP  = 000013            2784#   2793#   2794#   2795#   2796#   2797#   2799    2800#   2801#   2802#   2803#   2804#
$TRPAD  016024            2774    2791#
$TSTNM  001102             490#    990*   2236    2262*   2267    2270    2430    2455
$TTYIN  014310            2332    2333    2350    2354#
$TYPBN= ****** U          2797
$TYPDS  015222            2584#   2796
$TYPE   015002            2519#   2784    2792
$TYPEC  015152            2540    2547    2554    2559#   2560
$TYPEX  015220            2565    2567    2570#
$TYPOC  015472            2669#   2793
$TYPON  015506            2668    2671#   2795
$TYPOS  015446            2664#   2794
$XTSTR  013670            2244#
$$GET4= 000000            1001#
$$SWO8= 000021            2270#   2271    2272#   2273#   2274#   2275#   2276#   2277#   2278#   2279#   2280#   2281#   2282#
                          2283#   2284#   2285#   2286#   2287#
$OFILL  015671            2665*   2669*   2679    2714#
$40CAT= ****** U          2437
```

```
. MAIN.  MACY11 30(1046)  25-OCT-77  10:10  PAGE 86
DVPCAA.P11    25-OCT-77 10:02          CROSS REFERENCE TABLE -- USER SYMBOLS                                    SEQ 0087

      = 022066            342#    346#    351#    353#    355#    357#    359#    487#    528     529#    530#    707     719
                          746#   1008     2270    2290    2354#   2355    2361    2414    2455    2501#   2572    2638#   2820
                         2842    3198#

  ABS.  022066     000


 ERRORS DETECTED:  0

 DVPCAA.BIN,DVPCAA.SEQ/CRF=DVPCAA.P11
 RUN-TIME:  17  9 .9 SECONDS
 RUN-TIME RATIO:  441./28=15.6
 CORE USED:  24K  (47 PAGES)
```