DL11-C,D,E

OFF-LINE TEST

MD-11-DZDLC-A

EP-DZDLC-A-DL-A

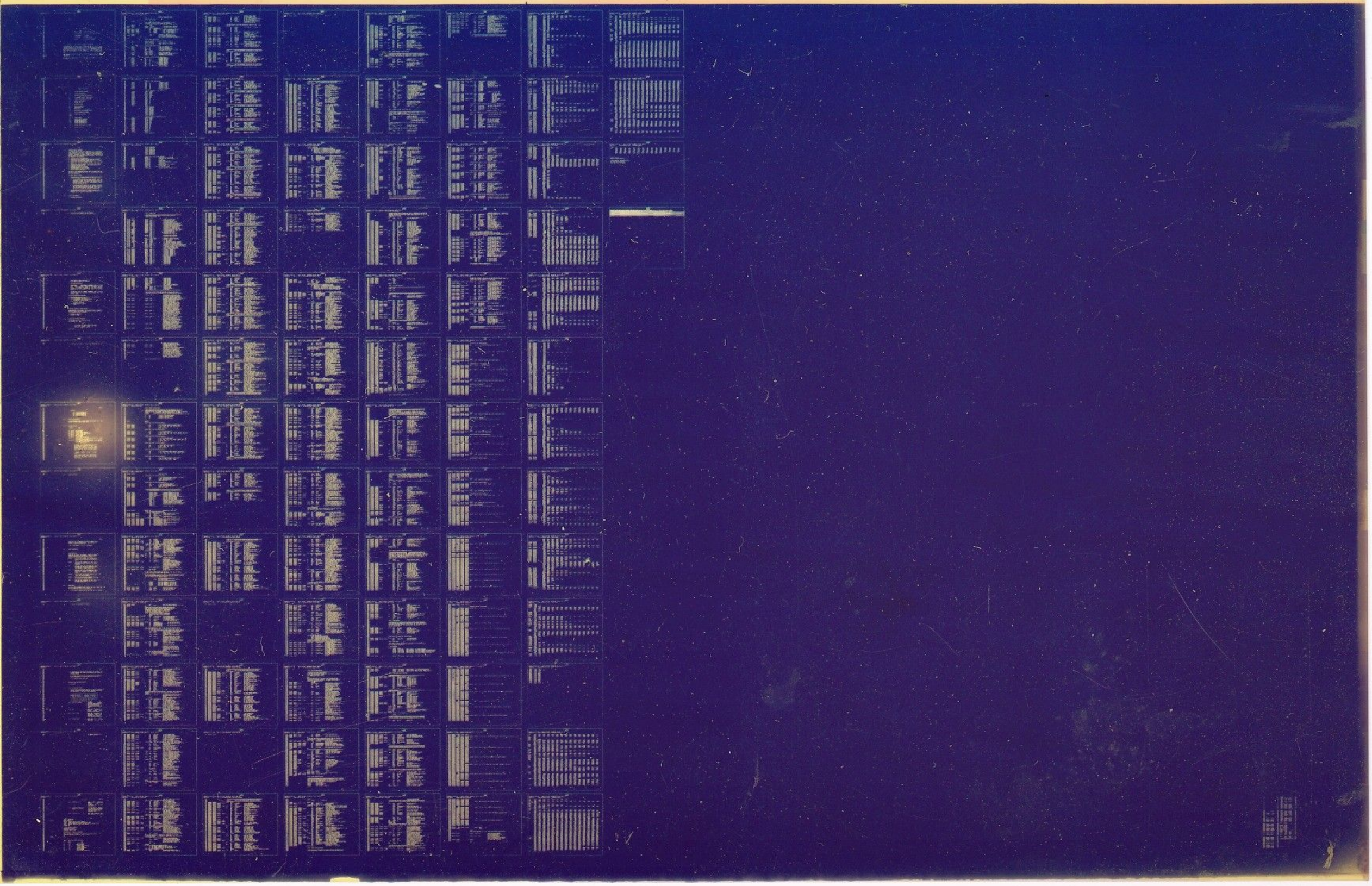
COPYRIGHT 1976

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA



```
.REM    %
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZDLC-A-D
PRODUCT NAME: DL11/C./D. OR /E OFF LINE TEST
DATE CREATED: JULY 1975
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: E. CROWLEY/B. BURGESS

COPYRIGHT (C) 1975, DIGITAL EQUIPMENT CORPORATION

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

TABLE OF CONTENTS

1.0 PROGRAM PURPOSE (ABSTRACT)

THIS PROGRAM HAS THE ABILITY TO TEST THE DL11 (ASYNCHRONOUS MODEM INTERFACE), OFF LINE. MODELS ABLE TO BE TESTED ARE C, D, AND E ONLY. THE USE OF A MODEM IS NOT REQUIRED FOR TESTING; HOWEVER, A SPECIAL CABLE CONNECTOR BCOSC AND A SPECIAL MODEM TEST CONNECTOR H315A IS REQUIRED. THIS PROGRAM IS CAPABLE OF THE FOLLOWING:

A. VERIFICATION OF MAINTENANCE BIT
B. VERIFICATION THAT TRANSMITTER CAN CAUSE AN INTERRUPT
C. VERIFICATION THAT RECEIVER 'DONE' CAN CAUSE AN INTERRUPT
D. CHECKS THAT 'REQ TO SEND' ASSERTS 'RING'
E. CHECKS THAT 'SEC XMIT' ASSERTS 'SEC REC' AND 'DATA SET INT'
F. CHECKS THAT 'DTR' CAN ASSERT 'CLR TO SEND' AND 'CAR DET'
G. VERIFIES THAT 'DATA SET I.E.' CAN CAUSE A RECVR INTR
H. CHECKS THE 'BREAK' FEATURE
I. PERFORMS NULL-DEL-NULL PATTERN
J. PERFORMS BINARY UP COUNT PATTERN
K. PERFORMS BINARY DOWN COUNT PATTERN
L. RUNS A WORSE CASE PATTERN

INCLUDED IN THE PROGRAM ARE SPECIAL USER ROUTINES - PRG #2, PRG #3, PRG #4, AND PRG #5 (WHICH WILL BE DESCRIBED FURTHER INTO THIS DOCUMENT).

NOTE WELL TWO(2) POINTS:

1. THIS PROGRAM IS CAPABLE OF TESTING SIXTEEN(16) DL11'S AND ASSUMES CONTIGUOUS ADDRESSING FROM 1ST DEVICE TO LAST.

   A. IF MULTIPLE DEVICES ARE NOT BEING TESTED, THUS NOT REQUIRING A PASS THRU THE PROGRAM ONCE PER DEVICE, THEN THE PROGRAM WILL DEFAULT TO TESTING THE 1ST POSSIBLE DL11-E DEVICE I.E., RCSR ADDRESS = 775610, AND TEST THIS DEVICE ONLY.

   B. IF MULTIPLE DEVICE TESTING IS NOT BEING CONDUCTED, AND THE DEVICE EXISTING IS NOT THE DEFAULT DL11-E, THEN THE USER ON STARTING THE PROGRAM WILL HAVE TO SET SW<0>=1 TO ENTER THE QUESTION & ANSWER MODE.

2. THIS PROGRAM HAS PROVISION FOR CHARACTER LENGTH I.E., IT ASSUMES DATA IS 8 BITS, BUT ALSO HAS THE ABILITY TO HANDLE 5, 6, OR 7 BITS OF DATA AS WELL.

2.0 SYSTEM REQUIREMENTS

A. HARDWARE REQUIREMENTS

```
                                        PDP-11 FAMILY PROCESSOR WITH 8K OF MEMORY
                                        M7800 DL11 ASYNCHRONOUS LINE INTERFACE MODULE
```

BC05C SPECIAL CABLE CONNECTOR
H315A SPECIAL MODEM TEST CONNECTOR

B. SOFTWARE REQUIREMENTS

THIS PROGRAM WAS SPECIFICALLY DESIGNED FOR THE 11/40 FRONT END OF THE 1080 CONSOLE PROCESSOR SYSTEM. IN THIS ENVIRONMENT IT WOULD BE LOADED BY THE TCDP (DECTAPE) DIAGNOSTIC MONITOR. HOWEVER, ANY 11/40 USER WITH 8K OF MEMORY CAN RUN THIS PROGRAM TO TEST ONE(1) OR MULTIPLE DL11'S.

THE PROGRAM HAS THE PROPER INTERFACE CODE TO ALLOW RUNNING UNDER THE AUTOMATED MANUFACTURING TEST LINE SYSTEM - ACT11.

3.0 RELATED DOCUMENTS AND STANDARDS

A. PROGRAMMING PRACTICES - DOCUMENT NO. 175-003-009-00
B. PDP11/40 PROCESSOR HANDBOOK
C. DL11 ASYNCHRONOUS LINE INTERFACE MANUAL
   DOCUMENT NO. DEC-11-HDLAA
D. PDP-11 MAINDEC SYSMAC' UTILITY PACKAGE
   -
   MAINDEC-11-DZQAC-A1
E. APPLICABLE CIRCUIT SCHEMATIC
   M7800

4.0 DIAGNOSTIC HIERARCHY PREREQUISITES

BEFORE RUNNING THIS PROGRAM, THE FOLLOWING TWO(2) DIAGNOSTIC PROGRAMS SHOULD BE RUN FOR VERIFICATION OF FUNCTIONALITY OF THE 11-INSTRUCTION SET AND MEMORY:

1. MAINDEC-11-DBQEA AND,
2. MAINDEC-11-DZQMC

5.0 LOADING AND STARTING PROCEDURE

LOAD PROGRAM IN MEMORY USING ABS LOADER
LOAD ADDRESS 200.

NOTE

IN THE CASE OF A 1080 SYSTEM ENVIRONMENT LOAD THE PROGRAM USING THE TCDP (DECTAPE) DIAGNOSTIC MONITOR.

PRESS START.

```
212
213                                 A.  THERE ARE ALSO THREE(3) OPTIONAL START ADDRESSES FOR  THE
```

PROGRAM:

204 - SELECTS PROGRAM #2
210 - SELECTS PROGRAM #3
214 - SELECTS PROGRAM #4
220 - SELECTS PROGRAM #5

6.0 SPECIAL ENVIRONMENTS

IF THIS PROGRAM IS RUN IN QUICK VERIFY MODE UNDER ACT11 THE PROGRAM IS DONE AFTER THE FIRST PASS.

7.0 PROGRAM OPTIONS

SWITCH	USE
15=1 OR UP	HALT ON ERROR
14=1 OR UP	LOOP ON TEST
13=1 OR UP	INHIBIT ERROR TYPEOUTS
12=1 OR UP	/C OR /D MODEL BEING TESTED
11=1 OR UP	INHIBIT ITERATIONS
10=1 OR UP	BELL ON ERROR
9=1 OR UP	LOOP ON ERROR
8=1 OR UP	LOOP ON TEST IN SWR<7:0>
<7:0>	HOLDS TEST NO. OF TEST TO BE LOOPED ON. USED IN CONJUNCTION WITH SW<8>.
0=1 OR UP	USED IN DEVICE TABLE CREATION (1 TO 16 DEVICES) I.E., DEFAULT DEVICE NOT DESIRED. ALSO USED FOR CHARACTER LENGTH SETTING. !! NOTE WELL !!

IF SW<08> IS SET THE USER CAN ONLY 'LOOP ON A TEST' OF THE DEFAULT DEVICE I.E. - DL11/E RCSR = 775610. IF THE USER DESIRES TO 'LOOP ON A TEST' OF OTHER THAN THE DEFAULT DEVICE HE MUST FIRST PATCH THE FIVE (5) LOCATIONS LABELED

DLRCSR: DLRDBR: DLXCSR: DLXDBR: DLVECT:

THAT APPEAR UNDER 'DL11 DEFINITIONS' HEADING AT THE FRONT OF THE LISTING. I.E. - WITH SW<08> SET SW<00> IS NOT FUNCTIONAL.

```
270
271                             8.0     EXECUTION TIMES
```

EXECUTION TIME IS DEPENDENT ON TYPE OF MEMORY AND NUMBER OF DL11'S BEING TESTED. A REPRESENTATIVE TIME FOR 1 ERROR FREE PASS IS:

11/40 - CORE MEMORY - 1 DL11/E - 20 SECONDS

9.0 ERROR INFORMATION

9.1 ERROR REPORTING

THERE ARE A TOTAL OF SEVEN(7) TYPES OF ERROR REPORTS GENERATED BY THE PROGRAM. THE KEY COLUMN HEADINGS WILL BE DESCRIBED BELOW FOR CLARITY -

DEVADR - THIS IS THE ADDRESS OF THE RECEIVER CONTROL STATUS REGISTER FOR THE FAILING DL11

REGADR - THIS IS THE ADDRESS OF THE DL11 REGISTER ON WHICH TESTING IS BEING CONDUCTED

WAS - THIS IS WHAT THE CONTENTS OF THE REGISTER OF THE DL11 UNDERGOING TEST WAS (ADDRESS IS UNDER COLUMN '(R2)')

S/B - THIS IS WHAT THE CONTENTS OF THE REGISTER OF THE DL11 UNDERGOING TEST SHOULD BE (ADDRESS IS UNDER COLUMN '(R2)')

WASADR - THIS IS WHAT THE MEMORY ADDRESS WAS (INPUT DATA BUFFER ADDRESS)

SHBADR - THIS IS WHAT THE MEMORY ADDRESS SHOULD BE (OUTPUT DATA BUFFER ADDRESS)

(REG) - THIS IS THE CONTENTS OF THE DL11 RECEIVER DATA BUFFER IN ERROR (ADDRESS IS UNDER COLUMN '(R2)')

9.2 ERROR HALTS

WITH THE 'HALT ON ERROR' SWITCH (SW15) NOT SET THERE ARE FOUR(4) PROGRAMMED 'HALTS' IN THE PROGRAM:

A. IN THE CASE OF ERROR REPORTING AND THERE IS NO TERMINAL

328 TO ALLOW THE INFORMATION TRANSFER.

B. IN THE POWER FAIL ROUTINE IF THE POWER UP SEQUENCE WAS STARTED BEFORE THE POWER DOWN SEQUENCE HAD A CHANCE TO COMPLETE ITSLEF.

C. IN THE END OF PASS ROUTINE IF MULTIPLE DEVICE TESTING IS BEING CONDUCTED BUT NO DEVICES ARE SHOWN AS ACTIVE.

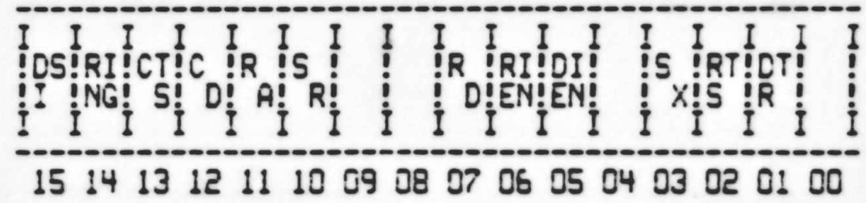
D. IN THE CASE OF SW<08> BEING SET.

10.0 PERFORMANCE AND PROGRESS REPORTS

NOT APPLICABLE.

11.0 DEVICE INFORMATION TABLES

A. THE FOLLOWING IS A PICTURE VIEW OF A DL11-E RECEIVER CONTROL STATUS REGISTER, WHICH WILL SHOW BIT ASSIGNMENTS AND DEFINITIONS, TO PROVIDE A HANDY REFERENCE:



BIT ASSIGNMENTS ARE DEFINED AS FOLLOWS:

BIT15 DATA SET INTERRUPT
1. INTERRUPT SEQUENCE INITIATED WHEN BIT05 SET.
2. SETS WHENEVER BITS 10, 11, 12 OR 14 CHANGE STATE
3. CLEARED BY INIT OR READING RCSR

BIT14 RING
1. WHEN SET, INDICATES A CONTROL SIGNAL BEING RECEIVED FROM DATASET.

BIT13 CLEAR TO SEND
1. WHEN SET INDICATES ON CONDITION; WHEN CLEAR INDICATES OFF CONDITION.
2. DEPENDENT ON STATE OF 'CTS' SIGNAL FROM DATASET

BIT12 CARRIER DETECT
1. SETS WHEN DATA CARRIER RECEIVED
2. WHEN CLEAR INDICATES END

```
                                                                OF CURRENT TRANSMISSION OR
                                                                AN ERROR CONDITION.
```

```
                                BIT11  RECEIVER ACTIVE          1. WHEN SET INDICATES
                                                                   RECEIVER INTERFACE IS
                                                                   ACTIVE.
                                                                2. CLEARED BY INIT OR RCVR
                                                                   DONE (BIT07).

                                BIT10  SECONDARY RECEIVE OR     1. PROVIDES RECEIVE CAPABIL-
                                       SUPERVISORY RECEIVED        ITY, WHEN SET, FOR REVERSE
                                       DATA                        CHANNEL OF REMOTE STATION.
                                                                   SETS WHEN BIT03 IS SET.
                                                                2. CLEARED BY INIT

                                BIT07  RECEIVER DONE            1. SETS WHEN CHARACTER HAS
                                                                   BEEN RECEIVED.  WILL INITI

                                                                %

                        .NLIST  CND,MD,MC
                        .LIST   ME,SEQ,BIN
167400                  $SWR=167400
                        .MCALL  .HEADER,.$CMTAG,.SETUP,.$ERROR
                        .MCALL  .SWRHI,.SWRLO,.$CATCH,.EQUATE,SETUP..$EOP
                        .MCALL  .$SCOPE,.$ERRTYP,.$TYPOCT,.$TYPDEC,.$TYPE,.$POWER,.$TRAP
                        .MCALL  .$RDOCT,.$READ,PUSH,POP,STARS,.$RDDEC
                        .ENABLE ABS

                        .TITLE  MAINDEC-11-DZDLC-A
                        ;*COPYRIGHT (C) 1975
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY E. CROWLEY/B. BURGESS
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                        ;*
000001                  $TN=1


                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH              USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        12            /C OR /D MODEL
                        ;*        11            INHIBIT ITERATIONS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            LOOP ON TEST IN SWR<7:0>
```

```
                                        :*          D           CREATION OF DEVICE'S TABLE
                                        :*                      OR CHANGE CHARACTER LENGTH

                                        .SBTTL  TRAP CATCHER

                    000000                      .=0
                                        :*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        :*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        :*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                    000174                      .=174
000174  000000    DISPREG: .WORD  0               ::SOFTWARE DISPLAY REGISTER
000176  000000    SWREG:   .WORD  0               ::SOFTWARE SWITCH REGISTER
                                        .SBTTL  STARTING ADDRESS(ES)
000200  000137  001446                  JMP     @#BEGIN ::JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  006300                  JMP     @#PRG2          :JUMP TO USER PROGRAM NO. 2
000210  000137  006570                  JMP     @#PRG3          :JUMP TO USER PROGRAM NO. 3
000214  000137  007040                  JMP     @#PRG4          :JUMP TO USER PROGRAM NO. 4
000220  000137  007432                  JMP     @#PRG5          :JUMP TO USER PROGRAM NO. 5

                    000052                      .=52                ;INFORMATION LOCATION FOR ACT11
000052  000000                          .WORD   0               ;NO POWER FAIL REQUIRED (BIT15=0)
                                                                ;IS NOT MEMORY SIZE DEPENDENT (BIT14=0)
                                                                ;IS SUITABLE FOR AUTOMATIC OPERATION (BIT13=0)

                                        .SBTTL  BASIC DEFINITIONS

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

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

                                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
                    000000  R0=     %0              ::GENERAL REGISTER
                    000001  R1=     %1              ::GENERAL REGISTER
                    000002  R2=     %2              ::GENERAL REGISTER
                    000003  R3=     %3              ::GENERAL REGISTER
                    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
000040                          PR1=    40              ;;PRIORITY LEVEL 1
000100                          PR2=    100             ;;PRIORITY LEVEL 2
000140                          PR3=    140             ;;PRIORITY LEVEL 3
000200                          PR4=    200             ;;PRIORITY LEVEL 4
000240                          PR5=    240             ;;PRIORITY LEVEL 5
000300                          PR6=    300             ;;PRIORITY LEVEL 6
000340                          PR7=    340             ;;PRIORITY LEVEL 7

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

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

```
                                .EQUIV  BIT08,BIT8
                                .EQUIV  BIT07,BIT7
                                .EQUIV  BIT06,BIT6
                                .EQUIV  BIT05,BIT5
                                .EQUIV  BIT04,BIT4
                                .EQUIV  BIT03,BIT3
                                .EQUIV  BIT02,BIT2
                                .EQUIV  BIT01,BIT1
                                .EQUIV  BIT00,BIT0

                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                          TBITVEC=14              ;;"T" BIT
000014                          TRTVEC= 14              ;;TRACE TRAP
000014                          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                          PWRVEC= 24              ;;POWER FAIL
000030                          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                          TRAPVEC=34              ;;"TRAP" TRAP
000060                          TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                          TPVEC=  64              ;;TTY PRINTER VECTOR
000240                          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
```

```
                                .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                  $REG4:  .WORD   0       ;;CONTAINS (($REGAD)+10)
001174  000000                  $REG5:  .WORD   0       ;;CONTAINS (($REGAD)+12)
001176  000000                  $REG6:  .WORD   0       ;;CONTAINS (($REGAD)+14)
001200  000000                  $REG7:  .WORD   0       ;;CONTAINS (($REGAD)+16)
001202  000000                  $TMP0:  .WORD   0       ;;USER DEFINED
001204  000000                  $TMP1:  .WORD   0       ;;USER DEFINED
001206  000000                  $TMP2:  .WORD   0       ;;USER DEFINED
001210  000000                  $TMP3:  .WORD   0       ;;USER DEFINED
001212  000000                  $TMP4:  .WORD   0       ;;USER DEFINED
001214  000000                  $TMP5:  .WORD   0       ;;USER DEFINED
001216  000000                  $TMP6:  .WORD   0       ;;USER DEFINED
001220  000000                  $TMP7:  .WORD   0       ;;USER DEFINED
001222  000000                  $TMP10: .WORD   0       ;;USER DEFINED
```

```
                                $TMP11: .WORD   0               ;;USER DEFINED
                                $TMP12: .WORD   0               ;;USER DEFINED
                                $TMP13: .WORD   0               ;;USER DEFINED
                                $TMP14: .WORD   0               ;;USER DEFINED
                                $TMP15: .WORD   0               ;;USER DEFINED
                                $TMP16: .WORD   0               ;;USER DEFINED
                                $TMP17: .WORD   0               ;;USER DEFINED
                                $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
                                $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
                                $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
                                $QUES:  .ASCII  /?/             ;;QUESTION MARK
                                $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
                                $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**************************************************

                                ;THE FOLLOWING TAG(S) ARE USER SUPPLIED BY CALLING THE MACRO
                                ;'MORETAGS' AS ONE OF THE ARGUMENTS TO THE SYSMAC ROUTINE .$CMTAG

                                TABFLG: .WORD   0               ;AN INDICATOR TO SHOW THAT THE
                                                                ;INFORMATION FOR MULTIPLE DEVICE
                                                                ;TESTING HAS ALREADY TRANSPIRED
                                                                ;& 'MAINDEC' NAME HAS BEEN PRINTED
                                DLBASE: .WORD   0               ;STORAGE & WORKING LOCATION FOR A DEVICE
                                                                ;RECEIVER STATUS REGISTER ADDRESS
                                KEEPAD: .WORD   0               ;STORAGE LOCATION FOR THE 1ST
                                                                ;DEVICE RCSR FROM WHICH
                                                                ;'BASEADD' IS RESTORED AT THE
                                                                ;END OF A COMPLETE PROGRAM PASS.
                                BASEADD:.WORD   0               ;STORAGE LOCATION WHICH HOLDS
                                                                ;THE RCSR ADDRESS OF THE 'NEXT'
                                                                ;DEVICE DURING MULTIPLE TESTING
                                KEEPIV: .WORD   0               ;STORAGE LOCATION FOR THE 1ST
                                                                ;DEVICE RECEIVER VECTOR FROM
                                                                ;WHICH 'BASEIV' IS RESTORED AT THE
                                                                ;END OF A COMPLETE PROGRAM PASS
                                BASEIV: .WORD   0               ;STORAGE LOCATION WHICH HOLDS
                                                                ;THE VECTOR ADDRESS OF THE 'NEXT'
                                                                ;DEVICE DURING MULTIPLE TESTING
                                MULTD:  .WORD   0               ;FLAG TO INDICATE TO 'END OF PASS'
                                                                ;ROUTINE THAT MULTIPLE DEVICE
                                                                ;TESTING IS BEING CONDUCTED
                                                                ;0=NO, 1=YES
                                ACTREG: .WORD   0               ;THIS IS THE DEVICE ACTIVE REGISTER
                                                                ;A BIT IS SET (STARTING AT
                                                                ;BIT0)FOR EACH CONTIGUOUS DEVICE
                                                                ;(A MAX. OF 16) THAT IS TO UNDERGO
                                                                ;TESTING.  THIS LOCATION IS
                                                                ;AUTOMATICALLY FILLED BASED ON
                                                                ;USER RESPONSE TO PROGRAM QUESTIONS
                                ROTADD: .WORD   0               ;A ROTATING POINTER TO SIGNAL
                                                                ;THE LAST DEVICE TESTED (IF
                                                                ;MULTIPLE DEVICE TESTING WAS BEING
                                                                ;DONE) IF LESS THAN A FULL COMPLE-
                                                                ;MENT OF DEVICES (16)WAS SELECTED
                                LASTADD:.WORD   0               ;STORAGE LOCATION FOR THE
                                                                ;RCSR ADDRESS OF THE LAST DEVICE
```

```
                                                        ;TESTED (IF MULTIPLE DEVICE
                                                        ;TESTING WAS SELECTED BY USER)
001302  000000          DLPRI:  .WORD   0             ;STORAGE LOCATION FOR THE DEVICE
                                                        ;INTERRUPT PRIORITY LEVEL
001304  000000          LESS1:  .WORD   0             ;THE PRIORITY LEVEL THE CPU
                                                        ;MUST BE AT TO ALLOW DEVICE INTERRUPTS.
                                                        ;THIS WILL BE 1 LEVEL LESS THAN
                                                        ;THE DEVICE LEVEL (BASED ON &
                                                        ;CALCULATED FROM USER RESPONSE TO
                                                        ;DEVICE PRIORITY LEVEL QUESTION)
001306  177740          STLMSK: 177740                ;THIS MASK IS USED BY THE 'STALL'
                                                        ;ROUTINE WHICH WAITS A RANDOM NO.
                                                        ;OF MILLISECONDS.  ITS' USE PREVENTS
                                                        ;A STALL > 37 MSEC.  THIS LOCATION
                                                        ;HOWEVER, CAN BE PATCHED BY THE
                                                        ;USER TO ALLOW LARGER 'STALLS'.

                        ;END OF USER SUPPLIED TAG(S)
```

```
                                .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


001310                          $ERRTB:

                                ;ERROR TABLE ITEM FOR ERROR MESSAGE 1

001310  015132                          EM1             ;"DL11 REGISTER REFERENCE CAUSED TIMEOUT"
001312  015201                          DH1             ;"   (PC)  (PS)  (SP)  TEST  DEVADR  REGADR  "
001314  015260                          DT1             ; (R7)  (PSW)  (R5)  (R0)  (R1)  (R2)
001316  000000                          0               ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR MESSAGE 2

001320  015276                          EM2             ;" DL11 REGISTER ERROR "
001322  015322                          DH2             ;"   (PC)  (PS)  (SP)  TEST  DEVADR  REGADR  WAS  S/B  "
001324  015420                          DT2             ;" (R7)  (PSW)  (R6)  (R0)  (R1)  (R2)  (R3)  (R4)  "
001326  000000                          0               ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR MESSAGE 3

001330  015442                          EM3             ;" DL11 DATA COMPARE ERROR "
001332  015472                          DH3             ;" (PC)  (PS)  (SP)  TEST  WASADR  SHBADR  WAS  S/B  "
001334  015570                          DT3             ;"  (R7)  (PSW)  (R6)  (R0)  (R1)  (R2)  (R3)  (R4)  "
001336  000000                          0               ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR MESSAGE 4

001340  015612                          EM4             ;" UNEXPECTED TRAP TO VECTOR AT LOCATION XXX "
001342  015664                          DH4             ;"  (PC)  (PS)  (SP)  TEST  "
001344  015722                          DT4             ;" (R7)  (PSW)  (R6)  (R0)  "
001346  000000                          0               ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR MESSAGE 5

001350  015734                          EM5             ;" DL11 SOFT ERROR (PARITY,FRAMING, OR OVERRUN)
001352  016011                          DH5             ;" (PC)  (PS)  (SP)  TEST  DEVADR  REGADR  (REG)  "
001354  016100                          DT5             ;"  (R7)  (PSW)  (R6)  (R0)  (R1)  (R2)  (R3)  "
001356  000000                          0

                                ;ERROR TABLE ITEM FOR ERROR MESSAGE 6

001360  015132                          EM1             ;"DL11 REGISTER REFERENCE CAUSED TIMEOUT"
001362  016120                          DH6             ;"   (PC)     (PS)     (SP)     REGADR"
001364  016160                          DT6             ;$ERRPC,$TMP0,$REG6,$REG2
```

```
                                        0               ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR MESSAGE 7

                                        EM5             ;" DL11 SOFT ERROR (PARITY,FRAMING, OR OVERRUN) "
                                        DH7             ;" (PC)  DEVADR  REGADR  (REG)"
                                        DT7             ;$ERRPC,$REG1,$REG2,$REG3
                                        0               ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR MESSAGE 10

                                        EM3             ;" DL11 DATA COMPARE ERROR "
                                        DH10            ;" (PC)  DEVADR  REGADR  (REG)  S/B"
                                        DT10            ;$ERRPC,$REG1,$REG2,$REG3,$REG4
                                        0               ;PRINT ALL OCTAL

                                ;;*****************************************************************
                                        ;DL11 DEFINITIONS
                                ;;*****************************************************************

                                        DLRCSR: 175610          ;CONTAINS ADDRESS OF RCVR CSR
                                        DLRDBR: 175612          ;CONTAINS ADDRESS OF RCVR DBR
                                        DLXCSR: 175614          ;CONTAINS ADDRESS OF XMIT CSR
                                        DLXDBR: 175616          ;CONTAINS ADDRESS OF XMIT DBR
                                        DLVECT: 300             ;CONTAINS VECTOR ADDRESS OF CURRENT DL11
                                        XFLG0:  0               ;FLAG FOR HARD XMIT ERRORS
                                        RFLG0:  0               ;FLAG FOR HARD RCVR ERRORS
                                        RFLG1:  0               ;FLAG FOR SOFT RCVR ERRORS
                                        RTRY:   0               ;COUNTS NO. OF RETRIES ON SOFT ERRORS
                                        OPTR:   0               ;CONTAINS POINTER TO OUTPUT BUFFER
                                        IPTR:   0               ;CONTAINS POINTER TO INPUT BUFFER
                                        LDOUT:  0               ;CONTAINS POINTER TO LOAD BUFFER ROUTINE
                                        TIMR1:  0               ;TIMERS FOR 256. BYTE BLOCK TRANSFERS
                                        TIMR2:  0
                                        INTFLG: 0               ;SOFTWARE INTR. FLAG


001446  000240          BEGIN:  NOP                     ;PROGRAM WILL START HERE
                        .SBTTL  INITIALIZE THE COMMON TAGS
                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001450  012706  001100          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001454  005026                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001456  022706  001140          CMP     #SWR,R6 ;;DONE?
001462  001374                  BNE     .-6             ;;LOOP BACK IF NO
001464  012706  001100          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                        ;;INITIALIZE A FEW VECTORS
001470  012737  010462  000020  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001476  012737  000340  000022  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001504  012737  010732  000030  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001512  012737  000340  000032  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001520  012737  013052  000034  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001526  012737  000340  000036  MOV     #340,@#TRAPVEC+2;LEVEL 7
001534  012737  013132  000024  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001542  012737  000340  000026  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
```

```
                                                CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
                                                CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
                                                MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
                                                MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
                                                MOV     #.,$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.
                                                MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
                                                MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
                                                MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
                                                MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
                                                CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
                                                BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
                                                BR      65$             ;;BRANCH IF NO TIMEOUT
                                        64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
                                                RTI
                                        65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
                                                MOV     #DISPREG,DISPLAY
                                        66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

                                                CLR     MULTD           ;CLEAR MULTIPLE DEVICE
                                                                        ;TESTING FLAG
                                                CLR     TABFLG          ;CLEAR TABLE CREATION FLAG
                                                MOV     #8.,$TMP15      ;SET CHARACTER LENGTH DESIGNATOR
                                                                        ;FOR 8 BITS --- THIS IS THE DEFAULT
                                                                        ;LENGTH ASSUMED BY THE PROGRAM
                                                                        ;UNLESS THE USER CHANGES IT THRU
                                                                        ;THE QUESTION AND ANSWER CYCLE
                                                                        ;INITIATED BY SETTING SW<0> TO A 1
                                                MOV     #200,DLPRI      ;SET STANDARD PRIORITY LEVEL
                                                                        ;FOR DEVICE
                                                BIT     #SW8,SWR        ;IS THE 'LOOP ON TEST' SWITCH SET?
                                                BEQ     1$              ;BRANCH IF NOT

;IF THE 'LOOP ON TEST' SWITCH WAS SET WE WILL TAKE THE NEXT BRANCH
;INSTRUCTION THUS BYPASSING TABLE CREATION
;
;IF THE USER DESIRED TO LOOP ON A TEST OF OTHER THAN THE DEFAULT DEVICE
;THEN HE SHOULD HAVE PREVIOUSLY FILLED THE FOLLOWING PROGRAM LOCATIONS
;WITH THE DESIRED DEVICE REGISTER VALUES:
;
;                               ******************************
;               UNDER           ;DL11 DEFINITIONS               ABOVE
;                               ******************************
;
;       DLRCSR:                 PATCH THE ADDRESS OF THE RCVR CSR
;       DLRDBR:                 PATCH THE ADDRESS OF THE RCVR DBR
;       DLXCSR:                 PATCH THE ADDRESS OF THE XMIT CSR
;       DLXDBR:                 PATCH THE ADDRESS OF THE XMIT DBR
;       DLVECT:                 PATCH THE VECTOR ADDRESS OF THIS DL11
;

                                                TYPE,   STMES           ;PRINT OUT 'MAINDEC' NAME
                                                TYPE,   FAILSA          ;TYPE FAILSAFE MESSAGE
                                                ERROR   +0              ;TYPE OUT THE PC VALUE
                                                TYPE,   PCMSG           ;FOLLOWED BY =PC
```

```
                                        HALT                    ;WAIT FOR USER TO RESPOND
                                        BR      ONCE            ;GO TO TEST DEVICE PATCHED IN BY USER
                                1$:
                                ;ENSURE THAT IF MULTIPLE DEVICE TESTING WAS BEING DONE
                                ;AND THE USER 'HALTED' THE PROGRAM BEFORE ALL DEVICES
                                ;WERE COMPLETED AND WENT BACK TO 'LOAD ADDRESS 200'
                                ;TO RESTART THE PROGRAM THAT AS A BARE MINIMUM
                                ;HE CAN RUN THE DEFAULT DEVICE (1ST RECEIVER
                                ;STATUS REGISTER ADDRESS 175610)
                                ;NOTE:  IF THIS IS NOT SUITABLE THE USER WILL
                                ;       HAVE TO SET SW0=1 (OR UP) IN ORDER TO
                                ;       RECREATE THE TABLE HE DESTROYED FROM
                                ;       ABOVE
001746  012767  175610  177304          MOV     #175610,DLBASE  ;1ST POSSIBLE RECEIVER CSR
001754  004767  006074                  JSR     PC,DLADDR       ;FORM DL ADDRESSES FOR
                                                                ;1ST POSSIBLE DEVICE
001760  012767  000300  177432          MOV     #300,DLVECT     ;1ST POSSIBLE INTERRUPT VECTOR
001766  005067  177264                  CLR     TABFLG          ;CLEAR TABLE CREATION FLAG

001772  012706  001100          RESTRT: MOV     #STACK,SP               ;SET UP STACK POINTER
001776  012737  015020  000004          MOV     #BUSERR,@#ERRVEC        ;SET UP BUS ERROR VECTOR
002004  012737  000340  000006          MOV     #340,@#ERRVEC+2
002012  012737  015044  000010          MOV     #RSVERR,@#RESVEC        ;SET UP RSVD INSTR. VECTOR
002020  012737  000340  000012          MOV     #340,@#RESVEC+2

                                ;THIS  NEXT SECTION WILL CHECK TO SEE IF MULTIPLE DEVICE TESTING
                                ;WILL TAKE PLACE I.E.-
                                ;       A)  HAS FREE RUNNING DEVICE TABLE ALREADY BEEN CREATED, AND/OR
                                ;       B)  IF IT HAS, DOES USER WISH TO CHANGE IT, OR DO WE TEST DEFAULT DEVICE?
002026  105767  177224                  TSTB    TABFLG          ;HAS TABLE CREATION BEEN PERFORMED?
002032  001010                          BNE     ONCE            ;BRANCH IF YES TO SKIP 'MAINDEC
                                                                ;TITLE' MESSAGE
002034  104401  016635                  TYPE    ,STMES          ;OTHERWISE, PRINT OUT 'MAINDEC'
                                                                ;NAME
002040  105167  177212                  COMB    TABFLG          ;IF TABLE CREATION HAS NOT BEEN
                                                                ;PERFORMED, THEN SET FLAG, AND DO SO
002044  032767  000001  177066          BIT     #SW0,SWR        ;THE PROGRAM HAS OBVIOUSLY BEEN
                                                                ;RESTARTED - DOES USER WISH TO
                                                                ;RESELECT VECTOR AND CONTROL REGISTER
                                                                ;ADDRESSES I.E. - CREATE A NEW TABLE?
002052  001012                          BNE     GO              ;BRANCH IF YES
002054  005077  177334          ONCE:   CLR     @DLXCSR         ;CLEAR OUT BOTH CSR'S
002060  005077  177324                  CLR     @DLRCSR
002064  005777  177322                  TST     @DLRDBR         ;FLUSH RCVR "DONE" BIT
002070  005777  177316                  TST     @DLRDBR
002074  000167  000670                  JMP     TST1            ;OTHERWISE, GO WITH EXISTING
                                                                ;TABLE OR NOT USE ANY TABLE AT
                                                                ;ALL WHICHEVER THE CASE MAY BE
                                                                ;DEFAULT CASE IS 1ST POSSIBLE
                                                                ;DEVICE)
                                ;IF WE COME THIS PATH THE USER HAS DECIDED 1 OF 2 ALTERNATIVES:
                                ;       A)  TO RUN MULTIPLE DEVICES
                                ;       B)  TO CREATE A NEW TABLE TO RUN FROM, OR
                                ;       C)  TO CHANGE THE CHARACTER LENGTH
002100  104401  017217          GO:     TYPE,   LENGTH          ;ASK USER FOR THE CHARACTER LENGTH
                                                                ;FOR WHICH HIS DEVICE IS SET
```

```
                                        RDDEC                   ;ACCEPT THE ANSWER TYPED BY USER
                                ;CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
                                        MOV     (SP)+,R0        ;GET THE ANSWER TYPED
                                        CMP     R0,#8.          ;IS THE NUMBER TOO HIGH?
                                        BHI     RETRY           ;IF YES - GO TO RETRY SITUATION
                                        CMP     R0,#5.          ;IS THE NUMBER TOO LOW?
                                        BLO     RETRY           ;IF YES - GO TO RETRY SITUATION
                                        MOV     R0,$TMP15       ;THE VALUE TYPED IS OK
                                                                ;STORE FOR FUTURE USE
                                        TYPE,   DEFAULT         ;ASK USER IF HE WISHES TO TEST OTHER
                                                                ;THAN THE DEFAULT DEVICE
                                        RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
                                        TST     (SP)+           ;LOOK AT THE ANSWER
                                        BNE     1$              ;BRANCH IF REPLY WAS YES
                                        JMP     @#FLUSH         ;OTHERWISE, SKIP REST OF INTERROGATION
                                1$:     MOV     #300,R0         ;START RESTORATION OF TRAPCATCHER
                                        MOV     #302,R1         ;AREA FROM LOCATIONS 300 TO 776
                                        MOV     #4,R2           ;SO THAT WE CREATE THE MULTIPLE
                                2$:     MOV     R1,(R0)         ;DEVICES TABLE WITH A CLEAN SLATE
                                        CLR     (R1)            ;       .
                                        ADD     R2,R0           ;       .
                                        ADD     R2,R1           ;       .
                                        CMP     #1000,R1        ;       .
                                        BLT     2$              ;       .
                                ;THE TRAPCATCHER VECTOR AREA FROM 300 - 776 SHOULD NOW BE RESTORED.
                                ;PROCEED TO FIND OUT THE 1ST DEVICE RECEIVER CONTROL REGISTER
                                ;ADDRESS
                                FIRSTD: TYPE    ,MFIRSTD        ;ASK USER FOR THE RECEIVER CONTROL
                                                                ;REGISTER ADDRESS OF HIS FIRST
                                                                ;DEVICE
                                        RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
                                                                ;AND STORE ON TOP OF STACK
                                ;CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
                                        MOV     (SP)+,R0        ;GET THE ANSWER TYPED
                                        CMP     R0,#176170      ;IS THE NUMBER TOO HIGH?
                                        BHI     RETRY0          ;IF YES-GO TO RETRY SITUATION
                                        CMP     R0,#175610      ;IS THE NUMBER TOO LOW?
                                        BLO     RETRY0          ;IF YES - GO TO RETRY SITUATION
                                        BITB    #BIT0,R0        ;NUMBER IS IN RANGE BUT IS IT
                                                                ;ON AN EVEN BOUNDARY?
                                        BNE     RETRY0          ;IF NO - GO TO RETRY SITUATION
                                ;CHECK TO SEE IF USER RESPONSE WAS TRULY A RCVR STATUS REGISTER
                                        BIT #7,R0               ;WAS THE LEAST SIGNIFICANT DIGIT OF THE
                                                                ;USER RESPONSE EQUAL TO A ZERO?
                                        BNE     RETRY0          ;BRANCH IF NOT
                                        MOV     R0,DLBASE       ;THE 1ST ADDRESS VALUE TYPED IS OK
                                                                ;STORE FOR FUTURE USE
                                ;NOW WE ARE READY TO FIND OUT THE DEVICE INTERRUPT VECTOR
                                        MOV     DLBASE,KEEPADD  ;GET 1ST ADDRESS VALUE
                                        JSR     PC,DLADDR       ;GO FORM DL ADDRESSES FOR
                                                                ;1ST DEVICE SELECTED
                                        MOV     KEEPADD,BASEADD ;RESTORE 1ST DEVICE ADDRESS
                                VECT:   TYPE    ,MVECT          ;ASK USER FOR A VECTOR ADDRESS
                                        RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
                                                                ;AND STORE ON TOP OF STACK
                                ;CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
```

```
  989  002272  012600                        MOV     (SP)+,R0        ;GET THE ANSWER TYPED
  990  002274  020027  000776                CMP     R0,#776         ;IS THE NUMBER TOO HIGH?
  991  002300  101032                        BHI     RETRY1          ;IF YES - GO TO RETRY SITUATION
  992  002302  020027  000300                CMP     R0,#300         ;IS THE NUMBER TOO LOW?
  993  002306  103427                        BLO     RETRY1          ;IF YES - GO TO RETRY SITUATION
  994  002310  132700  000001                BITB    #BIT0,R0        ;NUMBER IS IN RANGE BUT IS IT
  995                                                                 ;ON AN EVEN BOUNDARY?
  996  002314  001024                        BNE     RETRY1          ;IF NO - GO TO RETRY SITUATION
  997                              ;CHECK TO SEE IF THE USER RESPONSE WAS TRULY A RCVR VECTOR ADDRESS
  998  002316  032700  000007                BIT     #7,R0           ;WAS THE LEAST SIGNIFICANT DIGIT OF THE
  999                                                                 ;USER RESPONSE EQUAL TO A ZERO?
 1000  002322  001021                        BNE     RETRY1          ;BRANCH IF NOT
 1001  002324  010067  177070                MOV     R0,DLVECT       ;THE VECTOR VALUE TYPED IS OK
 1002                                                                 ;STORE FOR FUTURE USE
 1003  002330  016767  177064  176730        MOV     DLVECT,KEEPIV   ;GET THE FIRST VECTOR VALUE
 1004  002336  016767  177056  176724        MOV     DLVECT,BASEIV   ;SAVE FIRST VECTOR VALUE
 1005  002344  000414                        BR      HOWMANY         ;GO TO SEE IF USER WANTS MORE
 1006                                                                 ;THAN 1 DEVICE
 1007  002346  104401  001252        RETRY:  TYPE,   $QUES           ;TYPE '?' INDICATING USER TYPED
 1008                                                                 ;SOMETHING WRONG FOR CHARACTER LENGTH
 1009  002352  000167  177522                JMP     GO              ;GO BACK TO REISSUE QUESTION
 1010  002356  104401  001252        RETRY0: TYPE    ,$QUES          ;TYPE '?' INDICATING USER TYPE
 1011                                                                 ;SOMETHING WRONG FOR 1ST ADDRESS
 1012  002362  000167  177612                JMP     FIRSTD          ;GO BACK TO REISSUE QUESTION
 1013  002366  104401  001252        RETRY1: TYPE    ,$QUES          ;TYPE '?' INDICATING USER TYPED
 1014                                                                 ;SOMETHING WRONG FOR VECTOR
 1015  002372  000167  177666                JMP     VECT            ;GO BACK TO REISSUE QUESTION
 1016  002376  104401  017552        HOWMANY:TYPE    ,MULDEV         ;ASK USER IF HE WISHES TO RUN
 1017                                                                 ;MULTIPLE DEVICES
 1018  002402  104410                        RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
 1019                                                                 ;AND STORE ON TOP OF STACK
 1020  002404  012600                        MOV     (SP)+,R0        ;GET THE ANSWER TYPED
 1021  002406  005700                        TST     R0              ;WAS THE ANSWER YES?
 1022  002410  001003                        BNE     1$              ;BRANCH IF IT WAS
 1023  002412  005067  176654                CLR     MULTD           ;OTHERWISE, INITIALIZE FLAG TO
 1024                                                                 ;INDICATE NON-MULTIPLE DEVICES
 1025  002416  000402                        BR      2$              ;SKIP NEXT INSTRUCTION
 1026  002420  105167  176646        1$:     COMB    MULTD           ;INITIALIZE FLAG TO INDICATE
 1027                                                                 ;RUNNING OF MULTIPLE DEVICES
 1028  002424                          2$:
 1029  002424  105767  176642                TSTB    MULTD           ;ARE THERE MULTIPLE DEVICES ON
 1030                                                                 ;THE SYSTEM?
 1031  002430  100406                        BMI     LASTD   ;IF SO, GO TO ASK NEXT QUESTION
 1032  002432  005067  176636                CLR     ACTREG          ;CLEAR DEVICE ACTIVE FLAG TO
 1033                                                                 ;INDICATE NO RUNNING OF MULTIPLE
 1034                                                                 ;DEVICES
 1035  002436  005067  176634                CLR     ROTADD          ;CLEAR DEVICE ADDRESS POINTER IN
 1036                                                                 ;USE WHEN RUNNING MULTIPLE DEVICES
 1037  002442  000167  000160                JMP     CONQUES         ;SKIP ASKING NEXT QUESTION
 1038  002446                          LASTD:
 1039                                  ;WE WILL NOW BEGIN TO SET UP THE DEVICE ACTIVE REGISTER FOR RUNNING
 1040                                  ;MULTIPLE DL11 DEVICES
 1041  002446  104401  017637                TYPE    ,MLASTD         ;ASK USER FOR THE RECEIVER
 1042                                                                 ;CONTROL REGISTER ADDRESS OF
 1043                                                                 ;HIS LAST DEVICE
 1044  002452  104410                        RDOCT                   ;ACCEPT THE ANSWER TYPED BY
```

```
1045                                                    ;USER AND STORE ON TOP OF STACK
1046                                            ;CHECK TO SEE IF THE USER RESPONSE WAS WITHIN LIMITS
1047  002454  012600                            1$:     MOV     (SP)+,R0        ;GET THE ANSWER TYPED
1048  002456  020027  176170                            CMP     R0,#176170      ;IS THE NUMBER TOO HIGH?
1049  002462  101132                                    BHI     RETRY2          ;IF YES - GO TO RETRY SITUATION
1050  002464  020027  175610                            CMP     R0,#175610      ;IS THE NUMBER TOO LOW?
1051  002470  103527                                    BLO     RETRY2          ;IF YES - GO TO RETRY SITUATION
1052  002472  132700  000001                            BITB    #BIT0,R0        ;NUMBER IS IN RANGE BUT IS IT
1053                                                                            ;ON AN EVEN BOUNDARY?
1054  002476  001124                                    BNE     RETRY2          ;IF NOT - GO TO RETRY SITUATION
1055                                            ;CHECK TO SEE IF USER RESPONSE WAS TRULY A RCVR STATUS REGISTER
1056  002500  032700  000007                            BIT     #7,R0           ;WAS THE LEAST SIGNIFICANT DIGIT OF THE
1057                                                                            ;USER RESPONSE EQUAL TO A ZERO?
1058  002504  001121                                    BNE     RETRY2          ;BRANCH IF NOT
1059  002506  010067  176566                            MOV     R0,LASTADD      ;THE LAST ADDRESS VALUE TYPED IS OK
1060                                                                            ;STORE FOR FUTURE USE
1061                                            ;NOW WE BEGIN TO ACTUALLY INITIALIZE THE DEVICE ACTIVE REGISTER
1062                                            ;FROM WHICH THE PROGRAM WILL CYCLE UNTIL ALL DEVICES HAVE BEEN TESTED
1063  002512  012767  000001  176556                    MOV     #1,ROTADD       ;SET UP POINTER FOR 'ACTREG'
1064  002520  005067  176550                            CLR     ACTREG          ;CLEAR DEVICE ACTIVE REGISTER
1065  002524  056767  176546  176542    2$:     BIS     ROTADD,ACTREG   ;MAKE 1ST DEVICE ACTIVE
1066  002532  000241                                    CLC                     ;CLEAR CARRY BIT FOR POINTER
1067                                                                            ;ROTATION
1068  002534  006167  176536                            ROL     ROTADD          ;ARE WE PAST 16 LINE RANGE?
1069  002540  103422                                    BCS     3$              ;BRANCH IF YES
1070  002542  062767  000010  176514                    ADD     #10,BASEADD     ;STEP UP BASE ADDRESS
1071  002550  026767  176524  176506                    CMP     LASTADD,BASEADD ;IS THIS THE LAST DEVICE?
1072  002556  101362                                    BHI     2$              ;BRANCH IF NOT
1073                                            ;NOTE:  IF THIS PATH IS TAKEN IT IS ASSUMED THAT AT LEAST 2 DEVICES
1074                                            ;       EXIST AND THAT ALL ADDRESSING IS CONTIGUOUS
1075  002560  056767  176512  176506                    BIS     ROTADD,ACTREG   ;INDICATE NEXT DEVICE ACTIVE
1076  002566  012767  000001  176502                    MOV     #1,ROTADD       ;RESET POINTER FOR 'ACTREG' FOR
1077                                                                            ;LATER USE IN END OF PASS ROUTINE
1078  002574  016767  176462  176462                    MOV     KEEPAD,BASEADD  ;RESET 1ST DEVICE RECEIVER
1079                                                                            ;CONTROLLER REGISTER ADDRESS FOR
1080                                                                            ;LATER USE IN END OF PASS ROUTINE
1081  002602  000167  000020                            JMP     CONQUES         ;GO TO CONTINUE QUESTIONING OF USER
1082  002605                                    3$:
1083                                            ;IF WE TAKE THIS PATH IT APPEARS THAT THERE ARE NOT AT LEAST
1084                                            ;TWO DEVICES PRESENT - IN RESPONSE TO USER TYPING 'YES' TO MULTIPLE
1085                                            ;DEVICES QUESTION
1086  002606  016767  176450  176450                    MOV     KEEPADD,BASEADD ;RESET 1ST DEVICE RECEIVER
1087                                                                            ;CONTROLLER REGISTER ADDRESS
1088  002614  104401  017735                            TYPE    ,MRANGE         ;INFORM USER TO CHECK AND RETYPE
1089                                                                            ;THE LAST DEVICE RCSR ADDRESS
1090  002620  104410                                    RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
1091                                                                            ;AND STORE ON TOP OF STACK
1092  002622  000167  177626                            JMP     1$
1093  002626                                    CONQUES:
1094                                            ;IF WE HAVE REACHED THIS PORTION WE KNOW:
1095                                            ;       A)  THE 'RXCSR' ADDRESS OF THE 1ST DEVICE
1096                                            ;       B)  THE 'RXCSR' ADDRESS OF THE LAST DEVICE, SAND
1097                                            ;       C)  THE INTERRUPT VECTOR OF THE 1ST DEVICE
1098                                            ;NOW LET'S FIND THE PRIORITY LEVEL
1099  002626  104401  020021                            TYPE    ,PLEVEL         ;ASK USER FOR PRIORITY LEVEL
1100  002632  104410                                    RDOCT                   ;ACCEPT ANSWER TYPED BY USER AND
```

```
                                                        ;STORE ON TOP OF STACK
002634  012600                          MOV     (SP)+,R0        ;GET THE ANSWER TYPED
002636  020027  000007                  CMP     R0,#7           ;IS THE NUMBER TOO HIGH?
002642  101046                          BHI     RETRY3          ;IF YES - GO TO RETRY SITUATION
002644  020027  000004                  CMP     R0,#4           ;'S THE NUMBER TOO LOW?
002650  103443                          BLO     RETRY3          ;IF YES GO TO RETRY SITUATION
002652  010067  176424                  MOV     R0,DLPRI        ;THE PRIORITY TYPED IN IS OK
                                                        ;STORE FOR FUTURE USE


                                ;THIS SECTION WILL CALCULATE THE PRIORITY LEVEL FOR THE
                                ;PROCESSOR BASED ON THE USER RESPONSE FOR PRIORITY LEVEL OF THE
                                ;DEVICE
002656  006367  176420                  ASL     DLPRI           ;FORM BITS <7-5> OF PSW
002662  006367  176416                  ASL     DLPRI           ;       .
002666  006367  176410                  ASL     DLPRI           ;       .
002672  006367  176404                  ASL     DLPRI           ;       .
002676  006367  176400                  ASL     DLPRI           ;
002702  016767  176374  176374          MOV     DLPRI,LESS1     ;START TO FORM LEVEL TO ALLOW
                                                                ;INTERRUPTS
002710  162767  000001  176366          SUB     #1,LESS1        ;DROP DEVICE LEVEL PRIORITY
                                                                ;BY 1 LEVEL FOR PSW
002716  042767  000037  176360          BIC     #37,LESS1       ;MAKE SURE THE T,N,Z,V & C
                                                                ;BITS FOR THE PROCESSOR ARE CLEAR
002724  005077  176464          FLUSH:  CLR     @DLXCSR         ;CLEAR OUT BOTH CSR'S
002730  005077  176454                  CLR     @DLRCSR
002734  005777  176452                  TST     @DLRDBR         ;FLUSH RCVR "DONE" BIT
002740  005777  176446                  TST     @DLRDBR
002744  000167  000020                  JMP     TST1            ;BEGIN TESTING
002750  104401  001252          RETRY2: TYPE    .$QUES          ;TYPE '?' INDICATING USER TYPED
                                                                ;SOMETHING WRONG FOR LAST ADDRESS
002754  000167  177466                  JMP     LASTD   ;GO BACK TO REISSUE QUESTION
002760  104401  001252          RETRY3: TYPE    .$QUES          ;TYPE'?' INDICATING USER TYPED
                                                                ;SOMETHING WRONG FOR PRIORITY
002764  000167  177636                  JMP     CONQUES         ;GO BACK TO REISSUE QUESTION


                                ;;*****************************************************************
                                ;*TEST 1         TEST THAT REFERENCE TO RCSR DOES NOT CAUSE TIMEOUT
                                ;;*****************************************************************
002770  000004                  TST1:   SCOPE
002772  016746  175006                  MOV     ERRVEC,-(SP)    ;SAVE THE TIMEOUT VECTOR
002776  012767  003014  175000          MOV     #1$,ERRVEC      ;GO TO 1$ IF TIMEOUT
003004  016702  176400                  MOV     DLRCSR,R2       ;REGADR = RCSR ADR
003010  005712                          TST     (R2)            ;USE REGADR ON BUS
003012  000407                          BR      3$              ;<GO TO NEXT TEST IF NO TIMEOUT>
003014  004767  011076          1$:     JSR     PC,$UERT1       ;GO SET UP EPROR INFO
003020  012767  003030  176216          MOV     #2$,$ESCAPE     ;RETURN TO 2$ AFTER ERROR PRINT
003026  104001                          ERROR+1                 ;DL REFERENCE CAUSED BUS TIMEOUT
003030  022626                  2$:     CMP     (SP)+,(SP)+     ;CLEAN STACK FROM TIMEOUT
003032  012667  174746          3$:     MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR


                                ;;*****************************************************************
                                ;*TEST 2         TEST THAT REFERENCE TO XCSR DOES NOT CAUSE TIMEOUT
                                ;;*****************************************************************
003036  000004                  TST2:   SCOPE
003040  016746  174740                  MOV     ERRVEC,-(SP)    ;SAVE THE TIMEOUT VECTOR
```

```
1157  003044  012767  003062  174732          MOV     #1$,ERRVEC      ;GO TO 1$ IF TIMEOUT
1158  003052  016702  176336                  MOV     DLXCSR,R2       ;REGADR = XCSR ADR
1159  003056  005712                          TST     (R2)            ;USE REGADR ON BUS
1160  003060  000407                          BR      3$              ;;<GO TO NEXT TEST IF NO TIMEOUT>
1161  003062  004767  011030          1$:     JSR     PC,SUERT1       ;GO SET UP ERROR INFO
1162  003066  012767  003076  176150          MOV     #2$,$ESCAPE     ;RETURN TO 2$ AFTER ERROR PRINT
1163  003074  104001                          ERROR+1                 ;DL REFERENCE CAUSED BUS TIMEOUT
1164  003076  022626                  2$:     CMP     (SP)+,(SP)+     ;CLEAN STACK FROM TIMEOUT
1165  003100  012667  174700          3$:     MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR
1166
1167                                          ;;**********************************************************************
1168                                          ;*TEST 3         TEST THAT REFERENCE TO RDBR DOES NOT CAUSE TIMEOUT
1169                                          ;;**********************************************************************
1170  003104  000004                  TST3:   SCOPE
1171  003106  016746  174672                  MOV     ERRVEC,-(SP)    ;SAVE THE TIMEOUT VECTOR
1172  003112  012767  003130  174664          MOV     #1$,ERRVEC      ;GO TO 1$ IF TIMEOUT
1173  003120  016702  176266                  MOV     DLRDBR,R2       ;REGADR = RDBR ADR
1174  003124  005712                          TST     (R2)            ;USE REGADR ON BUS
1175  003126  000407                          BR      3$              ;;<GO TO NEXT TEST IF NO TIMEOUT>
1176  003130  004767  010762          1$:     JSR     PC,SUERT1       ;GO SET UP ERROR INFO
1177  003134  012767  003144  176102          MOV     #2$,$ESCAPE     ;RETURN TO 2$ AFTER ERROR PRINT
1178  003142  104001                          ERROR+1                 ;DL REFERENCE CAUSED BUS TIMEOUT
1179  003144  022626                  2$:     CMP     (SP)+,(SP)+     ;CLEAN STACK FROM TIMEOUT
1180  003146  012667  174632          3$:     MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR
1181
1182                                          ;;**********************************************************************
1183                                          ;*TEST 4         TEST THAT REFERENCE TO XDBR DOES NOT CAUSE TIMEOUT
1184                                          ;;**********************************************************************
1185  003152  000004                  TST4:   SCOPE
1186  003154  016746  174624                  MOV     ERRVEC,-(SP)    ;SAVE THE TIMEOUT VECTOR
1187  003160  012767  003176  174616          MOV     #1$,ERRVEC      ;GO TO 1$ IF TIMEOUT
1188  003166  016702  176224                  MOV     DLXDBR,R2       ;REGADR = XDBR ADR
1189  003172  005712                          TST     (R2)            ;USE REGADR ON BUS
1190  003174  000407                          BR      3$              ;;<GO TO NEXT TEST IF NO TIMEOUT>
1191  003176  004767  010714          1$:     JSR     PC,SUERT1       ;GO SET UP ERROR INFO
1192  003202  012767  003212  176034          MOV     #2$,$ESCAPE     ;RETURN TO 2$ AFTER ERROR PRINT
1193  003210  104001                          ERROR+1                 ;DL REFERENCE CAUSED BUS TIMEOUT
1194  003212  022626                  2$:     CMP     (SP)+,(SP)+     ;CLEAN STACK FROM TIMEOUT
1195  003214  012667  174564          3$:     MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR
1196
1197                                          ;;**********************************************************************
1198                                          ;*TEST 5         TEST THAT RCSR IS ALL ZEROES ON ENTRY
1199                                          ;;**********************************************************************
1200  003220  000004                  TST5:   SCOPE
1201  003222  005004                          CLR     R4              ;RESULT IN RCSR S/B = 0
1202  003224  016702  176160                  MOV     DLRCSR,R2       ;REGADR = RCSR ADR
1203  003230  020412                          CMP     R4,(R2)         ;[RCSR]=000000 ??
1204  003232  001403                          BEQ     TST6            ;;<BR IF YES>
1205  003234  004767  010576                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
1206  003240  104002                          ERROR+2                 ;RCSR NOT CLEAR ON START UP
1207                                          ;;**********************************************************************
1208                                          ;*TEST 6         TEST THAT "READY" BIT IS ONLY BIT SET IN XCSR
1209                                          ;;**********************************************************************
1210  003242  000004                  TST6:   SCOPE
1211  003244  012704  000200                  MOV     #200,R4         ;RESULT IN XCSR S/B = 000200
1212  003250  016702  176140                  MOV     DLXCSR,R2       ;REGADR = XCSR ADR
```

```
1213  003254  020412                          CMP     R4,(R2)         ;[XCSR]=000200 ??
1214  003256  001403                          BEQ     TST7            ;;<BR IF YES>
1215  003260  004767  010552                  JSR     PC,SUER2        ;GO SETUP ERROR INFO
1216  003264  104002                          ERROR+2                 ;[XCSR] INCORRECT ON START UP
1217                                  ;;**************************************************************
1218                                  ;*TEST 7         TEST THAT "MAINT" BIT CAN BE SET AND CLEARED
1219                                  ;;**************************************************************
1220  003266  000004                  TST7:   SCOPE
1221  003270  012704  000204                  MOV     #204,R4         ;RESULT IN XCSR S/B = 000204
1222  003274  016702  176114                  MOV     DLXCSR,R2       ;REGADR = XCSR ADR
1223  003300  052712  000004                  BIS     #BIT2,(R2)      ;SET THE "MAINT" BIT
1224  003304  020412                          CMP     R4,(R2) ;RESULT IN XCSR OK ??
1225  003306  001403                          BEQ     1$              ;;<BR IF YES>
1226  003310  004767  010522                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
1227  003314  104002                          ERROR+2                 ;MAINT. BIT FAILED TO SET PROPERLY
1228  003316  012704  000200          1$:     MOV     #200,R4         ;RESULT IN XCSR S/B = 000200
1229  003322  042712  000004                  BIC     #BIT2,(R2)      ;NOW CLEAR THE "MAINT" BIT
1230  003326  020412                          CMP     R4,(R2)         ;RESULT IN XCSR OK ??
1231  003330  001403                          BEQ     TST10           ;;<BR IF YES>
1232  003332  004767  010500                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
1233  003336  104002                          ERROR+2                 ;MAINT BIT FAILED TO CLEAR PROPERLY
1234                                  ;;**************************************************************
1235                                  ;*TEST 10        TEST THAT XMIT I.E. CAN CAUSE AN INTR
1236                                  ;;**************************************************************
1237  003340  000004                  TST10:  SCOPE
1238  003342  005067  176076                  CLR     INTFLG          ;INIT SOFTWARE INTR FLAG
1239  003346  016705  176046                  MOV     DLVECT,R5       ;GET VECTOR ADDRESS
1240  003352  012765  003440  000004          MOV     #2$,4(R5)       ;GO TO 4$ ON INTR
1241  003360  016765  175716  000006          MOV     DLPRI,6(R5)     ;PRIORITY LEVEL 4
1242  003366  005005                          CLR     R5              ;INIT INTR. TIMER
1243  003370  012704  000200                  MOV     #200,R4         ;RESULT IN XCSR S/B = 000200
1244  003374  016702  176014                  MOV     DLXCSR,R2       ;REGADR = XCSR ADR
1245  003400  052712  000100                  BIS     #100,(R2)       ;SET INTR. ENABLE BIT 06
1246  003404  005767  176034          1$:     TST     INTFLG          ;DID INTR OCCUR YET ??
1247  003410  001020                          BNE     3$              ;BR IF IT DID
1248  003412  005305                          DEC     R5              ;COUNT THE TIMER
1249  003414  001373                          BNE     1$              ;BR IF NO TIMEOUT
1250  003416  012704  000300                  MOV     #300,R4         ;RESULT IN XCSR S/B = 000300
1251  003422  004767  010410                  JSR     PC,SUER2        ;GO SETUP ERROR INFO
1252  003426  012767  003436  175610          MOV     #4$,$ESCAPE     ;RETURN TO 4$ AFTER ERROR PRINT
1253  003434  104002                          ERROR+2                 ;INTR.  FAILED
1254  003436                          4$:
1255  003436  000412                          BR      TST11           ;;<GO TO NEXT TEST>
1256  003440  005167  176000          2$:     COM     INTFLG          ;SET THE SOFTWARE FLAG
1257  003444  042712  000100                  BIC     #100,(R2)       ;TURN OFF I.E. BIT
1258  003450  000002                          RTI                     ;RETURN CONTROL TO INTR. ROUTINE
1259  003452  020412                  3$:     CMP     R4,(R2)         ;RESULT IN XCSR OK ??
1260  003454  001403                          BEQ     TST11           ;;<BR IF YES>
1261  003456  004767  010354                  JSR     PC,SUER2        ;GO SET UP ERROR INFO.
1262  003462  104002                          ERROR+2                 ;XMIT INTR. NOT SERVICED PROPERLY
1263                                  ;;**************************************************************
1264                                  ;*TEST 11        TEST THAT RCVR I.E. BIT CAN BE SET AND CLEARED
1265                                  ;;**************************************************************
1266  003464  000004                  TST11:  SCOPE
1267  003466  012704  000100                  MOV     #100,R4         ;RESULT IN RCSR S/B = 000100
1268  003472  016702  175712                  MOV     DLRCSR,R2       ;REGADR = RCSR ADR
```

```
1269  003476  052712  000100              BIS     #BIT6,(R2)      ;SET I.E. BIT
1270  003502  020412                      CMP     R4,(R2)         ;DID IT SET PROPERLY ??
1271  003504  001403                      BEQ     1$              ;;<BR IF YES>
1272  003506  004767  010324              JSR     PC,SUER2        ;GO SET UP ERROR INFO.
1273  003512  104002                      ERROR+2                 ;RCVR I.E. BIT FAILED TO SET PROPERLY
1274  003514  005004              1$:     CLR     R4              ;RESULT IN RCSR S/B = 000000
1275  003516  042712  000100              BIC     #BIT6,(R2)      ;CLEAR THE I.E. BIT
1276  003522  020412                      CMP     R4,(R2)         ;DID IT CLEAR PROPERLY ??
1277  003524  001403                      BEQ     TST12           ;;<BR IF YES>
1278  003526  004767  010304              JSR     PC,SUER2        ;GO SET UP ERROR INFO
1279  003532  104002                      ERROR+2                 ;RCVR I.E. BIT FAILED TO CLEAR PROPERLY
1280                                  ;;**********************************************************************
1281                                  ;*TEST 12        TEST THAT RCVR "DONE" CAN GENERATE AN INTR.
1282                                  ;;**********************************************************************
1283  003534  000004              TST12:  SCOPE
1284  003536  016705  175656              MOV     DLVECT,R5       ;GET THE VECTOR ADDRESS
1285  003542  012725  003702              MOV     #3$,(R5)+       ;GO TO 3$ ON RCVR INTR.
1286  003546  016715  175530              MOV     DLPRI,(R5)      ;AT LEVEL 4
1287  003552  005067  175666              CLR     INTFLG          ;INIT THE SOFTWARE FLAG
1288  003556  005005                      CLR     R5              ;INIT INTR. TIMER
1289  003560  105067  175420              CLRB    $TMP1           ;INIT WHERE DATA WILL BE STORED
1290  003564  016702  175620              MOV     DLRCSR,R2       ;REGADR = RCSR ADR
1291  003570  005012                      CLR     (R2)            ;INIT THE RCSR TO 000000
1292  003572  052712  000100              BIS     #BIT6,(R2)      ;ENABLE RCVR INTERRUPTS
1293  003576  052762  000004  000004      BIS     #BIT2,4(R2)     ;NOW TURN ON MAINT MODE
1294  003604  112762  000252  000006      MOVB    #252,6(R2)      ;LOAD XMIT BUFFER REG.
1295  003612  005767  175626      1$:     TST     INTFLG          ;DID RCVR INTR. YET ??
1296  003616  001044                      BNE     4$              ;BR IF IT DID
1297  003620  005305                      DEC     R5              ;COUNT THE TIMER
1298  003622  001373                      BNE     1$              ;BR IF NO TIMEOUT
1299  003624  013767  177776  175350      MOV     @#PSW,$TMP0     ;SAVE ERROR PSW
1300  003632  042762  000004  000004      BIC     #BIT2,4(R2)     ;DISABLE MAINT MODE
1301  003640  042712  000100              BIC     #100,(R2)       ;DISABLE RCVR INTR.
1302  003644  010667  175326              MOV     SP,$REG6        ;SAVE THE ERROR SP
1303  003650  010201                      MOV     R2,R1           ;DEVADR = RCSR ADR
1304  003652  011203                      MOV     (R2),R3         ;GET THE WAS DATA
1305  003654  012704  000200              MOV     #200,R4         ;[RCSR] S/B = 000200
1306  003660  004767  010200              JSR     PC,SUERR1       ;GO SET UP ERROR INFO.
1307  003664  012767  003674  175352      MOV     #2$,$ESCAPE     ;RETURN TO 2$ AFTER ERROR ALWAYS
1308  003672  104002                      ERROR+2                 ;RCVR INTERRUPT FAILED
1309  003674  005762  000002      2$:     TST     2(R2)           ;REFERENCE RCVR DATA BUFFER
1310                                                              ;TO CLEAR RCSR IN CASE RCVR
1311                                                              ;INTERRUPTS COULD NOT BE ENABLED
1312  003700  000440                      BR      TST13           ;;<GO TO NEXT TEST>
1313  003702  042762  000004  000004  3$: BIC     #BIT2,4(R2)     ;DISABLE THE MAINT MODE
1314  003710  116267  000002  175266      MOVB    2(R2),$TMP1     ;GET THE RECEIVED DATA
1315  003716  042712  000100              BIC     #BIT6,(R2)      ;TURN OFF RCVR INTR. ENAB
1316  003722  005167  175516              COM     INTFLG          ;SET THE SOFTWARE FLAG
1317  003726  000002                      RTI                     ;RETURN TO MAINLINE
1318  003730  005004              4$:     CLR     R4              ;[RCSR] S/B=0
1319  003732  005712                      TST     (R2)            ;IS IT ALL ZEROES ??
1320  003734  001403                      BEQ     5$              ;;<BR IF YES>
1321  003736  004767  010074              JSR     PC,SUER2        ;GO SET UP ERROR INFO
1322  003742  104002                      ERROR+2                 ;RCVR INTR NOT SERVICED PROPERLY
1323  003744  016701  175442      5$:     MOV     DLRDBR,R1       ;SAVE WAS ADDRESS
1324  003750  016702  175442              MOV     DLXDBR,R2       ;SAVE THE S/B ADDRESS
```

```
1325  003754  016703  175224                  MOV     $TMP1,R3        ;GET THE WAS DATA
1326  003760  042703  177400                  BIC     #177400,R3      ;REMOVE JUNK FROM HI BYTE
1327  003764  012704  000252                  MOV     #252,R4         ;DATA S/B = 252
1328  003770  020403                          CMP     R4,R3           ;WAS = S/B ??
1329  003772  001403                          BEQ     TST13           ;;<BR IF YES>
1330  003774  004767  010064                  JSR     PC,SUERR1       ;GO SET UP THE ERROR INFO
1331  004000  104003                          ERROR+3                 ;DATA COMPARE ERROR
1332                                  ;;**************************************************************
1333                                  ;*TEST 13        TEST THAT "REQ TO SEND" ASSERTS "RING"
1334                                  ;;**************************************************************
1335  004002  000004                  TST13:  SCOPE
1336  004004  032767  010000  175126          BIT     #SW12,SWR       ;ARE WE TESTING /C OR /D MODEL?
1337  004012  001047                          BNE     TST14           ;;<BRANCH IF YES>
1338  004014  012704  140004                  MOV     #140004,R4      ;RESULT IN RCSR S/B = 140004
1339  004020  016702  175364                  MOV     DLRCSR,R2       ;REGADR = RCSR ADR
1340  004024  005012                          CLR     (R2)            ;INIT THE RCSR TO 000000
1341  004026  052712  000004                  BIS     #BIT2,(R2)      ;SET "REQ TO SEND"
1342  004032  032777  100000  175350          BIT     #BIT15,@DLRCSR  ;DID "RING" SET "DATA SET INT" ?
1343  004040  001003                          BNE     1$              ;;<BR IF YES>
1344  004042  004767  007770                  JSR     PC,SUER2        ;GO SET UP ERROR INFO.
1345  004046  104002                          ERROR+2                 ;"RING" TRANSITION FAILED TO SET "DATA SET INT"
1346                                                                  ;NOTE: "BIT  #BIT15,(R2)" RESETS BIT15
1347  004050  012704  040004          1$:     MOV     #40004,R4       ;RESULT IN RCSR S/B = 40004
1348  004054  020412                          CMP     R4,(R2)         ;BOTH "RING" AND "REQ TO SEND" ASSERTED ?
1349  004056  001403                          BEQ     2$              ;;<BR IF YES>
1350  004060  004767  007752                  JSR     PC,SUER2        ;GO SET UP ERROR INFO.
1351  004064  104002                          ERROR+2                 ;"RING" OR "REQ TO SEND" FAILED TO SET
1352  004066  005004                  2$:     CLR     R4              ;RESULT IN RCSR S/B = 000000
1353  004070  042712  000004                  BIC     #BIT2,(R2)      ;CLEAR "REQ TO SEND"
1354  004074  032777  100000  175306          BIT     #BIT15,@DLRCSR  ;DID "DATA SET INT" GET SET ??
1355  004102  001403                          BEQ     3$              ;;<BR IF NOT>
1356  004104  004767  007726                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
1357  004110  104002                          ERROR+2                 ;CLEARING "RING" SET "DATA SET INT"
1358  004112  020412                  3$:     CMP     R4,(R2)         ;RCSR CONTAIN ALL ZEROES ??
1359  004114  001406                          BEQ     TST14           ;;<BR IF YES>
1360  004116  004767  007714                  JSR     PC,SUER2        ;GO SET UP ERROR INFO.
1361  004122  016767  000002  175050          MOV     .+6,$REG7       ;SAVE THE ERROR PC
1362  004130  104002                          ERROR+2                 ;CLEARING  "REQ TO SEND" FAILED TO CLEAR "RING"
1363                                  ;;**************************************************************
1364                                  ;*TEST 14        TEST THAT "SEC XMIT" ASSERTS "SEC REC" AND "DATA SET INT"
1365                                  ;;**************************************************************
1366  004132  000004                  TST14:  SCOPE
1367  004134  032767  010000  174776          BIT     #SW12,SWR       ;ARE WE TESTING /C OR /D MODEL?
1368  004142  001046                          BNE     TST15           ;;<BRANCH IF YES>
1369  004144  016702  175240                  MOV     DLRCSR,R2       ;REGADR = RCSR ADR
1370  004150  005012                          CLR     (R2)            ;INIT RCSR TO 000000
1371  004152  012704  102010                  MOV     #102010,R4      ;CONTENTS OF RCSR S/B = 102010
1372  004156  052712  000010                  BIS     #BIT3,(R2)      ;SET "SEC XMIT" BIT
1373  004162  032777  100000  175220          BIT     #BIT15,@DLRCSR  ;DID "DATA SET INT" SET ??
1374  004170  001003                          BNE     1$              ;;<BR IF YES>
1375  004172  004767  007640                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
1376  004176  104002                          ERROR+2                 ;"DATA SET INT" FAILED TO SET-NOTE THAT
1377                                                                  ;"BIT  #BIT15,(R2)" RESETS BIT15
1378  004200  012704  002010          1$:     MOV     #2010,R4        ;RESULT IN RCSR S/B = 2010
1379  004204  020412                          CMP     R4,(R2)         ;ARE "SEC XMIT" AND "SEC REC" BOTH SET ?
1380  004206  001403                          BEQ     2$              ;;<BR IF YES>
```

```
1381  004210  004767  007622                JSR     PC,SUER2        ;GO SET UP ERROR INFO
1382  004214  104002                        ERROR+2                 ;"SEC XMIT" OR "SEC REC" FAILED TO SET
1383                                                                  ;OR "DATA SET INT" FAILED TO BE CLEARED
1384                                                                  ;WHEN REFERENCING RCSR
1385  004216  012704  100000        2$:     MOV     #BIT15,R4       ;RESULT IN RCSR S/B = 100000
1386  004222  042712  000010                BIC     #BIT3,(R2)      ;CLEAR "SEC XMIT" BIT
1387  004226  032777  100000  175154        BIT     #BIT15,@DLRCSR  ;DID CLEARING IT SET "DATA SET INT"??
1388  004234  001003                        BNE     3$              ;<BR IF YES>
1389  004236  004767  007574                JSR     PC,SUER2        ;GO SET UP ERROR INFO.
1390  004242  104002                        ERROR+2                 ;CLEARING "SEC XMIT" FAILED TO SET "DATA
1391                                                                  ;SET INT. (NOTE THAT REFERENCING RCSR
1392                                                                  ; CLEARS "DATA SET INT"
1393  004244  005004        3$:     CLR     R4              ;RESULT IN RCSR S/B = 000000
1394  004246  020412                CMP     R4,(R2)         ;"SEC XMIT" AND "SEC REC" CLEAR ?
1395  004250  001403                BEQ     TST15           ;<BR IF YES>
1396  004252  004767  007560        JSR     PC,SUER2        ;GO SETUP ERROR INFO
1397  004256  104002                ERROR+2                 ;"SEC XMIT" OR "SEC REC" FAILED TO CLEAR
1398                                                          ;OR REFERENCING RCSR  FAILED TO CLEAR "DATA SET INT"
1399                                ;;*****************************************************************
1400                                ;*TEST 15       TEST THAT "DTR" CAN ASSERT "CLR TO SEND" AND "CAR DET"
1401                                ;;*****************************************************************
1402  004260  000004        TST15:  SCOPE
1403  004262  032767  010000  174650        BIT     #SW12,SWR       ;ARE WE TESTING /C OR /D MODEL?
1404  004270  001046                        BNE     TST16           ;<BRANCH IF YES>
1405  004272  016702  175112                MOV     DLRCSR,R2       ;REGADR = RCSR ADR
1406  004276  005012                        CLR     (R2)            ;INIT RCSR TO 000000
1407  004300  012704  130002                MOV     #130002,R4      ;RESULT IN RCSR S/B = 130002
1408  004304  052712  000002                BIS     #BIT1,(R2)      ;SET "DTR" BIT
1409  004310  032777  100000  175072        BIT     #BIT15,@DLRCSR  ;DID "DATA SET INT" SET ??
1410  004316  001003                        BNE     1$              ;<BR IF YES>
1411  004320  004767  007512                JSR     PC,SUER2        ;GO SET UP ERROR INFO.
1412  004324  104002                        ERROR+2                 ;"DATA SET INT" FAILED TO SET -
1413                                                                  ;NOTE: THE REFERENCE TO RCSR  ABOVE WILL
1414                                                                  ;WILL UNCONDITIONALLY CLEAR RCSR BIT 15.
1415  004326  012704  030002        1$:     MOV     #30002,R4       ;RESULT IN RCSR S/B = 30002
1416  004332  020412                CMP     R4,(R2)         ;"DTR" "CLR TO SEND", AND "CAR DET" ALLSET
1417  004334  001403                BEQ     2$              ;<BR IF ALL SET>
1418  004336  004767  007474        JSR     PC,SUER2        ;GO SET UP ERROR INFO
1419  004342  104002                ERROR+2                 ;"DTR" "CLR TO SEND" OR "CAR DET" FAILED
1420                                                          ;TO SET OR "DATA SET INT" FAILED TO CLEAR
1421  004344  012704  100000        2$:     MOV     #BIT15,R4       ;RESULT IN RCSR S/B = 100000
1422  004350  042712  000002                BIC     #BIT1,(R2)      ;NOW CLEAR "DTR"
1423  004354  032777  100000  175026        BIT     #BIT15,@DLRCSR  ;DID "DATA SET INT" SET ??
1424  004362  001003                        BNE     3$              ;<BR IF YES>
1425  004364  004767  007446                JSR     PC,SUER2        ;GO SETUP ERROR INFO
1426  004370  104002                        ERROR+2                 ;"DATA SET INT" FAILED TO SET WHEN "DTR"
1427                                                                  ;WENT TO A ZERO.
1428  004372  005004        3$:     CLR     R4              ;RESULT IN RCSR S/B = 000000
1429  004374  020412                CMP     R4,(R2)         ;DID ALL BITS CLEAR??
1430  004376  001403                BEQ     TST16           ;<BR IF YES>
1431  004400  004767  007432        JSR     PC,SUER2        ;GO SET UP ERROR INFO
1432  004404  104002                ERROR+2                 ;"DTR" "CLR TO SEND" OR "CAR DET" FAILED
1433                                                          ;TO CLEAR PROPERLY
1434                                ;;*****************************************************************
1435                                ;*TEST 16       TEST THAT "DATA SET INT ENAB" CAN SET AND CLEAR
1436                                ;;*****************************************************************
```

```
 1437  004406  000004                    TST16:  SCOPE
 1438  004410  032767  010000  174522            BIT     #SW12,SWR       ;ARE WE TESTING /C OR /D MODEL?
 1439  004416  001023                            BNE     TST17           ;;<BRANCH IF YES>
 1440  004420  016702  174764                    MOV     DLRCSR,R2       ;REGADR = RCSR ADR
 1441  004424  012704  000040                    MOV     #40,R4          ;RESULT IN RCSR S/B = 000040
 1442  004430  052712  000040                    BIS     #BIT5,(R2)      ;SET THE "DATA SET I.E." BIT
 1443  004434  020412                            CMP     R4,(R2)         ;DID IT SET OK ??
 1444  004436  001403                            BEQ     1$              ;;<BR IF YES>
 1445  004440  004767  007372                    JSR     PC,SUER2        ;GO SET UP ERROR INFO
 1446  004444  104002                            ERROR+2                 ;"DAT SET I. E." FAILED TO SET
 1447  004446  005004                    1$:     CLR     R4              ;MAKE S/B DATA = 000000
 1448  004450  042712  000040                    BIC     #BIT5,(R2)      ;NOW CLEAR THE "DATA SET I.E." BIT
 1449  004454  020412                            CMP     R4,(R2)         ;DID IT CLEAR OK ??
 1450  004456  001403                            BEQ     TST17           ;;<BR IF YES>
 1451  004460  004767  007352                    JSR     PC,SUER2        ;GO SET UP ERROR INFO.
 1452  004464  104002                            ERROR+2                 ;"DATA SET I.E." FAILED TO CLEAR
 1453                                    ;;**********************************************************************
 1454                                    ;*TEST 17        TEST THE "DATA SET I.E." CAN CAUSE A RCVR INTR
 1455                                    ;;**********************************************************************
 1456  004466  000004                    TST17:  SCOPE
 1457  004470  032767  010000  174442            BIT     #SW12,SWR       ;ARE WE TESTING A /C OR /D MODEL?
 1458  004476  001054                            BNE     TST20           ;;<BRANCH IF YES>
 1459  004500  016705  174714                    MOV     DLVECT,R5       ;GET THE VECTOR ADDR
 1460  004504  012725  004572                    MOV     #3$,(R5)+       ;GO TO 3$ ON RCVR INTR.
 1461  004510  016715  174566                    MOV     DLPRI,(R5)      ;AT LEVEL 4
 1462  004514  005005                            CLR     R5              ;INIT INTR. TIMER
 1463  004516  005067  174722                    CLR     INTFLG          ;INIT SOFTWARE FLAG
 1464  004522  005004                            CLR     R4              ;RESULT IN RCSR S/B = 0 AFTER INTR.
 1465  004524  016702  174660                    MOV     DLRCSR,R2       ;REGADR = RCSR ADR
 1466  004530  052712  000040                    BIS     #BIT5,(R2)      ;SET THE "DATA SET I.E." BIT
 1467  004534  052712  000002                    BIS     #BIT1,(R2)      ;NOW SET "DTR" TO GEN INTR.
 1468  004540  005767  174700            1$:     TST     INTFLG          ;DID INTR OCCUR YET ??
 1469  004544  001016                            BNE     4$              ;BR IF YES
 1470  004546  005305                            DEC     R5              ;COUNT THE TIMER
 1471  004550  001[illegible]                    BNE     1$              ;BR IF NO TIMEOUT
 1472  004552  004[illegible]  007260            JSR     PC,SUER2        ;GO SET UP ERROR INFO
 1473  004556  005012                            CLR     (R2)            ;TURN IT ALL OFF
 1474  004560  012767  004570  174456            MOV     #2$,$ESCAPE     ;COME BACK TO 2$ IN ALL CASES
 1475  004566  104002                            ERROR+2                 ;"DATA SET" INTR FAILED TO OCCUR
 1476  004570                            2$:
 1477  004570  000417                            BR      TST20           ;;<GO TO NEXT TEST>
 1478  004572  005012                    3$:     CLR     (R2)            ;ZERO THE RCSR
 1479  004574  005167  174644                    COM     INTFLG          ;SET THE SOFTWARE FLAG
 1480  004600  000002                            RTI                     ;RETURN TO SENDER
 1481  004602  032712  100000            4$:     BIT     #BIT15,(R2)     ;DID "DATA SET INT" GET SET BY INTR. SERVICE ??
 1482  004606  001003                            BNE     5$              ;;<BR IF YES>
 1483  004610  004767  007222                    JSR     PC,SUER2        ;GO SET UP ERROR INFO
 1484  004614  104002                            ERROR+2                 ;DATA SET INTR. NOT SERVICED PROPERLY
 1485  004616  020412                    5$:     CMP     R4,(R2)         ;ALL BITS IN RCSR CLEAR ??
 1486  004620  001403                            BEQ     TST20           ;;<BR IF YES>
 1487  004622  004767  007210                    JSR     PC,SUER2        ;GO SET UP ERROR INFO
 1488  004626  104002                            ERROR+2                 ;INTR. SERVICE FAILED TO CLEAR RCSR
 1489                                    ;;**********************************************************************
 1490                                    ;*TEST 20        TEST THAT THE "BREAK" BIT CAN BE SET AND CLEARED
 1491                                    ;;**********************************************************************
 1492  004630  000004                    TST20:  SCOPE
```

```
1493  004632  032767  010000  174300          BIT     #SW12,SWR       ;ARE WE TESTING /C OR /D MODEL?
1494  004640  001024                          BNE     TST21           ;;<BRANCH IF YES>
1495  004642  012704  000201                  MOV     #201,R4         ;RESULT S/B = 201 IN XCSR
1496  004646  016702  174542                  MOV     DLXCSR,R2       ;SET UP REGADR
1497  004652  052712  000001                  BIS     #BIT0,(R2)      ;SET THE "BREAK" BIT
1498  004656  020412                          CMP     R4,(R2)         ;DID IT SET PROPERLY ??
1499  004660  001403                          BEQ     1$              ;;<BR IF YES>
1500  004662  004767  007150                  JSR     PC,SUER2        ;GO SET UP ERROR INFO.
1501  004666  104002                          ERROR+2                 ;"BREAK" BIT FAILED TO SET PROPERLY
1502  004670  012704  000200          1$:     MOV     #200,R4         ;RESULT S/B = 200 IN XCSR
1503  004674  042712  000001                  BIC     #BIT0,(R2)      ;CLEAR THE "BREAK" BIT
1504  004700  020412                          CMP     R4,(R2)         ;DID IT CLEAR PROPERLY ??
1505  004702  001403                          BEQ     TST21           ;;<BR IF YES>
1506  004704  004767  007126                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
1507  004710  104002                          ERROR+2                 ;"BREAK" FAILED TO CLEAR PROPERLY
1508
1509
1510                                  ;;*****************************************************************
1511                                  ;*TEST 21        TEST THAT A "RESET" CLEARS THE "BREAK" BIT
1512                                  ;;*****************************************************************
1513  004712  000004                  TST21:  SCOPE
1514  004714  012704  000200                  MOV     #200,R4         ;RESULT S/B = 200
1515  004720  016702  174470                  MOV     DLXCSR,R2       ;SET UP REGADR
1516  004724  052712  000001                  BIS     #BIT0,(R2)      ;SET THE "BREAK" BIT
1517  004730  000005                          RESET                   ;CLEAR IT WITH A "RESET"
1518  004732  020412                          CMP     R4,(R2)         ;DID IT CLEAR ??
1519  004734  001403                          BEQ     TST22           ;;<BR IF YES>
1520  004736  004767  007074                  JSR     PC,SUER2        ;GO SET UP ERROR INFO.
1521  004742  104002                          ERROR+2                 ;RESET INSTR. FAILED TO CLEAR "BREAK"
1522
```

```
                                ;;**********************************************************************
                                ;*TEST 22       TEST TO TURN AROUND NULL-DEL-NULL PATTERN
                                ;;**********************************************************************
005744  000004                  TST22:  SCOPE
005746  012767  000001  174266          MOV     #1,$TIMES       ;;DO 1 ITERATION
005754  004767  007212                  JSR     PC,SUVEC        ;GO SET UP VECTORS
005760  005067  174444                  CLR     RTRY            ;INITIALIZE RETRY FLAG
005764  012767  014346  174444  1$:     MOV     #LDOUT1,LDOUT   ;SET POINTER TO LOAD ROUTINE
005772  004767  007222                  JSR     PC,PRIME        ;GO SET UP BUFFERS AND DEVICE
005776  005767  174420          2$:     TST     XFLGO           ;ANY HARD XMIT ERRORS ??
005002  001040                          BNE     5$              ;BR IF YES
005004  005767  174414                  TST     RFLGO           ;ANY HARD RECEIVER ERROR ??
005010  001053                          BNE     7$              ;BR IF YES
005012  005767  174410                  TST     RFLG1           ;ANY SOFT RECEIVER ERRORS ??
005016  001065                          BNE     9$              ;BR IF YES
005020  022767  022322  174406          CMP     #BUFEND,IPTR    ;RECEIVED 256. BYTES ??
005026  001003                          BNE     3$              ;BR IF NOT
005030  004767  007456                  JSR     PC,CHKDAT       ;GO CHECK THE DATA BUFFERS
005034  000500                          BR      TST23           ;;<GO TO NEXT TEST>
005036  005367  174376          3$:     DEC     TIMR1           ;DEC TIMEOUT COUNTER 1
005042  001355                          BNE     2$              ;BR IF NO TIMEOUT
005044  005367  174372                  DEC     TIMR2           ;DEC TIMEOUT COUNTER 2
005050  001352                          BNE     2$              ;BR IF NO TIMEOUT
005052  042777  000100  174330          BIC     #100,@DLRCSR    ;TURN OFF THE INTRS.
005060  042777  000104  174326          BIC     #104,@DLXCSR
005066  104401  016326                  TYPE    ,XMSG1          ;GO TYPE TIMEOUT MESSAGE
005072  012767  005102  174144          MOV     #4$,$ESCAPE     ;GO TO 4$ AFTER ERROR PRINT
005100  104000                          ERROR                   ;PRINT ERROR PC
005102                          4$:
005102  000455                          BR      TST23           ;;<GO TO NEXT TEST>
005104  016701  174300          5$:     MOV     DLRCSR,R1       ;PUT DEVADR IN R1
005110  016702  174300                  MOV     DLXCSR,R2       ;PUT REGADR IN R2
005114  011203                          MOV     (R2),R3         ;GET THE WAS DATA
005116  012704  000204                  MOV     #204,R4         ;PUT S/B DATA IN R4
005122  004767  006736                  JSR     PC,SUERR1       ;GO SET UP ERROR INFO
005126  012767  005136  174110          MOV     #6$,$ESCAPE     ;GO TO 6$ AFTER PRINTING ERROR
005134  104002                          ERROR+2                 ;TRANSMITTER FALSE INTERRUPT
005136                          6$:
005136  000437                          BR      TST23           ;;<GO TO NEXT TEST>
005140  016701  174244          7$:     MOV     DLRCSR,R1       ;SAVE THE DEVADR
005144  010102                          MOV     R1,R2           ;SAVE THE REGADR
005146  011203                          MOV     (R2),R3         ;GET THE WAS DATA
005150  012704  000200                  MOV     #200,R4         ;RESULT S/B = 200
005154  004767  006704                  JSR     PC,SUERR1       ;GO SET UP ERROR INFO
005160  012767  005170  174056          MOV     #8$,$ESCAPE     ;GO TO 8$ AFTER ERROR PRINT
005166  104002                          ERROR+2                 ;RECEIVER FALSE INTERRUPT
005170                          8$:
005170  000422                          BR      TST23           ;;<GO TO NEXT TEST>
005172  016701  174212          9$:     MOV     DLRCSR,R1       ;SAVE THE DEVADR
005176  016702  174210                  MOV     DLRDBR,R2       ;SAVE REGADR
005202  016703  173776                  MOV     $TMP1,R3        ;GET CONTENTS OF ERROR RDBR
005206  004767  006652                  JSR     PC,SUERR1       ;GO SETUP ERROR INFO
005212  012767  005222  174024          MOV     #10$,$ESCAPE    ;GO TO 10$ AFTER ERROR PRINT
005220  104005                          ERROR+5                 ;REPORT SOFT ERROR (PARITY,FRAMING, OR OVERRUN
005222  005267  174202          10$:    INC     RTRY            ;COUNT ONE TRY
005226  022767  000003  174174          CMP     #3,RTRY         ;TRIED THREE TIMES
```

```
1579  005234  001253                    BNE     1$              ;BR IF NOT
```

```
1580                                    ;;*****************************************************************
1581                                    ;*TEST 23       TEST TO TURN AROUND BINARY UP COUNT PATTERN
1582                                    ;;*****************************************************************
1583  005236  000004              TST23:  SCOPE
1584  005240  012767  000001  173774        MOV     #1,$TIMES       ;;DO 1 ITERATION
1585  005246  004767  006720              JSR     PC,SUVEC        ;GO SET UP VECTORS
1586  005252  005067  174152              CLR     RTRY            ;INITIALIZE RETRY FLAG
1587  005256  012767  014370  174152  1$:  MOV     #LDOUT2,LDOUT   ;SET POINTER TO LOAD ROUTINE
1588  005264  004767  006730              JSR     PC,PRIME        ;GO SET UP BUFFERS AND DEVICE
1589  005270  005767  174126        2$:   TST     XFLGO           ;ANY HARD XMIT ERRORS ??
1590  005274  001040                      BNE     5$              ;BR IF YES
1591  005276  005767  174122              TST     RFLGO           ;ANY HARD RECEIVER ERROR ??
1592  005302  001053                      BNE     7$              ;BR IF YES
1593  005304  005767  174116              TST     RFLG1           ;ANY SOFT RECEIVER ERRORS ??
1594  005310  001065                      BNE     9$              ;BR IF YES
1595  005312  022767  022322  174114      CMP     #BUFEND,IPTR    ;RECEIVED 256. BYTES ??
1596  005320  001003                      BNE     3$              ;BR IF NOT
1597  005322  004767  007164              JSR     PC,CHKDAT       ;GO CHECK THE DATA BUFFERS
1598  005326  000500                      BR      TST24           ;;<GO TO NEXT TEST>
1599  005330  005367  174104        3$:   DEC     TIMR1           ;DEC TIMEOUT COUNTER 1
1600  005334  001355                      BNE     2$              ;BR IF NO TIMEOUT
1601  005336  005367  174100              DEC     TIMR2           ;DEC TIMEOUT COUNTER 2
1602  005342  001352                      BNE     2$              ;BR IF NO TIMEOUT
1603  005344  042777  000100  174036      BIC     #100,@DLRCSR    ;TURN OFF THE INTRS.
1604  005352  042777  000104  174034      BIC     #104,@DLXCSR
1605  005360  104401  016405              TYPE    ,XMSG2          ;GO TYPE TIMEOUT MESSAGE
1606  005364  012767  005374  173655      MOV     #4$,$ESCAPE     ;GO TO 4$ AFTER ERROR PRINT
1607  005372  104000                      ERROR                   ;PRINT ERROR PC
1608  005374                        4$:
1609  005374  000455                      BR      TST24           ;;<GO TO NEXT TEST>
1610  005376  016701  174006        5$:   MOV     DLRCSR,R1       ;PUT DEVADR IN R1
1611  005402  016702  174006              MOV     DLXCSR,R2       ;PUT REGADR IN R2
1612  005406  011203                      MOV     (R2),R3         ;GET THE WAS DATA
1613  005410  012704  000204              MOV     #204,R4         ;PUT S/B DATA IN R4
1614  005414  004767  006444              JSR     PC,SUERR1       ;GO SET UP ERROR INFO
1615  005420  012767  005430  173616      MOV     #6$,$ESCAPE     ;GO TO 6$ AFTER PRINTING ERROR
1616  005426  104002                      ERROR+2                 ;TRANSMITTER FALSE INTERRUPT
1617  005430                        6$:
1618  005430  000437                      BR      TST24           ;;<GO TO NEXT TEST>
1619  005432  016701  173752        7$:   MOV     DLRCSR,R1       ;SAVE THE DEVADR
1620  005436  010102                      MOV     R1,R2           ;SAVE THE REGADR
1621  005440  011203                      MOV     (R2),R3         ;GET THE WAS DATA
1622  005442  012704  000200              MOV     #200,R4         ;RESULT S/B = 200
1623  005446  004767  006412              JSR     PC,SUERR1       ;GO SET UP ERROR INFO
1624  005452  012767  005462  173564      MOV     #8$,$ESCAPE     ;GO TO 8$ AFTER ERROR PRINT
1625  005460  104002                      ERROR+2                 ;RECEIVER FALSE INTERRUPT
1626  005462                        8$:
1627  005462  000422                      BR      TST24           ;;<GO TO NEXT TEST>
1628  005464  016701  173720        9$:   MOV     DLRCSR,R1       ;SAVE THE DEVADR
1629  005470  016702  173716              MOV     DLRDBR,R2       ;SAVE REGADR
1630  005474  016703  173504              MOV     $TMP1,R3        ;GET CONTENTS OF ERROR RDBR
1631  005500  004767  006360              JSR     PC,SUERR1       ;GO SETUP ERROR INFO
1632  005504  012767  005514  173532      MOV     #10$,$ESCAPE    ;GO TO 10$ AFTER ERROR PRINT
1633  005512  104005                      ERROR+5                 ;REPORT SOFT ERROR (PARITY,FRAMING, OR OVERRUN
1634  005514  005267  173710       10$:   INC     RTRY            ;COUNT ONE TRY
1635  005520  022767  000003  173702      CMP     #3,RTRY         ;TRIED THREE TIMES
```

```
  1636  005526  001253                    BNE     1$              ;BR IF NOT
```

```
                                        ;;*****************************************************************
                                        ;*TEST 24       TEST TO TURN AROUND BINARY DOWN COUNT PATTERN
                                        ;;*****************************************************************
005530  000004                  TST24:  SCOPE
005532  012767  000001  173502          MOV     #1,$TIMES       ;;DO 1 ITERATION
005540  004767  006426                  JSR     PC,SUVEC        ;GO SET UP VECTORS
005544  005067  173660                  CLR     RTRY            ;INITIALIZE RETRY FLAG
005550  012767  014410  173660  1$:     MOV     #LDOUT3,LDOUT   ;SET POINTER TO LOAD ROUTINE
005556  004767  006436                  JSR     PC,PRIME        ;GO SET UP BUFFERS AND DEVICE
005562  005767  173634          2$:     TST     XFLGO           ;ANY HARD XMIT ERRORS ??
005566  001040                          BNE     5$              ;BR IF YES
005570  005767  173630                  TST     RFLGO           ;ANY HARD RECEIVER ERROR ??
005574  001053                          BNE     7$              ;BR IF YES
005576  005767  173624                  TST     RFLG1           ;ANY SOFT RECEIVER ERRORS ??
005602  001065                          BNE     9$              ;BR IF YES
005604  022767  022322  173622          CMP     #BUFEND,IPTR    ;RECEIVED 256. BYTES ??
005612  001003                          BNE     3$              ;BR IF NOT
005614  004767  006672                  JSR     PC,CHKDAT       ;GO CHECK THE DATA BUFFERS
005620  000500                          BR      TST25           ;;<GO TO NEXT TEST>
005622  005367  173612          3$:     DEC     TIMR1           ;DEC TIMEOUT COUNTER 1
005626  001355                          BNE     2$              ;BR IF NO TIMEOUT
005630  005367  173606                  DEC     TIMR2           ;DEC TIMEOUT COUNTER 2
005634  001352                          BNE     2$              ;BR IF NO TIMEOUT
005636  042777  000100  173544          BIC     #100,@DLRCSR    ;TURN OFF THE INTRS.
005644  042777  000104  173542          BIC     #104,@DLXCSR
005652  104401  016466                  TYPE    ,XMSG3          ;GO TYPE TIMEOUT MESSAGE
005656  012767  005666  173360          MOV     #4$,$ESCAPE     ;GO TO 4$ AFTER ERROR PRINT
005664  104000                          ERROR                   ;PRINT ERROR PC
005666                          4$:
005666  000455                          BR      TST25           ;;<GO TO NEXT TEST>
005670  016701  173514          5$:     MOV     DLRCSR,R1       ;PUT DEVADR IN R1
005674  016702  173514                  MOV     DLXCSR,R2       ;PUT REGADR IN R2
005700  011203                          MOV     (R2),R3         ;GET THE WAS DATA
005702  012704  000204                  MOV     #204,R4         ;PUT S/B DATA IN R4
005706  004767  006152                  JSR     PC,SUERR1       ;GO SET UP ERROR INFO
005712  012767  005722  173324          MOV     #6$,$ESCAPE     ;GO TO 6$ AFTER PRINTING ERROR
005720  104002                          ERROR+2                 ;TRANSMITTER FALSE INTERRUPT
005722                          6$:
005722  000437                          BR      TST25           ;;<GO TO NEXT TEST>
005724  016701  173460          7$:     MOV     DLRCSR,R1       ;SAVE THE DEVADR
005730  010102                          MOV     R1,R2           ;SAVE THE REGADR
005732  011203                          MOV     (R2),R3         ;GET THE WAS DATA
005734  012704  000200                  MOV     #200,R4         ;RESULT S/B = 200
005740  004767  006120                  JSR     PC,SUERR1       ;GO SET UP ERROR INFO
005744  012767  005754  173272          MOV     #8$,$ESCAPE     ;GO TO 8$ AFTER ERROR PRINT
005752  104002                          ERROR+2                 ;RECEIVER FALSE INTERRUPT
005754                          8$:
005754  000422                          BR      TST25           ;;<GO TO NEXT TEST>
005756  016701  173426          9$:     MOV     DLRCSR,R1       ;SAVE THE DEVADR
005762  016702  173424                  MOV     DLRDBR,R2       ;SAVE REGADR
005766  016703  173212                  MOV     $TMP1,R3        ;GET CONTENTS OF ERROR RDBR
005772  004767  006066                  JSR     PC,SUERR1       ;GO SETUP ERROR INFO
005776  012767  006006  173240          MOV     #10$,$ESCAPE    ;GO TO 10$ AFTER ERROR PRINT
006004  104005                          ERROR+5                 ;REPORT SOFT ERROR (PARITY,FRAMING, OR OVERRUN
006006  005267  173416          10$:    INC     RTRY            ;COUNT ONE TRY
006012  022767  000003  173410          CMP     #3,RTRY         ;TRIED THREE TIMES
```

```
  1693  003020  001253                    BNE     1$              ;BR IF NOT
```

```
                                        ;;*****************************************************************
                                        ;*TEST 25    TEST TO TURN AROUND WORST CASE PATTERN
                                        ;;*****************************************************************
006022  000004                  TST25:  SCOPE
006024  012767  000001  173210          MOV     #1,$TIMES       ;;DO 1 ITERATION
006032  004767  006134                  JSR     PC,SUVEC        ;GO SET UP VECTORS
006036  005067  173366                  CLR     RTRY            ;INITIALIZE RETRY FLAG
006042  012767  014444  173366  1$:     MOV     #LDOUT4,LDOUT   ;SET POINTER TO LOAD ROUTINE
006050  004767  006144                  JSR     PC,PRIME        ;GO SET UP BUFFERS AND DEVICE
006054  005767  173342          2$:     TST     XFLGO           ;ANY HARD XMIT ERRORS ??
006060  001042                          BNE     5$              ;BR IF YES
006062  005767  173336                  TST     RFLGO           ;ANY HARD RECEIVER ERROR ??
006066  001056                          BNE     7$              ;BR IF YES
006070  005767  173332                  TST     RFLG1           ;ANY SOFT RECEIVER ERRORS ??
006074  001071                          BNE     9$              ;BR IF YES
006076  022767  022322  173330          CMP     #BUFEND,IPTR    ;RECEIVED 256. BYTES ??
006104  001004                          BNE     3$              ;BR IF NOT
006106  004767  006400                  JSR     PC,CHKDAT       ;GO CHECK THE DATA BUFFERS
006112  000167  002012                  JMP     $EOP                    ;GO TO NEXT TEST
006116  005367  173316          3$:     DEC     TIMR1           ;DEC TIMEOUT COUNTER 1
006122  001354                          BNE     2$              ;BR IF NO TIMEOUT
006124  005367  173312                  DEC     TIMR2           ;DEC TIMEOUT COUNTER 2
006130  001351                          BNE     2$              ;BR IF NO TIMEOUT
006132  042777  000100  173250          BIC     #100,@DLRCSR    ;TURN OFF THE INTRS.
006140  042777  000104  173246          BIC     #104,@DLXCSR
006146  104401  016551                  TYPE    .XMSG4          ;GO TYPE TIMEOUT MESSAGE
006152  012767  006162  173064          MOV     #4$,$ESCAPE     ;GO TO 4$ AFTER ERROR PRINT
006160  104000                          ERROR                   ;PRINT ERROR PC
006162  000167  001742          4$:     JMP     $EOP            ;GO TO NEXT TEST
006166  016701  173216          5$:     MOV     DLRCSR,R1       ;PUT DEVADR IN R1
006172  016702  173216                  MOV     DLXCSR,R2       ;PUT REGADR IN R2
006176  011203                          MOV     (R2),R3         ;GET THE WAS DATA
006200  012704  000204                  MOV     #204,R4         ;PUT S/B DATA IN R4
006204  004767  005654                  JSR     PC,SUERR1       ;GO SET UP ERROR INFO
006210  012767  006220  173026          MOV     #6$,$ESCAPE     ;GO TO 6$ AFTER PRINTING ERROR
006216  104002                          ERROR+2                 ;TRANSMITTER FALSE INTERRUPT
006220  000167  001704          6$:     JMP     $EOP            ;GO TO NEXT TEST
006224  016701  173160          7$:     MOV     DLRCSR,R1       ;SAVE THE DEVADR
006230  010102                          MOV     R1,R2           ;SAVE THE REGADR
006232  011203                          MOV     (R2),R3         ;GET THE WAS DATA
006234  012704  000200                  MOV     #200,R4         ;RESULT S/B = 200
006240  004767  005620                  JSR     PC,SUERR1       ;GO SET UP ERROR INFO
006244  012767  006254  172772          MOV     #8$,$ESCAPE     ;GO TO 8$ AFTER ERROR PRINT
006252  104002                          ERROR+2                 ;RECEIVER FALSE INTERRUPT
006254  000167  001650          8$:     JMP     $EOP            ;GO TO NEXT TEST
006260  016701  173124          9$:     MOV     DLRCSR,R1       ;SAVE THE DEVADR
006264  016702  173122                  MOV     DLRDBR,R2       ;SAVE REGADR
006270  016703  172710                  MOV     $TMP1,R3        ;GET CONTENTS OF ERROR RDBR
006274  004767  005564                  JSR     PC,SUERR1       ;GO SETUP ERROR INFO
006300  012767  006310  172736          MOV     #10$,$ESCAPE    ;GO TO 10$ AFTER ERROR PRINT
006306  104005                          ERROR+5                 ;REPORT SOFT ERROR (PARITY,FRAMING, OR OVERRUN
006310  005267  173114          10$:    INC     RTRY            ;COUNT ONE TRY
006314  022767  000003  173106          CMP     #3,RTRY         ;TRIED THREE TIMES
006322  001247                          BNE     1$              ;BR IF NOT
006324  000167  001600                  JMP     $EOP            ;GO TO END OF PASS ROUTINE
```

```
                                        ;THIS IS PROGRAM #2
                                        ;THE FOLLOWING USER UTILITY PROGRAM WILL ALLOW:
                                        ;       A)  SELECTION OF A TRANSMITTER DATA BUFFER
                                        ;       B)  SELECTION OF A CHARACTER FOR CONTINUOUS TRANSFER
                                        ;       C)  SELECTION OF AN EXPIRATION TIME IN MILLISECONDS
                                        ;           BETWEEN EACH TRANSMITTER DATA BUFFER CHARACTER TRANSFER
                                        ;       D)  A TIGHT SCOPE LOOP LOCK ON A SPECIFIC CHARACTER
                                        ;
006330  012706  001100          PRG2:   MOV     #STACK,SP       ;INITIALIZE THE STACK POINTER
006334  012737  013052  000034          MOV     #$TRAP,@#TRAPVEC        ;TRAP VECTOR FOR TRAP CALLS
006342  012737  000340  000036          MOV     #340,@#TRAPVEC+2        ;LEVEL 7
006350  012737  010732  000030          MOV     #$ERROR,@#EMTVEC        ;EMT VECTOR FOR ERROR ROUTINE
006356  012737  000340  000032          MOV     #340,@#EMTVEC+2         ;LEVEL 7
006364  104401  016664                  TYPE    ,PROG2M         ;INDICATE THAT USER SELECTED
                                                                ;PROGRAM #2
006370  104401  020306          PRG2A:  TYPE    ,LINTAD         ;ASK USER FOR THE TRANSMITTER
                                                                ;DATA BUFFER ADDRESS OF THE DEVICE
                                                                ;HE WISHES TO TEST
006374  104410                          RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
                                                                ;AND STORE ON TOP OF STACK
                                        ;CHECK TO SEE IF THE USER RESPONSE WAS WITHIN LIMITS
006376  012602                          MOV     (SP)+,R2        ;GET THE ANSWER TYPED
006400  020227  176176                  CMP     R2,#176176      ;IS THE NUMBER TOO HIGH?
006404  101065                          BHI     REDO1           ;IF YES - GO TO RETRY SITUATION
006406  020227  175616                  CMP     R2,#175616      ;IS THE NUMBER TOO LOW?
006412  103462                          BLO     REDO1           ;IF YES - GO TO RETRY SITUATION
006414  132702  000001                  BITB    #BIT0,R2        ;NUMBER IS IN RANGE BUT IS IT
                                                                ;ON AN EVEN BOUNDARY?
006420  001057                          BNE     REDO1           ;IF NOT GO TO RETRY SITUATION
                                        ;CHECK TO SEE IF USER RESPONSE WAS TRULY A XMIT BUFFER REGISTER
006422  010203                          MOV     R2,R3           ;GET THE USER RESPONSE
006424  142703  000370                  BICB    #370,R3         ;MASK OFF LOWER BYTE EXCEPT FOR
                                                                ;LEAST SIGNIFICANT DIGIT
006430  122703  000006                  CMPB    #6,R3           ;WAS THE LEAST SIGNIFICANT DIGIT OF THE
                                                                ;USER RESPONSE EQUAL TO A SIX?
006434  001051                          BNE     REDO1           ;BRANCH IF NOT
006436  010267  172540                  MOV     R2,$TMPO        ;THE TRANSMITTER ADDRESS
                                                                ;TYPED IS OK - STORE FOR
                                                                ;FUTURE USE
                                        ;NOW CHECK TO MAKE SURE THE DEVICE IS PRESENT
006442  016746  171336                  MOV     ERRVEC,-(SP)    ;SAVE THE TIMEOUT VECTOR
006446  012767  006460  171330          MOV     #2$,ERRVEC      ;SET UP TIMEOUT SERVICE ADDRESS
006454  005712                          TST     (R2)            ;IF PRESENT WE WILL EXECUTE THE
                                                                ;NEXT INSTRUCTION - IF NOT
                                                                ;WE GO TO 2$:
006456  000412                          BR      4$              ;BRANCH IF PRESENT
006460  004767  005460          2$:     JSR     PC,SUERT2       ;GO SET UP FOR ERROR INFORMATION
006464  012767  006474  172552          MOV     #3$,$ESCAPE     ;POINT OF RETURN AFTER ERROR REPORT
006472  104006                          ERROR   +6              ;XDBR REFERENCE CAUSED TIMEOUT
006474  022626                  3$:     CMP     (SP)+,(SP)+     ;CLEAN STACK FROM TIMEOUT
006476  012667  171302                  MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR
006502  000426                          BR      REDO1           ;GO TO RETRY SITUATION
006504  012667  171274          4$:     MOV     (SP)+,ERRVEC    ;DEVICE REGISTER IS PRESENT!
                                                                ;RESTORE TIMEOUT VECTOR
                                        ;WE ARE NOW READY FOR THE CHARACTER TO BE TRANSMITTED, AND THE
```

E04


```
1806                                    ;DELAY TIME (IN MILLISECONDS) THAT IS TO TRANSPIRE BETWEEN
1807                                    ;SUCCESSIVE CHARACTER TRANSFERS
1808  006510  104401  020367            PRG2B:  TYPE    ,SELCAR         ;ASK USER FOR THE CHARACTER HE
1809                                                                    ;WISHES TO TRANSFER
1810  006514  104410                            RDOCT                   ;ACCEPT THE ANSWER TYPED BY
1811                                                                    ;USER AND STORE ON TOP OF STACK
1812  006516  012667  172462                    MOV     (SP)+,$TMP1     ;GET THE ANSWER TYPED
1813                                    ;NOTE:  THE USER RESPONSE FOR THE CHARACTER WAS TO BE THE
1814                                    ;       OCTAL ASCII EQUIVALENT OF THE CHARACTER E.G. A=101
1815  006522  104401  020475                    TYPE    ,SELDLY         ;ASK THE USER FOR THE DELAY
1816                                                                    ;IN MSEC (OCTAL NO.) BETWEEN
1817                                                                    ;CHARACTER TRANSFERS
1818  006526  104410                            RDOCT                   ;ACCEPT THE ANSWER TYPED BY
1819                                                                    ;USER AND STORE ON TOP OF STACK
1820  006530  012667  172452                    MOV     (SP)+,$TMP2     ;GET THE ANSWER TYPED
1821  006534  116767  172446  000012  1$:       MOVB    $TMP2,2$        ;SET THE DELAY COUNT ARGUMENT
1822                                                                    ;FOR TIMER ROUTINE
1823  006542  116777  172436  172432            MOVB    $TMP1,@$TMP0    ;LOAD THE TRANSMITTER DATA
1824                                                                    ;BUFFER WITH THE CHARACTER
1825  006550  004767  004750                    JSR     PC,DELAY        ;GO OFF TO WAIT THE SPECIFIED
```

```
                                                                ;NO. OF MSEC. BEFORE ISSUING
                                                                ;ANOTHER CHARACTER
006554  000000                  2$:     .WORD   0               ;THIS IS WHERE THE DELAY COUNT RESIDES
006556  000766                          BR      1$              ;GO BACK TO ISSUE ANOTHER CHARACTER
006560  104401  001252          RED01:  TYPE    ,$QUES          ;TYPE A QUESTION MARK(?)
006564  000167  177600                  JMP     PRG2A           ;REITERATE THE XDBR QUESTION TO USER
                                ;THIS IS PROGRAM #3
                                ;THE FOLLOWING USER UTILITY PROGRAM WILL ALLOW:
                                ;       A)  SELECTION OF A RECEIVER DATA BUFFER
                                ;       B)  SELECTION OF A CHARACTER FOR CONTINUOUS TRANSFER
                                ;       C)  SELECTION OF AN EXPIRATION TIME IN MILLISECONDS
                                ;           BETWEEN EACH RECEIVER DATA BUFFER CHARACTER TRANSFER
                                ;       D)  A TIGHT SCOPE LOOP LOCK ON A SPECIFIC CHARACTER
                                ;
006570  012706  001100          PRG3:   MOV     #STACK,SP       ;INITIALIZE THE STACK POINTER
006574  012737  013052  000034          MOV     #$TRAP,@#TRAPVEC        ;TRAP VECTOR FOR TRAP CALLS
006602  012737  000340  000036          MOV     #340,@#TRAPVEC+2        ;LEVEL 7
006610  012737  010732  000030          MOV     #$ERROR,@#EMTVEC        ;EMT VECTOR FOR ERROR ROUTINE
006616  012737  000340  000032          MOV     #340,@#EMTVEC+2         ;LEVEL 7
006624  104401  016730                  TYPE    ,PRG3M          ;INDICATE THAT USER SELECTED
                                                                ;PROGRAM #3
006630  104401  020570          PRG3A:  TYPE    ,LINRAD         ;ASK USER FOR THE RECEIVER DATA
                                                                ;BUFFER ADDRESS OF THE DEVICE
                                                                ;HE WISHES TO TEST
006634  104410                          RDOCT                   ;ACCEPT THE ANSWER TYPED BY
                                                                ;USER AND STORE ON TOP OF STACK
                                ;CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
006636  012602                          MOV     (SP)+,R2        ;GET THE ANSWER TYPED
006640  020227  176172                  CMP     R2,#176172      ;IS THE NUMBER TOO HIGH?
006644  101071                          BHI     RED02           ;IF YES - GO TO RETRY SITUATION
006646  020227  175612                  CMP     R2,#175612      ;IS THE NUMBER TOO LOW?
006652  103466                          BLO     RED02           ;IF YES - GO TO RETRY SITUATION
006654  132702  000001                  BITB    #BIT0,R2        ;NUMBER IS IN RANGE BUT IS IT
                                                                ;ON AN EVEN BOUNDARY?
006660  001063                          BNE     RED02           ;IF NOT - GO TO RETRY SITUATION
                                ;CHECK TO SEE IF USER RESPONSE WAS TRULY A RCVR DBR ADDRESS
006662  010203                          MOV     R2,R3           ;GET THE USER RESPONSE
006664  142703  000370                  BICB    #370,R3         ;MASK OFF LOWER BYTE EXCEPT FOR
                                                                ;LEAST SIGNIFICANT DIGIT
005670  122703  000002                  CMPB    #2,R3           ;WAS THE LEAST SIGNIFICANT DIGIT OF THE
                                                                ;USER RESPONSE EQUAL TO A TWO?
006674  001055                          BNE     RED02           ;BRANCH IF NOT
006676  010267  172300                  MOV     R2,$TMP0        ;THE RECEIVER ADDRESS TYPED IS
                                                                ;OK - STORE FOR FUTURE USE
                                ;NOW CHECK TO MAKE SURE THE DEVICE IS PRESENT
006702  016746  171076                  MOV     ERRVEC,-(SP)    ;SAVE THE TIMEOUT VECTOR
006706  012767  006720  171070          MOV     #2$,ERRVEC      ;SET UP TIMEOUT SERVICE ADDRESS
006714  005712                          TST     (R2)            ;IF PRESENT WE WILL EXECUTE THE
                                                                ;NEXT INSTRUCTION - IF NOT WE
                                                                ;GO TO 2$:
006716  000412                          BR      4$              ;BRANCH IF PRESENT
006720  004767  005220          2$:     JSR     PC,SUERT2       ;GO SET UP FOR ERROR INFORMATION
006724  012767  006734  172312          MOV     #3$,$ESCAPE     ;POINT OF RETURN AFTER ERROR REPORT
006732  104006                          ERROR   +6              ;RDBR REFERENCE CAUSED TIMEOUT
006734  022626                  3$:     CMP     (SP)+,(SP)+     ;CLEAN STACK FROM TIMEOUT
```

```
1882  006736  012667  171042                  MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR
1883  006742  000432                          BR      REDO2           ;GO TO RETRY SITUATION
1884  006744  012667  171034          4$:     MOV     (SP)+,ERRVEC    ;DEVICE REGISTER IS PRESENT!
1885                                                                  ;RESTORE TIMEOUT VECTOR
1886                                  ;WE ARE NOW READY FOR THE CHARACTER TO BE TRANSMITTED, AND THE
1887                                  ;DELAY TIME (IN MILLISECONDS) THAT IS TO TRANSPIRE BETWEEN SUCCESSIVE
1888                                  ;CHARACTER TRANSFERS
1889  006750  104401  020367          PRG3B:  TYPE    ,SELCAR         ;ASK USER FOR THE CHARACTER
1890                                                                  ;HE WISHES TO TRANSFER
1891  006754  104410                          RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
1892                                                                  ;AND STORE ON TOP OF STACK
1893  006756  012667  172222                  MOV     (SP)+,$TMP1     ;GET THE ANSWER TYPED
1894                                  ;NOTE:  THE USER RESPONSE FOR THE CHARACTER WAS TO BE THE
1895                                  ;       OCTAL ASCII EQUIVALENT OF THE CHARACTER E.G. B=102
1896  006762  104401  020475                  TYPE    ,SELDLY         ;ASK THE USER FOR THE DELAY
1897                                                                  ;IN MSEC (OCTAL NO.) BETWEEN
1898                                                                  ;CHARACTER TRANSFERS
1899  006766  104410                          RDOCT                   ;ACCEPT THE ANSWER TYPED BY
1900                                                                  ;USER AND STORE ON TOP OF STACK
1901  006770  012667  172212                  MOV     (SP)+,$TMP2     ;GET THE ANSWER TYPED
1902  006774  062702  000002                  ADD     #2,R2           ;GET THE CORRESPONDING XCSR
1903                                                                  ;ADDRESS FOR RECEIVER UNDER-
1904                                                                  ;GOING TEST
1905  007000  052712  000004          1$:     BIS     #BIT2,(R2)      ;SET MAINTENANCE BIT IN XCSR
1906  007004  116767  172176  000012          MOVB    $TMP2,2$        ;SET THE DELAY COUNT ARGUMENT
1907                                                                  ;FOR TIMER ROUTINE
1908  007012  116777  172166  172162          MOVB    $TMP1,@$TMP0    ;LOAD THE RECEIVER DATA BUFFER
1909                                                                  ;WITH THE CHARACTER
1910  007020  004767  004500                  JSR     PC,DELAY        ;GO OFF TO WAIT THE SPECIFIED
1911                                                                  ;NO. OF MSEC. BEFORE ISSUING
1912                                                                  ;ANOTHER CHARACTER
1913  007024  000000          2$:     .WORD   0                       ;THIS IS WHERE THE DELAY COUNT RESIDES
1914  007026  000764                  BR      1$                      ;GO BACK TO ISSUE ANOTHER CHARACTER
1915  007030  104401  001252          REDO2:  TYPE    ,$QUES          ;TYPE A QUESTION MARK(?)
1916  007034  000167  177570                  JMP     PRG3A           ;REITERATE THE RDBR QUESTION TO
1917                                                                  ;USER
1918
1919                                  ;THIS IS PROGRAM #4
1920                                  ;THE FOLLOWING USER UTILITY PROGRAM WILL ALLOW:
1921                                  ;       A)  SELECTION OF A TRANSMITTER DATA BUFFER
1922                                  ;       B)  SELECTION OF A SINGLE CHARACTER TO BE SENT, RECEIVED
1923                                  ;           AND CHECKED WITH MAINTENANCE BIT SET
1924                                  ;
1925
1926  007040  012706  001100          PRG4:   MOV     #STACK,SP       ;INITIALIZE THE STACK POINTER
1927  007044  012737  013052  000034          MOV     #$TRAP,@#TRAPVEC        ;TRAP VECTOR FOR TRAP CALLS
1928  007052  012737  000340  000036          MOV     #340,@#TRAPVEC+2        ;LEVEL 7
1929  007060  012737  010732  000030          MOV     #$ERROR,@#EMTVEC        ;EMT VECTOR FOR ERROR ROUTINE
1930  007066  012737  000340  000032          MOV     #340,@#EMTVEC+2         ;LEVEL 7
1931  007074  104401  016774                  TYPE    ,PRG4M          ;INDICATE THAT USER SELECTED
1932                                                                  ;PROGRAM #4
1933  007100  104401  020306          PRG4A:  TYPE    ,LINTAD         ;ASK USER FOR THE TRANSMITTER
1934                                                                  ;DATA BUFFER ADDRESS OF THE
1935                                                                  ;DEVICE HE WISHES TO TEST
1936  007104  104410                          RDOCT                   ;ACCEPT THE ANSWER TYPED BY
1937                                                                  ;USER AND STORE ON TOP OF STACK
```

```
1938                                    ;CHECK TO SEE IF THE USER RESPONSE WAS WITHIN LIMITS
1939  007106  012602                            MOV     (SP)+,R2        ;GET THE ANSWER TYPED
1940  007110  020227  176176                    CMP     R2,#176176      ;IS THE NUMBER TOO HIGH?
1941  007114  101136                            BHI     REDO3           ;IF YES - GO TO RETRY SITUATION
1942  007116  020227  175616                    CMP     R2,#175616      ;IS THE NUMBER TOO LOW?
1943  007122  103533                            BLO     REDO3           ;IF YES - GO TO RETRY SITUATION
1944  007124  132702  000001                    BITB    #BIT0,R2        ;NUMBER IS IN RANGE BUT IS IT
1945                                                                    ;ON AN EVEN BOUNDARY?
1946  007130  001130                            BNE     REDO3           ;IF NO - GO TO RETRY SITUATION
1947                                    ;CHECK TO SEE IF USER RESPONSE WAS TRULY A XMIT BUFFER REGISTER
1948  007132  010203                            MOV     R2,R3           ;GET THE USER RESPONSE
1949  007134  142703  000370                    BICB    #370,R3         ;MASK OFF LOWER BYTE EXCEPT FOR
1950                                                                    ;LEAST SIGNIFICANT DIGIT
1951  007140  122703  000006                    CMPB    #6,R3           ;WAS THE LEAST SIGNIFICANT DIGIT OF THE
1952                                                                    ;USER RESPONSE EQUAL TO A SIX?
1953  007144  001122                            BNE     REDO3           ;BRANCH IF NOT
1954  007146  010267  172030                    MOV     R2,$TMP0        ;THE TRANSMITTER ADDRESS TYPED
1955                                                                    ;IS OK - STORE FOR FUTURE USE
1956                                    ;NOW CHECK TO MAKE SURE THE DEVICE IS PRESENT
1957  007152  016746  170626                    MOV     ERRVEC,-(SP)    ;SAVE THE TIMEOUT VECTOR
1958  007156  012767  007170  170620            MOV     #2$,ERRVEC      ;SET UP TIMEOUT SERVICE ADDRESS
1959  007164  005712                            TST     (R2)            ;IF PRESENT WE WILL EXECUTE THE
1960                                                                    ;NEXT INSTRUCTION - IF NOT WE
1961                                                                    ;GO TO 2$:
1962  007166  000412                            BR      4$              ;BRANCH IF PRESENT
1963  007170  004767  004750           2$:     JSR     PC,SUERT2       ;GO SET UP FOR ERROR INFORMATION
1964  007174  012767  007204  172042            MOV     #3$,$ESCAPE     ;POINT OF RETURN AFTER ERROR REPORT
1965  007202  104006                            ERROR   +6              ;XDBR REFERENCE CAUSED TIMEOUT
1966  007204  022626                    3$:     CMP     (SP)+,(SP)+     ;CLEAN STACK FROM TIMEOUT
1967  007206  012667  170572                    MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR
1968  007212  000477                            BR      REDO3           ;GO TO RETRY SITUATION
1969  007214  012667  170564           4$:     MOV     (SP)+,ERRVEC    ;DEVICE REGISTER IS PRESENT!
1970                                                                    ;RESTORE TIMEOUT VECTOR
1971  007220  104401  020646                    TYPE    ,RSTALL         ;ASK THE USER IF HE DESIRES SOME
1972                                                                    ;RANDOM NO. OF MSEC. WAIT TIME
1973                                                                    ;BEFORE CHECKING FOR XCSR DONE
1974                                                                    ;FLAG
1975  007224  104410                            RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
1976                                                                    ;AND STORE ON TOP OF STACK
1977  007226  012667  171754                    MOV     (SP)+,$TMP2     ;GET THE ANSWER TYPED
1978                                    ;WE ARE NOW READY FOR THE CHARACTER TO BE TRANSMITTED
1979  007232  104401  020367           PRG4B:  TYPE    ,SELCAR         ;ASK USER FOR THE CHARACTER HE
1980                                                                    ;WISHES TO TRANSFER
1981  007236  104410                            RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
1982                                                                    ;AND STORE ON TOP OF STACK
1983  007240  012667  171740                    MOV     (SP)+,$TMP1     ;GET THE ANSWER TYPED
1984                                    ;NOTE:  THE USER RESPONSE FOR THE CHARACTER WAS TO BE THE OCTAL
1985                                    ;       ASCII EQUIVALENT OF THE CHARACTER E.G. C=103
1986                                    ;
1987  007244  104401  017217           PRG4C:  TYPE,   LENGTH  ;ASK USER FOR THE CHARACTER LENGTH
1988                                                                    ;FOR WHICH HIS DEVICE IS SET
1989  007250  104411                            RDDEC                   ;ACCEPT THE ANSWER TYPED BY USER
1990                                    ;CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
1991  007252  012600                            MOV     (SP)+,R0        ;GET THE ANSWER TYPED
1992  007254  020027  000010                    CMP     R0,#8.          ;IS THE NUMBER TOO HIGH?
1993  007260  101060                            BHI     REDO3A          ;IF YES - GO TO RETRY SITUATION
```

```
1994  007262  020027  000005            CMP     R0,#5.          ;IS THE NUMBER TOO LOW?
1995  007266  103455                    BLO     RED03A          ;IF YES - GO TO RETRY SITUATION
1996  007270  010067  171740            MOV     R0,$TMP15       ;THE VALUE TYPED IS OK
1997                                                            ;STORE FOR FUTURE USE
1998  007274  016767  171702  171706    MOV     $TMP0,$TMP3     ;GET THE XDBR ADDRESS
1999  007302  162767  000002  171700    SUB     #2,$TMP3        ;FORM THE XCSR ADDRESS
2000  007310  005767  171672        1$: TST     $TMP2           ;DO WE RANDOM STALL?
2001  007314  001402                    BEQ     2$              ;BRANCH IF IT WASN'T DESIRED
2002  007316  004767  004246            JSR     PC,STALL        ;GO STALL RANDOM VALUE OF MSEC.
2003  007322  004767  004352        2$: JSR     PC,TIMETX       ;GO WAIT FOR TRANSMITTER DONE
2004                                                            ;BIT TO SET
2005  007326  104401  017104            TYPE    ,XDB            ;TYPE TRANSMITTER DONE BIT MESSAGE
2006  007332  104000                    ERROR   +0              ;XCSR DONE BIT NEVER SET
2007  007334  052777  000004  171646    BIS     #BIT2,@$TMP3    ;SET THE MAINTENANCE BIT IN THE
2008                                                            ;TRANSMITTER CONTROL STATUS REGISTER
2009  007342  016777  171636  171632    MOV     $TMP1,@$TMP0    ;LOAD TRANSMITTER DATA BUFFER
2010                                                            ;WITH SELECTED CHARACTER
2011  007350  004767  004306            JSR     PC,TIMERX       ;GO WAIT FOR RECEIVER DONE BIT
2012                                                            ;TO SET
2013  007354  104401  017153            TYPE    ,RDB            ;TYPE RECEIVER DONE BIT MESSAGE
2014  007360  104000                    ERROR   +0              ;RCSR DONE BIT NEVER SET
2015  007362  016767  171622  171622    MOV     $TMP3,$TMP4     ;GET THE TRANSMITTER CONTROL
2016                                                            ;STATUS REGISTER ADDRESS
2017  007370  162767  000002  171614    SUB     #2,$TMP4        ;FORM THE RECEIVER DATA BUFFER
2018                                                            ;ADDRESS
2019  007376  017767  171610  171610    MOV     @$TMP4,$TMP5    ;STORE THE CHARACTER FROM THE
2020                                                            ;RECEIVER BUFFER + REST OF CONTENTS
2021  007404  004767  004326            JSR     PC,DATCHK       ;GO TO COMPARE EXPECTED & RECEIVED
2022                                                            ;DATA
2023  007410  000737                    BR      1$              ;GO BACK TO ISSUE ANOTHER CHARACTER
2024  007412  104401  001252    RED03:  TYPE    ,$QUES          ;TYPE A QUESTION MARK(?)
2025  007416  000167  177456            JMP     PRG4A           ;REITERATE THE XDBR QUESTION TO USER
2026  007422  104401  001252    RED03A: TYPE,   $QUES           ;TYPE '?' INDICATING USER TYPED
2027                                                            ;SOMETHING WRONG FOR CHARACTER LENGTH
2028  007426  000167  177612            JMP     PRG4C           ;GO BACK TO REISSUE QUESTION
2029
2030                                    ;THIS IS PROGRAM #5
2031                                    ;THE FOLLOWING USER UTILITY PROGRAM WILL ALLOW USER PARAMETERS
2032                                    ;FOR RUNNING A BINARY COUNT IN MAINTENANCE MODE
2033                                    ;
2034
2035  007432  012706  001100    PRG5:   MOV     #STACK,SP       ;INITIALIZE THE STACK POINTER
2036  007436  012737  013052  000034    MOV     #$TRAP,@#TRAPVEC        ;TRAP VECTOR FOR TRAP CALLS
2037  007444  012737  000340  000036    MOV     #340,@#TRAPVEC+2        ;LEVEL 7
2038  007452  012737  010732  000030    MOV     #$ERROR,@#EMTVEC        ;EMT VECTOR FOR ERROR ROUTINE
2039  007460  012737  000340  000032    MOV     #340,@#EMTVEC+2         ;LEVEL 7
2040  007466  104401  017040            TYPE    ,PRG5M          ;INDICATE THAT USER SELECTED
2041                                                            ;PROGRAM #5
2042  007472  104401  020306    PRG5A:  TYPE    ,LINTAD         ;ASK USER FOR THE TRANSMITTER DATA
2043                                                            ;BUFFER ADDRESS OF THE DEVICE
2044                                                            ;HE WISHES TO TEST
2045  007476  104410                    RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
2046                                                            ;AND STORE ON TOP OF STACK
2047                                    ;CHECK TO SEE IF THE USER RESPONSE WAS WITHIN LIMITS
2048  007500  012602                    MOV     (SP)+,R2        ;GET THE ANSWER TYPED
2049  007502  020227  176176            CMP     R2,#176176      ;IS THE NUMBER TOO HIGH?
```

```
2050  007506  101152                          BHI     REDO4           ;IF YES - GO TO RETRY SITUATION
2051  007510  020227  175616                  CMP     R2,#175616      ;IS THE NUMBER TOO LOW?
2052  007514  103547                          BLO     REDO4           ;IF YES - GO TO RETRY SITUATION
2053  007516  132702  000001                  BITB    #BIT0,R2        ;NUMBER IS IN RANGE BUT IS IT
2054                                                                  ;ON AN EVEN BOUNDARY?
2055  007522  001144                          BNE     REDO4           ;IF NOT - GO TO RETRY SITUATION
2056                                ;CHECK TO SEE IF USER RESPONSE WAS TRULY A XMIT BUFFER REGISTER
2057  007524  010203                          MOV     R2,R3           ;GET THE USER RESPONSE
2058  007526  142703  000370                  BICB    #370,R3 ;MASK OFF LOWER BYTE EXCEPT FOR
2059                                                                  ;LEAST SIGNIFICANT DIGIT
2060  007532  122703  000006                  CMPB    #6,R3           ;WAS THE LEAST SIGNIFICANT DIGIT OF THE
2061                                                                  ;USER RESPONSE EQUAL TO A SIX?
2062  007536  001136                          BNE     REDO4           ;BRANCH IF NOT
2063  007540  010267  171436                  MOV     R2,$TMP0        ;THE TRANMITTER ADDRESS TYPED
2064                                                                  ;IS OK - STORE FOR FUTURE USE
2065                                ;NOW CHECK TO MAKE SURE THE DEVICE IS PRESENT
2066  007544  016746  170234                  MOV     ERRVEC,-(SP)    ;SAVE THE TIMOUT VECTOR
2067  007550  012767  007562  170226          MOV     #2$,ERRVEC      ;SET UP TIMEOUT SERVICE ADDRESS
2068  007556  005712                          TST     (R2)            ;IF PRESENT WE WILL EXECUTE THE
2069                                                                  ;NEXT INSTRUCTION - IF NOT WE
2070                                                                  ;GO TO 2$:
2071  007560  000412                          BR      4$              ;BRANCH IF PRESENT
2072  007562  004767  004356          2$:     JSR     PC,SUERT2       ;GO SETUP FOR ERROR INFORMATION
2073  007566  012767  007576  171450          MOV     #3$,$ESCAPE     ;POINT OF RETURN AFTER ERROR REPORT
2074  007574  104006                          ERROR   +6              ;XDBR REFERENCE CAUSED TIMEOUT
2075  007576  022626                  3$:     CMP     (SP)+,(SP)+     ;CLEAN STACK FROM TIMEOUT
2076  007600  012667  170200                  MOV     (SP)+,ERRVEC    ;RESTORE TIMEOUT VECTOR
2077  007604  000513                          BR      REDO4           ;GO TO RETRY SITUATION
2078  007606  012667  170172          4$:     MOV     (SP)+,ERRVEC    ;DEVICE REGISTER IS PRESENT!
2079                                                                  ;ASK THE USER IF HE DESIRES SOME
2080                                                                  ;RANDOM NO. OF MSEC. WAIT TIME
2081                                                                  ;BEFORE CHECKING XCSR DONE FLAG
2082  007612  104401  017217          PRG5C:  TYPE,   LENGTH  ;ASK USER FOR THE CHARACTER LENGTH
2083                                                                  ;FOR WHICH HIS DEVICE IS SET
2084  007616  104411                          RDDEC                   ;ACCEPT THE ANSWER TYPED BY USER
2085                                ;CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
2086  007620  012600                          MOV     (SP)+,R0        ;GET THE ANSWER TYPED
2087  007622  020027  000010                  CMP     R0,#8.          ;IS THE NUMBER TOO HIGH?
2088  007626  101106                          BHI     REDO4A          ;IF YES - GO TO RETRY SITUATION
2089  007630  020027  000005                  CMP     R0,#5.          ;IS THE NUMBER TOO LOW?
2090  007634  103503                          BLO     REDO4A          ;IF YES - GO TO RETRY SITUATION
2091  007636  010067  171372                  MOV     R0,$TMP15       ;THE VALUE TYPED IS OK
2092                                                                  ;STORE FOR FUTURE USE
2093  007642  104401  020646                  TYPE    ,RSTALL         ;RANDOM NO. OF MSEC. WAIT TIME
2094  007646  104410                          RDOCT                   ;ACCEPT THE ANSWER TYPED BY USER
2095                                                                  ;AND STORE ON TOP OF STACK
2096  007650  012667  171332                  MOV     (SP)+,$TMP2     ;GET THE ANSWER TYPED
2097                                ;WE ARE NOW READY TO INITIALIZE THE BINARY COUNT AND GET
2098                                ;THE BINARY CHARACTER
2099                                ;
2100  007654  012767  177777  171342          MOV     #-1,$TMP11      ;SET LEAD IN VARIABLE TO -1
2101  007662  016767  171336  171336  PRG5B:  MOV     $TMP11,$TMP12   ;STORE PREVIOUS BINARY CHARACTER
2102  007670  005267  171332                  INC     $TMP12          ;FLIP BINARY CHARACTER AGAIN
2103  007674  042767  177400  171324          BIC     #177400,$TMP12  ;MASK TO 8 BITS
2104  007702  016767  171320  171314          MOV     $TMP12,$TMP11   ;STORE BINARY CHARACTER
2105  007710  016767  171312  171266          MOV     $TMP12,$TMP1    ;STORE BINARY CHARACTER
```

```
2106
2107  007716  016767  171260  171264          MOV     $TMP0,$TMP3     ;FOR FUTURE USE
2108  007724  162767  000002  171256          SUB     #2,$TMP3        ;GET THE XDBR ADDRESS
2109  007732  005767  171250          1$:     TST     $TMP2           ;FORM THE XCSR ADDRESS
2110  007736  001402                          BEQ     2$              ;DO WE RANDOM STALL?
2111  007740  004767  003624                  JSR     PC,STALL        ;BRANCH IF IT WASN'T DESIRED
2112  007744  004767  003730          2$:     JSR     PC,TIMETX       ;GO STALL RANDOM VALUE OF MSEC.
2113                                                                  ;GO WAIT FOR TRANSMITTER DONE
2114  007750  104401  017104                  TYPE    ,XDB            ;BIT TO SET
2115  007754  104000                          ERROR   +0              ;TYPE TRANSMITTER DONE BIT MESSAGE
2116  007756  052777  000004  171224          BIS     #BIT2,@$TMP3    ;XCSR DONE BIT NEVER SET
2117                                                                  ;SET THE MAINTENANCE BIT IN THE
2118  007764  016777  171214  171210          MOV     $TMP1,@$TMP0    ;TRANSMITTER CONTROL STATUS REGISTER
2119                                                                  ;LOAD TRANSMITTER DATA BUFFER
2120  007772  004767  003664                  JSR     PC,TIMERX       ;WITH SELECTED CHARACTER
2121  007776  104401  017153                  TYPE    ,RDB            ;GO WAIT FOR RECEIVER DONE BIT TO SET
2122  010002  104000                          ERROR   +0              ;TYPE RECEIVER DONE BIT MESSAGE
2123  010004  016767  171200  171200          MOV     $TMP3,$TMP4     ;RCSR DONE BIT NEVER SET
2124                                                                  ;GET THE TRANSMITTER CONTROL
2125  010012  162767  000002  171172          SUB     #2,$TMP4        ;STATUS REGISTER ADDRESS
2126                                                                  ;FORM THE RECEIVER DATA BUFFER
2127  010020  017767  171166  171166          MOV     @$TMP4,$TMP5    ;ADDRESS
2128                                                                  ;STORE THE CHARACTER FROM THE
2129  010026  004767  003704                  JSR     PC,DATCHK       ;RECEIVER BUFFER + REST OF CONTENTS
2130                                                                  ;GO TO COMPARE EXPECTED &  RECEIVED
2131  010032  000713                          BR      PRG5B           ;DATA
2132                                                                  ;GO BACK TO ISSUE ANOTHER BINARY
2133  010034  104401  001252          RED04:  TYPE    ,$QUES          ;CHARACTER
2134  010040  000167  177426                  JMP     PRG5A           ;TYPE A QUESTION MARK(?)
2135  010044  104401  001252          RED04A: TYPE,   $QUES           ;GO BACK TO REITERATE XDBR QUESTION
2136                                                                  ;TYPE '?' INDICATING USER TYPED
2137  010050  000167  177536                  JMP     PRG5C           ;SOMETHING WRONG FOR CHARACTER LENGTH
2138                                                                  ;GO BACK TO REISSUE QUESTION
2139
2140                                  ;THIS ROUTINE WILL SET UP:
2141                                  ;       RCSR - RECEVER STATUS REGISTER
2142                                  ;       RBUF - RECEIVER BUFFER REGISTER
2143                                  ;       XCSR - TRANSMITTER STATUS REGISTER
2144                                  ;       XBUF - TRANSMITTER BUFFER REGISTER
2145                                  ;INITIALLY, IN RESPONSE TO USER REPLY TO 1ST DEVICE HE WANTS
2146                                  ;TESTED, AND THEREAFTER, AT THE END OF A PROGRAM TO CYCLE THRU
2147  010054  016767  171200  171326  DLADDR: MOV     DLBASE,DLRCSR   ;ALL DEVICES FOR MULTIPLE DEVICE TESTING (IF REQUIRED)
2148                                                                  ;STORE RECEIVER STATUS REGISTER
2149  010062  062767  000002  171170          ADD     #2,DLBASE       ;OF CURRENT DEVICE
2150                                                                  ;FORM RECEIVER BUFFER REGISTER
2151  010070  016767  171164  171314          MOV     DLBASE,DLRDBR   ;OF CURRENT DEVICE
2152                                                                  ;STORE RECEIVER BUFFER REGISTER
2153  010076  062767  000002  171154          ADD     #2,DLBASE       ;OF CURRENT DEVICE
2154                                                                  ;FORM TRANSMITTER STATUS REGISTER
2155  010104  016767  171150  171302          MOV     DLBASE,DLXCSR   ;OF OF CURRENT DEVICE
2156                                                                  ;STORE TRANSMITTER STATUS REGISTER
2157  010112  062767  000002  171140          ADD     #2,DLBASE       ;OF CURRENT DEVICE
2158                                                                  ;FOR TRANSMITTER BUFFER REGISTER
2159  010120  016767  171134  171270          MOV     DLBASE,DLXDBR   ;OF CURRENT DEVICE
2160                                                                  ;STORE TRANSMITTER BUFFER REGISTER
2161  010126  000207                          RTS     PC              ;OF CURRENT DEVICE
                                                                      ;RETURN
```

```
                                .SBTTL  END OF PASS ROUTINE

                                ;;*****************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO RESTRT

010130                          $EOP:
                                ;THIS NEXT SECTION UP TO THE NEXT LINE OF ASTERISKS WAS
                                ;SUPPLIED BY THE MACRO 'EOPBEG'.  THE MACRO NAME APPEARS IN
                                ;THE SOURCE PROGRAM AS ONE OF THE ARGUMENTS TO THE .$EOP
                                ;SYSMAC UTILITY ROUTINE CALL.
010130  000004                        SCOPE
010132  105767  171134                TSTB    MULTD           ;ARE WE RUNNING MULTIPLE DEVICES?
010136  001501                        BEQ     3$              ;BRANCH IF NOT FOR NORMAL
                                                                ;'END PASS # XX' TYPEOUT
010140  005767  171130                TST     ACTREG          ;ARE ANY DEVICES ACTIVE?
010144  001011                        BNE     1$              ;BRANCH IF YES
010146  104401  020074                TYPE    ,FOULUP         ;INDICATE SOMETHING WRONG!
                                                                ;MULTIPLE DEVICES ARE BEING
                                                                ;RUN SUPPOSEDLY, BUT NONE ARE
                                                                ;SHOWN ACTIVE
010152  000000                        HALT                    ;WAIT FOR A USER RESPONSE
010154  005067  171112                CLR     MULTD           ;CLEAR MULTIPLE DEVICE FLAG
010160  005067  171072                CLR     TABFLG          ;CLEAR TABLE CREATION FLAG
010164  000167  171602                JMP     RESTRT          ;GET READY TO START ALL OVER
                                                                ;AGAIN - ALL DEVICES WERE
                                                                ;DESELECTED SOMEHOW!
010170  062767  000010  171066  1$:   ADD     #10,BASEADD     ;FORM A NEW BASE
                                                                ;ADDRESS FOR START OF NEXT BLOCK
                                                                ;OF REGISTERS FOR NEXT DEVICE
010176  062767  000010  171064        ADD     #10,BASEIV      ;FORM A NEW BASE ADDRESS FOR
                                                                ;START OF NEXT BLOCK OF VECTORS
                                                                ;FOR NEXT DEVICE
010204  000241                        CLC                     ;CLEAR LAST DEVICE INDICATOR
010206  006167  171064                ROL     ROTADD          ;UPDATE NEXT POSSIBLE DEVICE ACTIVE
                                                                ;POINTER
010212  103431                        BCS     2$              ;BRANCH IF THIS WAS THE
                                                                ;LAST DEVICE TO BE TESTED ON
                                                                ;THIS PASS
010214  036767  171056  171052        BIT     ROTADD,ACTREG   ;IS THIS DEVICE TRULY ACTIVE
                                                                ;(AS PER USER)
010222  001762                        BEQ     1$              ;BRANCH IF NOT TO SEE IF NEXT
                                                                ;ONE POSSIBLE IS
010224  016767  171034  171026        MOV     BASEADD,DLBASE  ;FORM THE RECEIVER STATUS REGISTER
                                                                ;ADDRESS OF NEXT DEVICE
010232  016767  171032  171160        MOV     BASEIV,DLVECT   ;GET NEXT DEVICE RCVR VECTOR
010240  000240                        NOP
010242  004767  177606                JSR     PC,DLADDR       ;GO FORM DL ADDRESSES FOR NEXT
                                                                ;DEVICE SELECTED
010246  005067  170630                CLR     $TSTNM          ;INITIALIZE TEST NO. FOR A PROGRAM
                                                                ;PASS OVER THE NEXT DEVICE ACTIVE
010252  005077  171136                CLR     @DLXCSR         ;CLEAR OUT BOTH CSR'S
010256  005077  171126                CLR     @DLRCSR
```

```
2218  010262  005777  171124                TST     @DLRDBR         ;FLUSH RCVR "DONE" BIT
2219  010266  005777  171120                TST     @DLRDBR
2220  010272  000167  172472                JMP     TST1            ;START TESTING THIS DEVICE
2221  010276                          2$:
2222                                  ;IF WE TAKE THIS PATH WE HAVE MADE A CYCLE THRU THE PROGRAM ONCE
2223                                  ;PER DEVICE I.E. - WE HAVE MADE A COMPLETE PASS
2224                                  ;NOW WE NEED TO RESTORE EVERYTHING FOR THE NEXT COMPLETE PASS
2225  010276  012767  000001  170772        MOV     #1,ROTADD       ;SET UP ROTATING POINTER FOR NEXT
2226                                                                  ;MULTIPLE PASS
2227  010304  016767  170752  170752        MOV     KEEPADD,BASEADD ;RESTORE BASE ADDRESS
2228  010312  016767  170750  170750        MOV     KEEPIV,BASEIV   ;RESTORE BASE INTERRUPT VECTOR
2229  010320  016767  170740  170732        MOV     BASEADD,DLBASE  ;RESTORE 1ST DEVICE BASE ADDRESS
2230  010326  016767  170736  171064        MOV     BASEIV,DLVECT   ;RESTORE 1ST DEVICE VECTOR ADDRESS
2231  010334  000240                        NOP
2232  010336  004767  177512                JSR     PC,DLADDR       ;FORM ADDRESSES FOR 1ST DEVICE
2233  010342                          3$:
2234                                  ;;**************************************************************
2235  010342  005067  170534                CLR     $TSTNM          ;;ZERO THE TEST NUMBER
2236  010346  005067  170670                CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
2237  010352  005267  170522                INC     $PASS           ;;INCREMENT THE PASS NUMBER
2238  010356  042767  100000  170514        BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
2239  010364  [illegible]                   DEC     (PC)+           ;;LOOP?
2240  010366  000001                  $EOPCT: .WORD   1
2241  010370  003022                          BGT     $DOAGN          ;;YES
2242  010372  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
2243  010374  000001                  $ENDCT: .WORD   1
2244  010376  010366                          $EOPCT
2245  010400  104401  010445                TYPE    ,$ENDMG         ;;TYPE "END PASS #"
2246  010404  016746  170470                MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
2247  010410  104405                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
2248  010412  104401  010442                TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
2249  010416  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
2250  010422  001405                        BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
2251  010424  000005                        RESET                   ;;CLEAR THE WORLD
2252  010426  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
2253  010430  000240                        NOP                     ;;SAVE ROOM
2254  010432  000240                        NOP                     ;;FOR
2255  010434  000240                        NOP                     ;;ACT11
2256  010436                          $DOAGN:
2257  010436  000137                        JMP     @(PC)+          ;;RETURN
2258  010440  001772                  $RTNAD: .WORD   RESTRT
2259  010442     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
2260  010445     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
2261  010452  050040  051501  020123
2262  010460  000043
2263                                  .SBTTL  SCOPE HANDLER ROUTINE
2264
2265                                  ;;**************************************************************
2266                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
2267                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
2268                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
2269                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2270                                  ;*SW14=1         LOOP ON TEST
2271                                  ;*SW11=1         INHIBIT ITERATIONS
2272                                  ;*SW09=1         LOOP ON ERROR
2273                                  ;*SW08=1         LOOP ON TEST IN SWR<7:0>
```

```
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT
                                        $SCOPE:
010462
010462  032777  040000  170450  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
010470  001111                          BNE     $OVER           ;;YES IF SW14=1
                                        ;*****START OF CODE FOR THE XOR TESTER*****
010472  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
010474  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
010500  012737  010520  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
010506  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
010512  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
010516  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
010520  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
010522  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
010526  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
010530                          6$:;*****END OF CODE FOR THE XOR TESTER*****
010530  032777  000400  170402          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
010536  001404                          BEQ     2$              ;;BR IF NO
010540  127767  170374  170334          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
010546  001462                          BEQ     $OVER           ;;BR IF YES
010550  105767  170327          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
010554  001421                          BEQ     3$              ;;BR IF NO
010556  126767  170333  170317          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
010564  101015                          BHI     3$              ;;BR IF NO
010566  032777  001000  170344          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
010574  001404                          BEQ     4$              ;;BR IF NO
010576  016767  170306  170302  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
010604  000443                          BR      $OVER
010606  105067  170271          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
010612  005067  170424                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
010616  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
010620  032777  004000  170312  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
010626  001011                          BNE     1$              ;;BR IF YES
010630  005767  170244                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
010634  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
010636  005267  170242                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
010642  026767  170374  170234          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
010650  002021                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
010652  012767  000001  170224  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
010660  016767  000044  170354          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
010666  105267  170210          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
010672  011667  170210                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
010676  011667  170206                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
010702  005067  170336                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
010706  112767  000001  170201          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
010714  016777  170162  170220  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
010722  016716  170160                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
010726  000002                          RTI                     ;;FIXES PS
010730  000100                  $MXCNT: 100                     ;;MAX. NUMBER OF ITERATIONS

                                        .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
                                        ;*SW10=1         BELL ON ERROR
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*CALL
                                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

010732                                  $ERROR:
010732  105267  170145                  7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
010736  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
010740  016777  170136  170174          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
010746  032777  002000  170164          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
010754  001402                          BEQ     1$              ;;NO - SKIP
010756  104401  001246                  TYPE    ,$BELL          ;;RING BELL
010762  005267  170124                  1$:     INC     $ERRTL          ;;COUNT THE NUMBER OF ERRORS
010766  011667  170124                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
010772  162767  000002  170116          SUB     #2,$ERRPC
011000  117767  170112  170106          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
011006  032777  020000  170124          BIT     #BIT13,@SWR     ;SKIP TYPEOUT IF SET
011014  001004                          BNE     20$             ;;SKIP TYPEOUTS
011016  004767  000044                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
011022  104401  001253                  TYPE    ,$CRLF
011026                                  20$:
011026  005777  170106                  2$:     TST     @SWR            ;;HALT ON ERROR
011032  100001                          BPL     3$              ;;SKIP IF CONTINUE
011034  000000                          HALT                    ;;HALT ON ERROR!
011036  032777  001000  170074          3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
011044  001402                          BEQ     4$              ;;BR IF NO
011046  016716  170036                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
011052  005767  170166                  4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
011056  001402                          BEQ     5$              ;;BR IF NONE
011060  016716  170160                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
011064                                  5$:
011064  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.


011066                                  $ERRTYP:
011066  104401  001253                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
011072  010046                          MOV     R0,-(SP)        ;;SAVE R0
011074  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
011076  153700  001114                  BISB    @#$ITEMB,R0
011102  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                ;;TYPE THE PC OF THE ERROR
011104  016746  170006                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;ERROR ADDRESS
011110  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
011112  000426                          BR      6$              ;;GET OUT
```

```
011114  005300          1$:   DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
011116  006300                ASL     R0              ;;      WORK FOR THE ERROR TABLE
011120  006300                ASL     R0
011122  006300                ASL     R0
011124  062700  001310        ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
011130  012067  000004        MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
011134  001404                BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
011136  104401                TYPE                    ;;TYPE THE "ERROR MESSAGE"
011140  000000          2$:   .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
011142  104401  001253        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
011146  012067  000004  3$:   MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
011152  001404                BEQ     5$              ;;SKIP TYPEOUT IF 0
011154  104401                TYPE                    ;;TYPE THE "DATA HEADER"
011156  000000          4$:   .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
011160  104401  001253        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
011164  011000          5$:   MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
011166  001004                BNE     7$              ;;GO TYPE THE DATA
011170  012600          6$:   MOV     (SP)+,R0        ;;RESTORE R0
011172  104401  001253        TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
011176  000207                RTS     PC              ;;RETURN
011200                  7$:
011200  013046                MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
011202  104402                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
011204  005710                TST     (R0)            ;;IS THERE ANOTHER NUMBER?
011206  001770                BEQ     6$              ;;BR IF NO
011210  104401  011216        TYPE    ,8$             ;;TYPE TWO(2) SPACES
011214  000771                BR      7$              ;;LOOP
011216  020040      000 8$:   .ASCIZ  /  /            ;;TWO(2) SPACES
011222                        .EVEN
```

```
                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

011222  017646  000000  $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
```

```
2442  011226  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
2443  011234  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
2444  011240  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
2445  011244  000406                          BR      $TYPON
2446  011246  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
2447  011254  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
2448  011262  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
2449  011270  010346                          MOV     R3,-(SP)        ;;SAVE R3
2450  011272  010446                          MOV     R4,-(SP)        ;;SAVE R4
2451  011274  010546                          MOV     R5,-(SP)        ;;SAVE R5
2452  011276  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
2453  011302  005404                          NEG     R4
2454  011304  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
2455  011310  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
2456  011314  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2457  011320  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
2458  011324  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
2459  011326  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2460  011330  000404                          BR      3$              ;;GO DO MSB
2461  011332  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
2462  011334  006105                          ROL     R5
2463  011336  006105                          ROL     R5
2464  011340  010503                          MOV     R5,R3
2465  011342  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2466  011344  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
2467  011350  100016                          BPL     7$              ;;BR IF NO
2468  011352  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
2469  011356  001002                          BNE     4$              ;;TEST FOR 0
2470  011360  005704                          TST     R4              ;;SUPPRESS THIS 0?
2471  011362  001403                          BEQ     5$              ;;BR IF YES
2472  011364  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2473  011366  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2474  011372  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2475  011376  110367  000040                  MOVB    R3,8$           ;;SAVE FOR TYPING
2476  011402  104401  011442                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
2477  011406  105367  000032          7$:     DECB    $OCNT           ;;COUNT BY 1
2478  011412  003347                          BGT     2$              ;;BR IF MORE TO DO
2479  011414  002402                          BLT     6$              ;;BR IF DONE
2480  011416  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2481  011420  000744                          BR      2$              ;;GO DO THE LAST DIGIT
2482  011422  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
2483  011424  012604                          MOV     (SP)+,R4        ;;RESTORE R4
2484  011426  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2485  011430  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2486  011436  012616                          MOV     (SP)+,(SP)
2487  011440  000002                          RTI                     ;;RETURN
2488  011442  000                     8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2489  011443  000                             .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2490  011444  000                     $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2491  011445  000                     $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2492  011446  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2493
2494                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
2495
2496                                  ;;**************************************************************
2497                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
```

```
                                        ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                        ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                        ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                        ;*REPLACED WITH SPACES.
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                        ;*      TYPDS                   ;;GO TO THE ROUTINE
                                        $TYPDS:
011450                                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011450  010046                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
011452  010146                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
011454  010246                                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
011456  010346                                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
011460  010546                                 MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
011462  012746  020200                        MOV     20(SP),R5       ;;GET THE INPUT NUMBER
011466  016605  000020                        BPL     1$              ;;BR IF INPUT IS POS.
011472  100004                                NEG     R5              ;;MAKE THE BINARY NUMBER POS.
011474  005405                                MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
011476  112766  000055  000001                CLR     R0              ;;ZERO THE CONSTANTS INDEX
011504  005000                        1$:     MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
011506  012703  011664                        MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
011512  112723  000040                        CLR     R2              ;;CLEAR THE BCD NUMBER
011516  005002                        2$:     MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
011520  016001  011654                        SUB     R1,R5           ;;FORM THIS BCD DIGIT
011524  160105                        3$:     BLT     4$              ;;BR IF DONE
011526  002402                                INC     R2              ;;INCREASE THE BCD DIGIT BY 1
011530  005202                                BR      3$
011532  000774                        4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
011534  060105                                TST     R2              ;;CHECK IF BCD DIGIT=0
011536  005702                                BNE     5$              ;;FALL THROUGH IF 0
011540  001002                                TSTB    (SP)            ;;STILL DOING LEADING 0'S?
011542  105716                                BMI     7$              ;;BR IF YES
011544  100407                        5$:     ASLB    (SP)            ;;MSD?
011546  106316                                BCC     6$              ;;BR IF NO
011550  103003                                MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
011552  116663  000001  177777                BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
011560  052702  000060                6$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
011564  052702  000040                7$:     MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
011570  110223                                TST     (R0)+           ;;JUST INCREMENTING
011572  005720                                CMP     R0,#10          ;;CHECK THE TABLE INDEX
011574  020027  000010                        BLT     2$              ;;GO DO THE NEXT DIGIT
011600  002746                                BGT     8$              ;;GO TO EXIT
011602  003002                                MOV     R5,R2           ;;GET THE LSD
011604  010502                                BR      6$              ;;GO CHANGE TO ASCII
011606  000764                        8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
011610  105726                                BPL     9$              ;;BR IF NO
011612  100003                                MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
011614  116663  177777  177776                CLRB    (R3)            ;;SET THE TERMINATOR
011622  105013                        9$:     MOV     (SP)+,R5        ;;POP STACK INTO R5
011624  012605                                MOV     (SP)+,R3        ;;POP STACK INTO R3
011626  012603                                MOV     (SP)+,R2        ;;POP STACK INTO R2
011630  012602                                MOV     (SP)+,R1        ;;POP STACK INTO R1
011632  012601                                MOV     (SP)+,R0        ;;POP STACK INTO R0
011634  012600                                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
011636  104401  011664                        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
011642  016666  000002  000004
```

```
2554  011650  012616                          MOV     (SP)+,(SP)
2555  011652  000002                          RTI                             ;;RETURN TO USER
2556  011654  023420                $DTBL:    10000.
2557  011656  001750                          1000.
2558  011660  000144                          100.
2559  011662  000012                          10.
2560  011664  000004                $DBLK:    .BLKW   4
2561
2562                                  .SBTTL  TTY INPUT ROUTINE
2563
2564                                  ;;**************************************************************
2565                                  .ENABL  LSB
2566
2567                                  .DSABL  LSB
2568
2569
2570                                  ;;**************************************************************
2571                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2572                                  ;*CALL:
2573                                  ;*      RDCHR                           ;;INPUT A SINGLE CHARACTER FROM THE TTY
2574                                  ;*      RETURN HERE                     ;;CHARACTER IS ON THE STACK
2575                                  ;*                                      ;;WITH PARITY BIT STRIPPED OFF
2576                                  ;
2577
2578  011674  011646                $RDCHR: MOV     (SP),-(SP)              ;;PUSH DOWN THE PC
2579  011676  016666  000004  000002        MOV     4(SP),2(SP)             ;;SAVE THE PS
2580  011704  105777  167234        1$:     TSTB    @$TKS                   ;;WAIT FOR
2581  011710  100375                        BPL     1$                      ;;A CHARACTER
2582  011712  117766  167230  000004        MOVB    @$TKB,4(SP)             ;;READ THE TTY
2583  011720  042766  177600  000004        BIC     #^C<177>,4(SP)          ;;GET RID OF JUNK IF ANY
2584  011726  026627  000004  000023        CMP     4(SP),#23               ;;IS IT A CONTROL-S?
2585  011734  001013                        BNE     3$                      ;;BRANCH IF NO
2586  011736  105777  167202        2$:     TSTB    @$TKS                   ;;WAIT FOR A CHARACTER
2587  011742  100375                        BPL     2$                      ;;LOOP UNTIL ITS THERE
2588  011744  117746  167176                MOVB    @$TKB,-(SP)             ;;GET CHARACTER
2589  011750  042716  177600                BIC     #^C177,(SP)             ;;MAKE IT 7-BIT ASCII
2590  011754  022627  000021                CMP     (SP)+,#21               ;;IS IT A CONTROL-Q?
2591  011760  001366                        BNE     2$                      ;;IF NOT DISCARD IT
2592  011762  000750                        BR      1$                      ;;YES, RESUME
2593  011764  026627  000004  000140  3$:     CMP     4(SP),#140              ;;IS IT UPPER CASE?
2594  011772  002407                        BLT     4$                      ;;BRANCH IF YES
2595  011774  026627  000004  000175        CMP     4(SP),#175              ;;IS IT A SPECIAL CHAR?
2596  012002  003003                        BGT     4$                      ;;BRANCH IF YES
2597  012004  042766  000040  000004        BIC     #40,4(SP)               ;;MAKE IT UPPER CASE
2598  012012  000002                4$:     RTI                             ;;GO BACK TO USER
2599                                  ;;**************************************************************
2600                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2601                                  ;*CALL:
2602                                  ;*      RDLIN                           ;;INPUT A STRING FROM THE TTY
2603                                  ;*      RETURN HERE                     ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2604                                  ;*                                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2605
2606  012014  010346                $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
2607  012016  005046                        CLR     -(SP)                   ;;CLEAR THE RUBOUT KEY
2608  012020  012703  012250        1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
2609  012024  022703  012260        2$:     CMP     #$TTYIN+8.,R3           ;;BUFFER FULL?
```

```
2610  012030  101456                          BLOS    4$              ;;BR IF YES
2611  012032  104406                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2612  012034  112313                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
2613  012036  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
2614  012042  001022                          BNE     5$              ;;BR IF NO
2615  012044  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
2616  012046  001007                          BNE     6$              ;;BR IF NO
2617  012050  112767  000134  000170         MOVB    #'\,9$          ;;TYPE A BACK SLASH
2618  012056  104401  012246                  TYPE    ,9$
2619  012062  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
2620  012066  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
2621  012070  020327  012250                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
2622  012074  103434                          BLO     4$              ;;BR IF YES
2623  012076  111367  000144                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
2624  012102  104401  012246                  TYPE    ,9$             ;;GO TYPE
2625  012106  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
2626  012110  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
2627  012112  001406                          BEQ     7$              ;;BR IF NO
2628  012114  112767  000134  000124         MOVB    #'\,9$          ;;TYPE A BACK SLASH
2629  012122  104401  012246                  TYPE    ,9$
2630  012126  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
2631  012130  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
2632  012134  001003                          BNE     8$              ;;BR IF NO
2633  012136  104401  012260                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
2634  012142  000726                          BR      1$              ;;GO START OVER
2635  012144  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
2636  012150  001011                          BNE     3$              ;;BRANCH IF NO
2637  012152  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
2638  012154  104401  001253                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
2639  012160  104401  012250                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
2640  012164  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
2641  012166  104401  001252          4$:     TYPE    ,$QUES          ;;TYPE A '?'
2642  012172  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
2643  012174  111367  000046          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2644  012200  104401  012246                  TYPE    ,9$
2645  012204  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2646  012210  001305                          BNE     2$              ;;LOOP IF NOT RETURN
2647  012212  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2648  012216  104401  001254                  TYPE    ,$LF            ;;TYPE A LINE FEED
2649  012222  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
2650  012224  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2651  012226  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2652  012230  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2653  012236  012766  012250  000004          MOV     #$TTYIN,4(SP)
2654  012244  000002                          RTI                     ;;RETURN
2655  012246     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2656  012247     000                          .BYTE   0               ;;TERMINATOR
2657  012250  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
2658  012260  052536  005015     000  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
2659  012265     136  006507  000012  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
2660  012272  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
2661  012300  020075     000
2662  012303     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
2663  012310  036440  000040
2664
2665                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
```

```
                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
                                ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
                                ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
                                ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
                                ;*CALL:
                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

012314  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
012316  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
012324  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
012326  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
012330  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
012332  104407          1$:     RDLIN                   ;;READ AN ASCIZ LINE
012334  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
012336  010067  000100          MOV     R0,5$           ;;AND SAVE IT
012342  005001                  CLR     R1              ;;CLEAR DATA WORD
012344  005002                  CLR     R2
012346  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
012350  001420                  BEQ     3$              ;;IF ZERO GET OUT
012352  122716  000060          CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
012356  003026                  BGT     4$              ;;IS AN OCTAL DIGIT
012360  122716  000067          CMPB    #'7,(SP)
012364  002423                  BLT     4$
012366  006301                  ASL     R1              ;;*2
012370  006102                  ROL     R2
012372  006301                  ASL     R1              ;;*4
012374  006102                  ROL     R2
012376  006301                  ASL     R1              ;;*8
012400  006102                  ROL     R2
012402  042716  177770          BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
012406  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
012410  000756                  BR      2$              ;;LOOP
012412  005726          3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
012414  010166  000012          MOV     R1,12(SP)       ;;SAVE THE RESULT
012420  010267  000026          MOV     R2,$HIOCT
012424  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
012426  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
012430  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
012432  000002                  RTI                     ;;RETURN
012434  005726          4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
012436  105010                  CLRB    (R0)            ;;SET A TERMINATOR
012440  104401                  TYPE                    ;;TYPE UP THRU THE BAD CHAR.
012442  000000          5$:     .WORD   0
012444  104401  001252          TYPE    $QUES           ;;"?" "CR" & "LF"
012450  000730                  BR      1$              ;;TRY AGAIN
012452  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE

                        .SBTTL  TYPE ROUTINE

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

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

012454  105767  166477          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMIN_?
012460  100002                          BPL     1$              ;;BR IF YES
012462  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
012464  000407                          BR      3$              ;;LEAVE
012466  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
012470  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
012474  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARAC ER TO BE TYPED ONTO STACK
012476  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
012500  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
012502  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
012504  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
012510  000002                          RTI                     ;;RETURN
012512  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
012516  001430                          BEQ     8$
012520  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
012524  001006                          BNE     5$
012526  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
012530  104401                          TYPE                    ;;TYPE A CR AND LF
012532  001253                          $CRLF
012534  105067  000130                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
012540  000755                          BR      2$              ;;GET NEXT CHARACTER
012542  004767  000056          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
012546  126726  166404          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
012552  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
012554  016746  166374                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
012560  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
012564  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
012566  004767  000032                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
012572  105367  000072                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
012576  000770                          BR      7$              ;;LOOP

                                        ;HORIZONTAL TAB PROCESSOR

012600  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
012604  004767  000014          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
012610  132767  000007  000052          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
012616  001372                          BNE     9$              ;;TAB STO?
012620  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
012622  000724                          BR      2$              ;;GET NEXT CHARACTER
012624  105777  166320          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
012630  100375                          BPL     $TYPEC
```

```
2778  012632  116677  000002  166312          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2779  012640  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2780  012646  001003                          BNE     1$              ;;BRANCH IF NO
2781  012650  105067  000014                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2782  012654  000406                          BR      $TYPEX          ;;EXIT
2783  012656  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2784  012664  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
2785  012666  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
2786  012670  000000                $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2787  012672  000207                $TYPEX: RTS     PC
2788
2789
2790                                  .SBTTL  READ A DECIMAL NUMBER FROM THE TTY
2791
2792                                  ;;*******************************   *******************************
2793                                  ;*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
2794                                  ;*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
2795                                  ;*ARE READ A "?" FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
2796                                  ;*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
2797                                  ;*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
2798                                  ;*POSITIVE 32767 TO NEGATIVE 32768.
2799                                  ;*CALL:
2800                                  ;*      RDDEC                   ;;READ A DECIMAL NUMBER
2801                                  ;*      RETURN HERE             ;;NUMBER IS ON TOP OF THE STACK
2802                                  ;
2803
2804  012674  011646                $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
2805  012676  016666  000004  000002        MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
2806  012704  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2807  012706  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2808  012710  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2809  012712  104407                1$:     RDLIN                   ;;READ AN ASCIZ LINE
2810  012714  012600                        MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
2811  012716  010067  000120                MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
2812  012722  005046                        CLR     -(SP)           ;;CLEAR DATA WORD
2813  012724  005002                        CLR     R2              ;;SIGN SET POSITIVE
2814  012726  122710  000055                CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
2815  012732  001001                        BNE     2$              ;;BR IF NO MINUS SIGN
2816  012734  112002                        MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
2817  012736  112001                2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
2818  012740  001424                        BEQ     3$              ;;GET OUT IF ZERO
2819  012742  122701  000060                CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
2820  012746  003032                        BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
2821  012750  122701  000071                CMPB    #'9,R1
2822  012754  002427                        BLT     5$
2823  012756  032716  170000                BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
2824  012762  001024                        BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
2825  012764  006316                        ASL     (SP)            ;;*2
2826  012766  011646                        MOV     (SP),-(SP)      ;;SAVE FOR LATER
2827  012770  006316                        ASL     (SP)            ;;*4
2828  012772  006316                        ASL     (SP)            ;;*8.
2829  012774  062616                        ADD     (SP)+,(SP)      ;;*10.
2830  012776  102416                        BVS     5$              ;;OVERFLOW ISN'T ALLOWED
2831  013000  162701  000060                SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
2832  013004  060116                        ADD     R1,(SP)         ;;ADD IN THIS DIGIT
2833  013006  102412                        BVS     5$              ;;OVERFLOW ISN'T ALLOWED
```

```
2834  013010  000752                        BR      2$              ;;LOOP
2835  013012  005702                3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
2836  013014  001401                        BEQ     4$              ;;BR IF NO
2837  013016  005416                        NEG     (SP)            ;;YES--NEGATE THE NUMBER
2838  013020  012666  000012        4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
2839  013024  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
2840  013026  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
2841  013030  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
2842  013032  000002                        RTI                     ;;RETURN
2843
2844  013034  005726                5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
2845  013036  105010                        CLRB    (R0)            ;;SET A TERMINATOR
2846  013040  104401                        TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
2847  013042  000000                6$:     .WORD   0               ;;POINTER GOES HERE
2848  013044  104401  001252                TYPE    ,$QUES          ;;"?" "CR" &"LF"
2849  013050  000720                        BR      1$              ;;TRY AGAIN
2850
2851                                .SBTTL  TRAP DECODER
2852
2853                                ;;**************************************************************
2854                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2855                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2856                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2857                                ;*GO TO THAT ROUTINE.
2858
2859  013052  010046                $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2860  013054  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
2861  013060  005740                        TST     -(R0)           ;;BACKUP BY 2
2862  013062  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2863  013064  006300                        ASL     R0              ;;POSITION FOR INDEXING
2864  013066  016000  013106                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2865  013072  000200                        RTS     R0              ;;GO TO ROUTINE
2866
2867
2868                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
2869
2870  013074  011646                $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
2871  013076  016666  000004  000002        MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
2872  013104  000002                        RTI                     ;;RESTORE THE PSW
2873
2874                                .SBTTL  TRAP TABLE
2875
2876                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2877                                ;*BY THE "TRAP" INSTRUCTION.
2878
2879                                ;       ROUTINE
2880                                        -------
2881  013106  013074                $TRPAD: .WORD   $TRAP2
2882  013110  012454                        $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
2883  013112  011246                        $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2884  013114  011222                        $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2885  013116  011262                        $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2886  013120  011450                        $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
2887
2888
2889  013122  011674                        $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
                                        $RDDEC  ;;CALL=RDDEC    TRAP+11(104411) READ A DECIMAL NUMBER FROM TTY

                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;**********************************************************************
                                ;POWER DOWN ROUTINE
013132  012737  013276  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
013140  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
013146  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
013150  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
013152  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
013154  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
013156  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
013160  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
013162  017746  165752                  MOV     @$WR,-(SP)      ;;PUSH @$WR ON STACK
013166  010667  000110                  MOV     SP,$SAVR6       ;;SAVE SP
013172  012737  013204  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
013200  000000                          HALT
013202  000776                          BR      .-2             ;;HANG UP

                                ;;**********************************************************************
                                ;POWER UP ROUTINE
013204  012737  013276  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
013212  016706  000064                  MOV     $SAVR6,SP       ;;GET SP
013216  005067  000060                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
013222  005267  000054          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
013226  001375                          BNE     1$              ;;OF  WORD
013230  012677  165704                  MOV     (SP)+,@$WR      ;;POP STACK INTO @$WR
013234  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
013236  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
013240  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
013242  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
013244  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
013246  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
013250  012737  013132  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
013256  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
013264  104401                          TYPE                    ;REPORT THE POWER FAILURE
013266  013304                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
013270  012716                          MOV     (PC)+,(SP)      ;;RESTART AT RESTRT
013272  001772                  $PWRAD: .WORD   RESTRT          ;;RESTART ADDRESS
013274  000002                          RTI
013276  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
013300  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
013302  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
013304  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
013312  000122                          .EVEN

                                ;;**********************************************************************
                                ;TRANSMIT INTERRUPT SERVICE ROUTINE FOR 256. BYTE BLOCK TRANSFERS
                                ;;**********************************************************************

013314  105777  166074          XINT:   TSTB    @DLXCSR         ;"READY" SET ??
013320  100416                          BMI     1$              ;BR IF YES
```

```
2946  013322  013767  177776  165652          MOV     @#PSW,$TMP0     ;SAVE THE ERROR PSW
2947  013330  010667  165642                  MOV     SP,$REG6        ;SAVE THE ERROR STACK POINTER
2948  013334  005167  166062                  COM     XFLG0           ;SET XMIT SOFTWARE ERROR FLAG
2949  013340  042777  000100  166042          BIC     #100,@DLRCSR    ;TURN OFF THE INTERRUPT ENABLES
2950  013346  042777  000100  166040          BIC     #100,@DLXCSR
2951  013354  000411                          BR      2$              ;GO TO EXIT
2952  013356  022767  021722  166046  1$:     CMP     #DLBUFI,OPTR    ;XMITTED 256. BYTES YET ??
2953  013364  001405                          BEQ     2$              ;BR IF YES
2954  013366  117777  166040  166022          MOVB    @OPTR,@DLXDBR   ;OUTPUT A BYTE
2955  013374  005267  166032                  INC     OPTR            ;UPDATE BUFFER POINTER
2956  013400  000002                  2$:     RTI                     ;RETURN TO MAINLINE TEST
2957
2958                                  ;;**************************************************************
2959                                  ;RECEIVER INTERRUPT SERVICE ROUTINE FOR 256. BYTE BLOCK TRANSFERS
2960                                  ;;**************************************************************
2961
2962  013402  105777  166002          RINT:   TSTB    @DLRCSR ;"DONE" SET ??
2963  013406  100410                          BMI     1$              ;BR IF YES
2964  013410  013767  177776  165564          MOV     @#PSW,$TMP0     ;SAVE THE ERROR PSW
2965  013416  010667  165554                  MOV     SP,$REG6        ;SAVR THE ERROR STACK POINTER
2966  013422  005167  165776                  COM     RFLG0           ;SET HARD RCVR ERROR FLAG
2967  013426  000415                          BR      2$              ;GO EXIT
2968  013430  005777  165756          1$:     TST     @DLRDBR         ;ANY SOFT ERRORS ??
2969  013434  100021                          BPL     3$              ;BR IF NOT
2970  013436  013767  177776  165536          MOV     @#PSW,$TMP0     ;SAVE THE ERROR PSW
2971  013444  010667  165526                  MOV     SP,$REG6        ;SAVE THE ERROR STACK POINTER
2972  013450  017767  165736  165526          MOV     @DLRDBR,$TMP1   ;SAVE THE ERROR REGISTER IN TMP1
2973  013456  005167  165744                  COM     RFLG1           ;SET THE SOFT ERROR FLAG
2974  013462  042777  000100  165724  2$:     BIC     #100,@DLXCSR    ;TURN OFF THE INTR. ENABLES
2975  013470  042777  000100  165712          BIC     #100,@DLRCSR
2976  013476  000411                          BR      4$              ;GO TO EXIT
2977  013500  022767  022322  165726  3$:     CMP     #BUFEND,IPTR    ;RECEIVED 256. BYTES YET ??
2978  013506  001405                          BEQ     4$              ;BR IF YES
2979  013510  117777  165676  165716          MOVB    @DLRDBR,@IPTR   ;INPUT A BYTE FROM THE DL11
2980  013516  005267  165712                  INC     IPTR            ;UPDATE BUFFER POINTER
2981  013522  000002                  4$:     RTI                     ;RETURN TO MAINLINE TEST
2982
2983                                  ;THE FOLLOWING ROUTINE IS USED BY THE USER UTILITY PROGRAMS TO WAIT
2984                                  ;A SPECIFIED NO. OF MILLISECONDS BETWEEN CHARACTER TRANSFERS
2985
2986  013524  017667  000000  000034  DELAY:  MOV     @(R6),DELCNT    ;GET THE NO. OF MSEC. DELAY COUNT
2987                                                                  ;TYPED IN BY USER
2988  013532  062716  000002                  ADD     #2,(R6)         ;SET UP THIS ROUTINE'S EXIT ADDRESS
2989  013536  005767  000024                  TST     DELCNT          ;IS THE DELAY COUNT ZERO?
2990  013542  001410                          BEQ     3$              ;BRANCH IF YES
2991  013544  012746  000226          1$:     MOV     #226,-(SP)      ;PUSH A 1 MSEC. COUNT TO STACK
2992  013550  005316                  2$:     DEC     (SP)            ;DECREMENT THE 1 MSEC. COUNT BY 1
2993  013552  001376                          BNE     2$              ;BRANCH IF 1 MSEC. NOT EATEN
2994                                                                  ;AWAY YET
2995  013554  005726                          TST     (SP)+           ;RESET STACK AFTER 1 MSEC. TIME UP
2996  013556  005367  000004                  DEC     DELCNT          ;DECREMENT THE TOTAL NO. OF
2997                                                                  ;MSECS. COUNT
2998  013562  001370                          BNE     1$              ;BRANCH IF WE HAVE MORE MSECS.
2999                                                                  ;TO WAIT
3000  013564  000207                  3$:     RTS     PC              ;GO BACK TO REISSUE A CHARACTER
3001  013566  000000                  DELCNT: .WORD   0               ;THE NO. OF MSECS. NEEDED TO
```

```
                                                        ;TRANSPIRE RESIDES HERE
                                ;THE FOLLOWING ROUTINE IS USED BY USER PROGRAM #4 AND WILL ALLOW
                                ;A RANDOM NUMBER OF MILLISECONDS BEFORE TRANSMISSION OF CHARACTER
                                ;
013570  016700  000062    STALL:  MOV     NUMONE,R0       ;GET THE LOW LIMIT
013574  006100                    ROL     R0              ;MULTIPLY BY 4
013576  006100                    ROL     R0              ;
013600  066700  000054            ADD     NUMTWO,R0       ;ADD IN THE HIGH LIMIT
013604  010067  000046            MOV     R0,NUMONE       ;STORE THIS AS NEW LOW LIMIT
013610  006100                    ROL     R0              ;MULTIPLY NEW LOW LIMIT BY 4
013612  006100                    ROL     R0              ;
013614  066700  000040            ADD     NUMTWO,R0       ;ADD IN THE HIGH LIMIT
013620  006100                    ROL     R0              ;MULTIPLY BY 4 AGAIN
013622  006100                    ROL     R0              ;
013624  010067  000030            MOV     R0,NUMTWO       ;STORE THIS AS NEW HIGH LIMIT
013630  016700  000022            MOV     NUMONE,R0       ;SAVE THE RANDOMLY GENERATED NO.
013634  046700  165446            BIC     STLMSK,R0       ;STRIP ALL BUT 1ST 5 BITS SO AS
                                                        ;NOT TO ALLOW THE STALL TO BE TOO
                                                        ;LARGE
013640  001405                    BEQ     2$              ;BRANCH IF RESULT WAS ZERO
013642  010067  000004            MOV     R0,1$           ;SET STALL TIME FOR DELAY ROUTINE
013646  004767  177652            JSR     PC,DELAY        ;GO OFF TO STALL
013652  000000          1$:       .WORD   0               ;THIS IS WHERE STALL TIME RESIDES
013654  000207          2$:       RTS     PC              ;RETURN TO ISSUE CHARACTER
013656  001233          NUMONE:   1233                    ;LOW LIMIT FOR RANDOM NO.
013660  007622          NUMTWO:   7622                    ;HIGH LIMIT FOR RANDOM NO.
                                ;THE FOLLOWING ROUTINE CHECKS THE 'DONE' BIT FOR BOTH THE RECEIVER
                                ;AND TRANSMITTER.  THIS ROUTINE IS USED BY PROGRAM #4
                                ;
013662  016767  165322  000044 TIMERX: MOV     $TMP3,DUT       ;GET THE TRANSMITTER CONTROL
                                                        ;STATUS REGISTER ADDRESS
013670  162767  000004  000036        SUB     #4,DUT          ;FORM THE RECEIVER CONTROL
                                                        ;STATUS REGISTER ADDRESS
013676  000403                    BR      TCONT           ;GO TO TIME OUT THE RECEIVERS'
                                                        ;DONE BIT
013700  016767  165304  000026 TIMETX: MOV     $TMP3,DUT       ;GET THE TRANSMITTER CONTROL
                                                        ;STATUS REGISTER ADDRESS
013706  005067  165304          TCONT:  CLR     $TMP6           ;INITIALIZE A TIME COUNT
013712  005267  165300          1$:     INC     $TMP6           ;INCREMENT THE TIME COUNT
013716  001405                  BEQ     2$              ;BRANCH IF TIME COUNTER OVERFLOWED
                                                        ;INDICATING DONE BIT NEVER SET
                                                        ;WITH PLENTY OF TIME ELAPSED
013720  105777  000010          TSTB    @DUT            ;SEE IF DONE BIT IS SET YET
013724  100372                  BPL     1$              ;WAIT SOME MORE IF IT ISN'T
013726  062716  000006          ADD     #6,@R6          ;DONE BIT IS SET - SET UP EXIT
                                                        ;RETURN TO SKIP ERROR REPORT
013732  000207          2$:     RTS     PC              ;RETURN TO PROGRAM #4
013734  000000          DUT:    .WORD   0               ;THIS IS WHERE THE RCSR OR XCSR
                                                        ;ADDRESS RESIDES
                                ;THIS ROUTINE IS USED BY PROGRAMS #4 & 5, AND WILL CHECK FOR CORRECT
                                ;EXPECTED AND RECEIVED DATA, IN ADDITION TO ANY ERROR BITS
                                ;
013736  016767  165252  165252 DATCHK: MOV     $TMP5,$TMP6     ;GET THE CONTENTS OF THE RECEIVER
                                                        ;BUFFER
013744  016767  165242  165214        MOV     $TMP4,$REG2     ;STORE THE ADDRESS OF THE RECEIVER
                                                        ;DATA BUFFER
```

```
 3058  013752  016767  165210  165204          MOV     $REG2,$REG1     ;GET THE ADDRESS OF THE RECEIVER
                                                                       ;DATA BUFFER
 3059  013760  162767  000002  165176          SUB     #2,$REG1        ;FORM THE ADDRESS OF THE RECEIVER
                                                                       ;STATUS REGISTER FROM IT
 3060  013766  016767  165232  165174          MOV     $TMP5,$REG3     ;STORE THE CONTENTS OF THE RECEIVER
                                                                       ;DATA BUFFER
 3061  013774  032767  170000  165214          BIT     #170000,$TMP6   ;ARE ANY ERROR BITS SET?
 3062  014002  001013                          BNE     1$              ;BRANCH IF YES
 3063  014004  004767  000720                  JSR     PC,UPMASK       ;GO TO MASK OFF BITS AS A FUNCTION OF
                                                                       ;CHARACTER LENGTH( 5, 6, 7, OR 8 BITS)
 3064  014010  026767  165200  165214          CMP     $TMP5,$TMP14    ;WAS RECEIVED CHARACTER THE
                                                                       ;SAME AS THE ONE TRANSMITTED?
 3065  014016  001406                          BEQ     2$              ;BRANCH IF YES
 3066  014020  016767  165206  165144          MOV     $TMP14,$REG4    ;STORE WHAT THE CONTENTS OF THE
                                                                       ;RECEIVER DATA BUFFER SHOULD BE
 3067  014026  104010                          ERROR   +10             ;DATA RECEIVED WRONG!
 3068  014030  000401                          BR      2$              ;GET SET TO RETURN AFTER ERROR REPORT
 3069  014032  104007                  1$:     ERROR   +7              ;ERROR BIT/S SET FROM TRANSMISSION
 3070  014034  000207                  2$:     RTS     PC              ;RETURN TO PROGRAM #4
 3071
 3072                                  ;;*****************************************************************
 3073                                          ;SUBROUTINE TO SETUP ERROR INFORMATION FOR ERROR MESSAGES
 3074                                  ;;*****************************************************************
 3075
 3076
 3077
 3078
 3079
 3080
 3081
```

```
014036  013767  177776  165136  SUER2:  MOV     @#PSW,$TMP0         ;SAVE THE [PSW]
014044  016701  165340                  MOV     DLRCSR,R1           ;PUT DEVADR IN R1
014050  011203                          MOV     (R2),R3             ;PUT WAS INFO IN R3
014052  010667  165120                  MOV     SP,$REG6            ;SAVE THE [SP]
014056  062767  000002  165112          ADD     #2,$REG6            ;CORRECT FOR CALLING JSR
014064  116700  165012          SUERR1: MOVB    $TSTNM,R0           ;PUT TEST NO. IN R0
014070  010067  165066                  MOV     R0,$REG0            ;SAVE [R0] THRU [R4]
014074  010167  165064                  MOV     R1,$REG1
014100  010267  165062                  MOV     R2,$REG2
014104  010367  165060                  MOV     R3,$REG3
014110  010467  165056                  MOV     R4,$REG4
014114  000207                          RTS     PC                  ;RETURN TO CALLING TEST

014116  013767  177776  165056  SUERT1: MOV     @#PSW,$TMP0                 ;SAVE THE [PSW]
014124  116700  164752                  MOVB    $TSTNM,R0                   ;PUT TEST NO. IN R0
014130  016701  165254                  MOV     DLRCSR,R1                   ;PUT DEVADR IN R1
014134  010067  165022                  MOV     R0,$REG0                    ;SAVE [R0]
014140  010167  165020                  MOV     R1,$REG1                    ;SAVE [R1]
014144  013767  177776  165032  SUERT2: MOV     @#PSW,$TMP1                 ;SAVE THE [PSW]
014152  010667  165020                  MOV     SP,$REG6                    ;SAVE THE [SP]
014156  062767  000002  165012          ADD     #2,$REG6                    ;CORRECT FOR CALLING JSR
014164  010267  164776                  MOV     R2,$REG2                    ;SAVE [R2]
014170  000207                          RTS     PC                          ;RETURN

                                ;SUBROUTINE TO SETUP VECTORS FOR 256. BYTE BLOCK TRANSFER TESTS

014172  016705  165222          SUVEC:  MOV     DLVECT,R5           ;GET FIRST VECTOR ADDRESS
014176  012725  013402                  MOV     #RINT,(R5)+         ;SET UP RCVR VECTOR
014202  016725  165074                  MOV     DLPRI,(R5)+
014206  012725  013314                  MOV     #XINT,(R5)+         ;SET UP XMIT VECTOR
014212  016715  165064                  MOV     DLPRI,(R5)
014216  000207                          RTS     PC                  ;RETURN TO CALLER

                                ;SUBROUTINE TO PRIME DATA BUFFERS AND DEVICE FOR 256. BYTE TRANSFER

014220  005077  165170          PRIME:  CLR     @DLXCSR             ;CLEAR XMIT AND RCVR CSR'S
014224  005077  165160                  CLR     @DLRCSR
014230  005067  165166                  CLR     XFLG0               ;INITIALIZE ERROR FLAGS
014234  005067  165164                  CLR     RFLG0
014240  005067  165162                  CLR     RFLG1
014244  012767  021322  165160          MOV     #DLBUF0,OPTR        ;SET UP OUTPUT POINTER
014252  012767  021722  165154          MOV     #DLBUFI,IPTR        ;SET UP INPUT POINTER
014260  004767  000044                  JSR     PC,CLDLBF           ;GO CLEAR THE BUFFERS
014264  004777  165146                  JSR     PC,@LDOUT           ;GO SET UP THE PATTERN
014270  005067  165144                  CLR     TIMR1               ;INIT TIMEOUT COUNTERS
014274  012767  000036  165140          MOV     #30.,TIMR2
014302  005777  165104                  TST     @DLRDBR             ;FLUSH "DONE" BIT IN RCVR CSR
014306  005777  165100                  TST     @DLRDBR
014312  052777  000100  165070          BIS     #100,@DLRCSR        ;ENABLE RCVR INTR.
014320  052777  000104  165066          BIS     #104,@DLXCSR        ;ENABLE XMIT INTR. AND MAINT MODE
014326  000207                          RTS     PC




                                ;THIS ROUTINE IS CALLED TO CLEAR THE INPUT AND OUTPUT BUFFERS
```

```
                                                                
                        CLDLBF: MOV     #DLBUF0,R5      ;R5 POINTS TO BEGINNING OF BUFFER AREA
                        1$:     CLR     (R5)+           ;CLEAR A WORD
                                CMP     #BUFEND,R5      ;DONE ALL WORDS ??
                                BNE     1$              ;BR IF NOT
                                RTS     PC              ;RETURN TO CALLER
                                
                        ;THIS ROUTINE IS CALLED TO SET UP THE NULL-DEL-NULL PATTERN
                                
                        LDOUT1: MOV     #DLBUF0,R5      ;R5 POINTS TO OUTPUT BUFFER
                        1$:     CLRB    (R5)+           ;MOVE A NULL CHAR
                                MOVB    #377,(R5)+      ;MOV A DEL CHAR
                                CMP     #DLBUFI,R5      ;ALL DONE ??
                                BNE     1$              ;BR IF NOT
                                RTS     PC              ;RETURN TO CALLER
                                
                        ;THIS ROUTINE IS USED TO LOAD AN ASCENDING BINARY COUNT PATTERN
                                
                        LDOUT2: CLR     R5              ;START WITH 000
                        1$:     MOVB    R5,DLBUF0(R5)   ;LOAD ONE BYTE
                                INC     R5              ;INCREMENT BYTE
                                CMP     #400,R5         ;DONE 000 THRU 377 ??
                                BNE     1$              ;BR IF NOT
                                RTS     PC              ;RETURN TO CALLER
                                
                        ;THIS ROUTINE IS USED TO LOAD A DESCENDING BINARY COUNT PATTERN
                                
                        LDOUT3: MOVB    #377,$TMP7      ;START WITH A 377 BYTE
                                MOV     #DLBUF0,R5      ;R5 POINTS TO OUTPUT BUFFER
                        1$:     MOVB    $TMP7,(R5)+     ;LOAD ONE BYTE
                                CMP     #DLBUFI,R5      ;ALL DONE ??
                                BEQ     2$              ;BR IF YES
                                DECB    $TMP7           ;GENERATE NEXT BYTE
                                BR      1$              ;GO MOVE IT
                        2$:     RTS     PC              ;RETURN TO CALLER
                                
                        ;THIS ROUTINE LOADS A COMPLEMENTING WORST CASE PATTERN
                                
                        LDOUT4: MOV     #DLBUF0,R5      ;R5 POINTS TO OUTPUT BUFFER
                                CLR     $TMP7           ;INIT. BYTE GENERATOR
                        1$:     MOVB    $TMP7,(R5)+     ;MOVE A BYTE
                                COMB    $TMP7           ;COMPLEMENT IT
                                MOVB    $TMP7,(R5)+     ;NOW LOAD THE 1'S COMPLEMENT
                                INCB    $TMP7+1         ;INCREMENT THE BYTE
                                MOVB    $TMP7+1,$TMP7   ;SET UP TO LOAD NEXT TWO
                                CMP     #DLBUFI,R5      ;ALL DONE ??
                                BNE     1$              ;BR IF NOT
                                RTS     PC              ;RETURN TO CALLER
                                
                        ;THIS ROUTINE CHECKS FOR DATA COMPARE ERRORS IN 256. BYTE BLOCK TRANSFERS
                                
                        CHKDAT: BIC     #104,@DLXCSR    ;DISABLE BOTH XMIT AND RCVR INTR. ENAB.
                                BIC     #100,@DLRCSR
                                MOV     #DLBUF0,R2      ;R2 POINTS TO S/B DATA IN OUTPUT BUFFER
                                JSR     PC,MASKING      ;GO TO MASK OFF BITS AS A FUNCTION OF
```

```
                                                                ;CHARACTER LENGTH(5, 6, 7, OR 8 BITS)
014536  012701  021722                MOV     #DLBUFI,R1      ;R1 POINTS TO WAS DATA IN RCVR. BUFFER
014542  122221                1$:     CMPB    (R2)+,(R1)+     ;DID S/B = WAS ??
014544  001004                        BNE     3$              ;BR IF NOT
014546  022701  022322        2$:     CMP     #BUFEND,R1      ;CHECKED ALL BYTES ??
014552  001373                        BNE     1$              ;BR IF NOT
014554  000207                        RTS     PC              ;RETURN TO CALLER
014556  013767  177776  164416  3$:   MOV     @#PSW,$TMPO     ;SAVE THE [PSW]
014564  010667  164406                MOV     SP,$REG6        ;SAVE THE [SP]
014570  114204                        MOVB    -(R2),R4        ;GET THE S/B DATA
014572  042704  177400                BIC     #177400,R4      ;CLEAR JUNK FROM HI BYTE
014576  114103                        MOVB    -(R1),R3        ;GET THE WAS DATA
014600  042703  177400                BIC     #177400,R3      ;CLEAR JUNK FROM HI BYTE
014604  004767  177254                JSR     PC,SUERR1       ;GO SET UP ERROR INFO.
014610  012767  014620  164426        MOV     #4$,$ESCAPE     ;RETURN TO 4$ AFTER ERROR PRINT
014616  104003                        ERROR+3                 ;DATA COMPARE ERROR
014620  005202                4$:     INC     R2              ;REPOSITION BUFFER POINTERS
014622  005201                        INC     R1
014624  000750                        BR      2$              ;GO CHECK NEXT BYTE

                        ;THIS ROUTINE IS USED BY THE PATTERN TESTS
                        ;IT WILL MASK OFF THE CHARACTER SENT OUT BY THE XMITTER
                        ;BEFORE THE COMPARISON OF DATA OF WHAT WAS RECEIVED AND WHAT WAS TRANSMITTED
                        ;IS DONE.  THE MASKING IS DONE AS A FUNCTION OF CHARACTER LENGTH WHICH
                        ;CAN BE EITHER 5, 6, 7, OR 8 BITS .

014626  005005          MASKING:      CLR     R5              ;INITIALIZE TABLE OFFSET
                                                              ;FOR PICKING UP MASK WORD
014630  022767  000010  164376        CMP     #8.,$TMP15      ;IS THE CHARACTER LENGTH 8 BITS?
014636  001427                        BEQ     3$              ;BRANCH IF IT IS
014640  062705  000002                ADD     #2,R5           ;SET UP FOR NEXT MASK WORD
                                                              ;IT COULD BE THIS ONE
014644  022767  000007  164362        CMP     #7.,$TMP15      ;IS THE CHARACTER LENGTH 7 BITS?
014652  001410                        BEQ     1$              ;BRANCH IF IT IS
014654  062705  000002                ADD     #2,R5           ;SET UP FOR NEXT MASK WORD
                                                              ;IT COULD BE THIS ONE
014660  022767  000006  164346        CMP     #6.,$TMP15      ;IS THE CHARACTER LENGTH 6 BITS?
014666  001402                        BEQ     1$              ;BRANCH IF IT IS
014670  062705  000002                ADD     #2,R5           ;SET UP FOR NEXT MASK WORD
                                                              ;IT MUST BE THIS ONE!!!!
014674  016505  014720        1$:     MOV     CHARL(R5),R5    ;PICK UP THE MASK WORD
014700  005105                        COM     R5              ;FORM THE BITS THAT ARE TO BE MASKED
014702  140522                2$:     BICB    R5,(R2)+        ;MASK A BYTE
014704  022702  021722                CMP     #DLBUFI,R2      ;ARE WE AT THE END OF THE XMITTER
                                                              ;OUTPUT BUFFER
014710  001374                        BNE     2$              ;BRANCH IF NO TO MASK NEXT BYTE
014712  012702  021322                MOV     #DLBUFO,R2      ;RESTORE R2 BEFORE RETURNING
014716  000207                3$:     RTS     PC              ;RETURN TO MAINLINE CODE
                        ;TABLE OF MASK WORDS
014720  000377          CHARL:  .WORD   377                   ;8. BITS IN LENGTH
014722  000177                  .WORD   177                   ;7. BITS IN LENGTH
014724  000077                  .WORD   77                    ;6. BITS IN LENGTH
014726  000037                  .WORD   37                    ;5. BITS IN LENGTH

                        ;THIS ROUTINE IS USED BY PROGRAMS #4 &
                        ;IT WILL MASK OFF THE CHARACTER SENT OUT BY THE TRANSMITTER
```

```
                                        ;BEFORE THE COMPARISON OF DATA OF WHAT WAS RECEIVED AND WHAT WAS
                                        ;TRANSMITTED IS DONE.  THE MASKING IS DONE AS A FUNCTION OF CHARACTER
                                        ;LENGTH WHICH CAN BE EITHER 5, 6, 7, OR 8 BITS.

014730  016767  164250  164274  UPMASK: MOV     $TMP1,$TMP14    ;PICK UP THE CHARACTER THAT WAS
                                                                ;SENT OUT FROM THE XMITTER
014736  005005                          CLR     R5              ;INITIIALIZE TABLE OFFSET
                                                                ;FOR PICKING UP MASK WORD
014740  022767  000010  164266          CMP     #8.,$TMP15      ;IS THE CHARACTER LENGTH 8 BITS?
014746  001423                          BEQ     2$              ;BRANCH IF IT IS
015750  062705  000002                  ADD     #2,R5           ;SET UP FOR NEXT MASK WORD
                                                                ;IT COULD BE THIS ONE
014754  022767  000007  164252          CMP     #7.,$TMP15      ;IS THE CHARACTER LENGTH 7 BITS?
014762  001410                          BEQ     1$              ;BRANCH IF IT IS
014764  062705  000002                  ADD     #2,R5           ;SET UP FOR NEXT MASK WORD
                                                                ;IT COULD BE THIS ONE
014770  022767  000006  164236          CMP     #6.,$TMP15      ;IS THE CHARACTER LENGTH 6 BITS?
014776  001402                          BEQ     1$              ;BRANCH IF IT IS
015000  062705  000002                  ADD     #2,R5           ;SET UP FOR NEXT MASK WORD
                                                                ;IT MUST BE THIS ONE!!!!
015004  016505  014720          1$:     MOV     CHARL(R5),R5    ;PICK UP THE MASK WORD
015010  005105                          COM     R5              ;FORM THE BITS THAT ARE TO BE MASKED
015012  140567  164214                  BICB    R5,$TMP14       ;MASK THE LOW BYTE
015016  000207                  2$:     RTS     PC              ;RETURN TO MAINLINE CODE

                                        ;ROUTINE TO SERVICE BUS ERROR TRAPS

015020  112767  000060  000632  BUSERR: MOVB    #60,EM4+46      ;SET UP ERROR MESSAGE
015026  112767  000060  000625          MOVB    #60,EM4+47
015034  112767  000064  000620          MOVB    #64,EM4+50
015042  000412                          BR      TRPCOM          ;GO SET UP AND REPORT BUS ERROR

                                        ;ROUTINE TO SERVICE RSVD INSTRUCTION TRAPS

015044  112767  000060  000606  RSVERR: MOVB    #60,EM4+46      ,SET UP ERROR MESSAGE
015052  112767  000061  000601          MOVB    #61,EM4+47
015060  112767  000060  000574          MOVB    #60,EM4+50
015066  000400                          BR      TRPCOM          ;GO SET UP AND REPORT RSVD INSTR. ERROR

                                        ;ROUTINE TO SET UP AND REPORT BUS ERROR AND RSVD INSTR ERRORS

015070  010667  164102          TRPCOM: MOV     SP,$REG6        ;SAVE THE TRAP SP
015074  116700  164002                  MOVB    $TSTNM,R0       ;PUT TEST NO. IN R0
015100  010067  164056                  MOV     R0,$REG0        ;SAVE TEST #
015104  016667  000002  164070          MOV     2(SP),$TMP0     ;SAVE THE ERROR PSW
015112  012767  015126  164124          MOV     #1$,$ESCAPE     ;GO TO 1$ AFTER ERROR PRINT
015120  011667  164054                  MOV     (SP),$REG7      ;SAVE THE ERROR PC
015124  104004                          ERROR+4                 ;REPORTED TRAP ERROR
015126  000137  001772          1$:     JMP     @#RESTRT        ;ATTEMPT TO RESTART THE PROGRAM
                                                                ;AND TRY AGAIN


                                        ;;*****************************************************************
                                        ;ERROR MESSAGE INFORMATION
                                        ;;*****************************************************************
```

```
3306                                        ;INFORMATION FOR ERROR MESSAGE 1
3307
3308  015132  046104  030461  051040  EM1:    .ASCIZ  'DL11 REGISTER REFERENCE CAUSED TIMEOUT'
3309  015140  043505  051511  042524
3310  015146  020122  042522  042506
3311  015154  042522  041516  020105
3312  015162  040503  051525  042105
3313  015170  052040  046511  047505
3314  015176  052125     000
3315  015201     040  050050  024503  DH1:    .ASCIZ  ' (PC)    (PS)    (SP)    TEST   DEVADR  REGADR'
3316  015206  020040  020040  050050
3317  015214  024523  020040  020040
3318  015222  051450  024520  020040
3319  015230  020040  042524  052123
3320  015236  020040  042040  053105
3321  015244  042101  020122  051040
3322  015252  043505  042101  000122
3323                                    .EVEN
3324  015260  001116  001202  001176  DT1:    .WORD   $ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,0
3325  015266  001162  001164  001166
3326  015274  000000
3327
3328                                        ;INFORMATION FOR ERROR MESSAGE 2
3329
3330  015276  046104  030461  051040  EM2:    .ASCIZ    11 REGISTER ERROR'
3331  015304  043505  051511  042524
3332  015312  020122  051105  047522
3333  015320  000122
3334  015322  024040  041520  020051  DH2:    .ASCIZ  ' (PC)    (PS)    (SP)    TEST   DEVADR  REGADR   WAS     S/B'
3335  015330  020040  024040  051520
3336  015336  020051  020040  024040
3337  015344  050123  020051  020040
3338  015352  052040  051505  020124
3339  015360  020040  042504  040526
3340  015366  051104  020040  042522
3341  015374  040507  051104  020040
3342  015402  053440  051501  020040
3343  015410  020040  051440  041057
3344  015416     000
3345          015420                    .EVEN
3346  015420  001116  001202  001176  DT2:    .WORD   $ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4,0
3347  015426  001162  001164  001166
3348  015434  001170  001172  000000
3349
3350                                        ;INFORMATION FOR MESSAGE 3
3351
3352  015442  046104  030461  042040  EM3:    .ASCIZ  'DL11 DATA COMPARE ERROR'
3353  015450  052101  020101  047503
3354  015456  050115  051101  020105
3355  015464  051105  047522  000122
3356  015472  024040  041520  020051  DH3:    .ASCIZ  ' (PC)    (PS)    (SP)    TEST   WASADR  SHBADR    WAS     S/B'
3357  015500  020040  024040  051520
3358  015506  020051  020040  024040
3359  015514  050123  020051  020040
3360  015522  052040  051505  020124
3361  015530  020040  040527  040523
```

```
 3362  015536  051104  020040  044123
 3363  015544  040502  051104  020040
 3364  015552  020040  040527  020123
 3365  015560  020040  020040  027523
 3366  015566  000102
 3367                                  .EVEN
 3368  015570  001116  001202  001176  DT3:    .WORD   $ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,$REG4,0
 3369  015576  001162  001164  001166
 3370  015604  001170  001172  000000
 3371
 3372                                  ;INFORMATION FOR MESSAGE 4
 3373
 3374  015612  047125  054105  042520  EM4:    .ASCIZ  'UNEXPECTED TRAP TO VECTOR AT LOCATION    '
 3375  015620  052103  042105  052040
 3376  015626  040522  020120  047524
 3377  015634  053040  041505  047524
 3378  015642  020122  052101  046040
 3379  015650  041517  052101  047511
 3380  015656  020116  020040  000040
 3381  015664  024040  041520  020051  DH4:    .ASCIZ  ' (PC)    (PS)    (SP)    TEST'
 3382  015672  020040  024040  051520
 3383  015700  020051  020040  024040
 3384  015706  050123  020051  020040
 3385  015714  052040  051505  000124
 3386                                  .EVEN
 3387  015722  001200  001202  001176  DT4:    .WORD   $REG7,$TMPO,$REG6,$REG0,0
 3388  015730  001162  000000
 3389
 3390                                  ;ERROR INFORMATION FOR ERROR MESSAGE 5
 3391
 3392  015734  046104  030461  051440  EM5:    .ASCIZ  'DL11 SOFT ERROR (PARITY,FRAMING, OR OVERRUN)'
 3393  015742  043117  020124  051105
 3394  015750  047522  020122  050050
 3395  015756  051101  052111  026131
 3396  015764  051106  046501  047111
 3397  015772  026107  047440  020122
 3398  016000  053117  051105  052522
 3399  016006  024516     000
 3400  016011     040  050050  024503  DH5:    .ASCIZ  ' (PC)    (PS)    (SP)    TEST   DEVADR  REGADR   (REG)'
 3401  016016  020040  020040  050050
 3402  016024  024523  020040  020040
 3403  016032  051450  024520  020040
 3404  016040  020040  042524  052123
 3405  016046  020040  042040  053105
 3406  016054  042101  020122  051040
 3407  016062  043505  042101  020122
 3408  016070  020040  051050  043505
 3409  016076  000051
 3410                                  .EVEN
 3411  016100  001116  001202  001176  DT5:    .WORD   $ERRPC,$TMPO,$REG6,$REG0,$REG1,$REG2,$REG3,0
 3412  016106  001162  001164  001166
 3413  016114  001170  000000
 3414
 3415                                  ;INFORMATION FOR ERROR MESSAGE 6
 3416
 3417  016120  024040  041520  020051  DH6:    .ASCIZ  ' (PC)    (PS)    (SP)   REGADR'
```

```
3418  016126  020040  024040  051520
3419  016134  020051  020040  024040
3420  016142  050123  020051  020040
3421  016150  042522  040507  051104
3422  016156     000
3423          016160                      .EVEN
3424  016160  001116  001204  001176  DT6:    .WORD   $ERRPC,$TMP1,$REG6,$REG2,0
3425  016166  001166  000000
3426
3427                                    ;INFORMATION FOR ERROR MESSAGE 7
3428
3429  016172  024040  041520  020051  DH7:    .ASCIZ  ' (PC)   DEVADR  REGADR   (REG)'
3430  016200  020040  042504  040526
3431  016206  051104  020040  042522
3432  016214  040507  051104  020040
3433  016222  024040  042522  024507
3434  016230     000
3435          016232                      .EVEN
3436  016232  001116  001164  001166  DT7:    .WORD   $ERRPC,$REG1,$REG2,$REG3,0
3437  016240  001170  000000
3438
3439                                    ;INFORMATION FOR ERROR MESSAGE 10
3440
3441  016244  024040  041520  020051  DH10:   .ASCIZ  ' (PC)   DEVADR  REGADR   (REG)    S/B'
3442  016252  020040  042504  040526
3443  016260  051104  020040  042522
3444  016266  040507  051104  020040
3445  016274  024040  042522  024507
3446  016302  020040  020040  027523
3447  016310  000102
3448                                    .EVEN
3449  016312  001116  001164  001166  DT10:   .WORD   $ERRPC,$REG1,$REG2,$REG3,$REG4,0
3450  016320  001170  001172  000000
3451                                    ;MISCELLANEOUS MESSAGES
3452
3453  016326  052516  046114  042055  XMSG1:  .ASCIZ  'NULL-DEL-NULL SEQUENCE TIMEOUT AT FOLLOWING PC'
3454  016334  046105  047055  046125
3455  016342  020114  042523  052521
3456  016350  047105  042503  052040
3457  016356  046511  047505  052125
3458  016364  040440  020124  047506
3459  016372  046114  053517  047111
3460  016400  020107  041520     000
3461  016405     102  047111  051101  XMSG2:  .ASCIZ  'BINARY UP COUNT SEQUENCE TIMEOUT AT FOLLOWING PC'
3462  016412  020131  050125  041440
3463  016420  052517  052116  051440
3464  016426  050505  042525  041516
3465  016434  020105  044524  042515
3466  016442  052517  020124  052101
3467  016450  043040  046117  047514
3468  016456  044527  043516  050040
3469  016464  000103
3470  016466  044502  040516  054522  XMSG3:  .ASCIZ  'BINARY DOWN COUNT SEQUENCE TIMEOUT AT FOLLOWING PC'
3471  016474  042040  053517  020116
3472  016502  047503  047125  020124
3473  016510  042523  052521  047105
```

```
3474  016516  042503  052040  046511
3475  016524  047505  052125  040440
3476  016532  020124  047506  046114
3477  016540  053517  047111  020107
3478  016546  041520     000
3479  016551     127  051117  052123  XMSG4:  .ASCIZ  'WORST CASE PATTERN SEQUENCE TIMEOUT AT FOLLOWING PC'
3480  016556  041440  051501  020105
3481  016564  040520  052124  051105
3482  016572  020116  042523  052521
3483  016600  047105  042503  052040
3484  016606  046511  047505  052125
3485  016614  040440  020124  047506
3486  016622  046114  053517  047111
3487  016630  020107  041520     000
3488
3489  016635     015  046412  044501  STMES:  .ASCIZ  <15><12>'MAINDEC-11-DZDLC-A'<15><12>
3490  016642  042116  041505  030455
3491  016650  026461  055104  046104
3492  016656  026503  006501  000012
3493
3494  016664  005015  047531  020125  PROG2M: .ASCIZ  <15><12>'YOU HAVE SELECTED PROGRAM NO. 2'<15><12>
3495  016672  040510  042526  051440
3496  016700  046105  041505  042524
3497  016706  020104  051120  043517
3498  016714  040522  020115  047516
3499  016722  020056  006462  000012
3500  016730  005015  047531  020125  PROG3M: .ASCIZ  <15><12>'YOU HAVE SELECTED PROGRAM NO. 3'<15><12>
3501  016736  040510  042526  051440
3502  016744  046105  041505  042524
3503  016752  020104  051120  043517
3504  016760  040522  020115  047516
3505  016766  020056  006463  000012
3506  016774  005015  047531  020125  PROG4M: .ASCIZ  <15><12>'YOU HAVE SELECTED PROGRAM NO. 4'<15><12>
3507  017002  040510  042526  051440
3508  017010  046105  041505  042524
3509  017015  020104  051120  043517
3510  017024  040522  020115  047516
3511  017032  020056  006464  000012
3512  017040  005015  047531  020125  PROG5M: .ASCIZ  <15><12>'YOU HAVE SELECTED PROGRAM NO. 5'<15><12>
3513  017046  040510  042526  051440
3514  017054  046105  041505  042524
3515  017062  020104  051120  043517
3516  017070  040522  020115  047516
3517  017076  020056  006465  000012
3518  017104  005015  051124  047101  XDB:    .ASCIZ  <15><12>'TRANSMITTER DONE BIT NEVER SET  PC= '
3519  017112  046523  052111  042524
3520  017120  020122  047504  042516
3521  017126  041040  052111  047040
3522  017134  053105  051105  051440
3523  017142  052105  020040  041520
3524  017150  020075     000
3525  017153     015  051012  041505  RDB:    .ASCIZ  <15><12>'RECEIVER DONE BIT NEVER SET  PC= '
3526  017160  044505  042526  020122
3527  017166  047504  042516  041040
3528  017174  052111  047040  053105
3529  017202  051105  051440  052105
```

```
3530  017210  020040  041520  020075
3531  017216     000
3532                              ;MESSAGES SEEKING USER RESPONSE
3533
3534  017217     015  053412  040510  LENGTH: .ASCIZ  <15><12>'WHAT IS THE CHARACTER LENGTH (5,6,7 OR 9 BITS)?'
3535  017224  020124  051511  052040
3536  017232  042510  041440  040510
3537  017240  040522  052103  051105
3538  017246  046040  047105  052107
3539  017254  020110  032450  033054
3540  017262  033454  047440  020122
3541  017270  020070  044502  051524
3542  017276  037451     000
3543  017301     015  042012  020117  DEFAULT:        .ASCII  <15><12>'DO YOU WISH TO TEST OTHER THAN THE'
3544  017306  047531  020125  044527
3545  017314  044123  052040  020117
3546  017322  042524  052123  047440
3547  017330  044124  051105  052040
3548  017336  040510  020116  044124
3549  017344     105
3550  017345     015  042012  043105          .ASCIZ  <15><12>'DEFAULT DEVICE (1/0 = YES/NO)?'
3551  017352  052501  052114  042040
3552  017360  053105  041511  020105
3553  017366  030450  030057  036440
3554  017374  054440  051505  047057
3555  017402  024517  000077
3556  017406  005015  044127  052101  MFIRSTD:.ASCIZ  <15><12>'WHAT IS THE 1ST R  EIVER STATUS REGISTER ADDRESS?  '
3557  017414  044440  020123  044124
3558  017422  020105  051461  020124
3559  017430  042522  042503  053111
3560  017436  051105  051440  040524
3561  017444  052524  020123  042522
3562  017452  044507  052123  051105
3563  017460  040440  042104  042522
3564  017466  051523  020077  000040
3565  017474  005015  044127  052101  MVECT:  .ASCIZ  <15><12>'WHAT IS THE 1ST RECEIVER''S VECTOR ADDRESS?  '
3566  017502  044440  020123  044124
3567  017510  020105  051461  020124
3568  017516  042522  042503  053111
3569  017524  051105  020123  042526
3570  017532  052103  051117  040440
3571  017540  042104  042522  051523
3572  017546  020077  000040
3573  017552  005015  047504  054440  MULDEV: .ASCIZ  <15><12>'DO YOU WANT TO TEST MULTIPLE DEVICES 1/0=YES/NO?  '
3574  017560  052517  053440  047101
3575  017566  020124  047524  052040
3576  017574  051505  020124  052515
3577  017602  052114  050111  042514
3578  017610  042040  053105  041511
3579  017616  051505  030440  030057
3580  017624  054475  051505  047057
3581  017632  037517  020040     000
3582  017637     015  053412  040510  MLASTD: .ASCIZ  <15><12>'WHAT IS THE STATUS REGISTER ADDRESS OF THE LAST RECEIVER?  '
3583  017644  020124  051511  052040
3584  017652  042510  051440  040524
3585  017660  052524  020123  042522
```

```
3586  017666  044507  052123  051105
3587  017674  040440  042104  042522
3588  017702  051523  047440  020106
3589  017710  044124  020105  040514
3590  017716  052123  051040  041505
3591  017724  044505  042526  037522
3592  017732  020040     000
3593  017735     015  051412  046517  MRANGE: .ASCIZ  <15><12>'SOMETHING WRONG-ANSWER THE LAST QUESTION AGAIN!  '
3594  017742  052105  044510  043516
3595  017750  053440  047522  043516
3596  017756  040455  051516  042527
3597  017764  020122  044124  020105
3598  017772  040514  052123  050440
3599  020000  042525  052123  047511
3600  020006  020116  043501  044501
3601  020014  020516  020040     000
3602  020021     015  053412  040510  PLEVEL: .ASCIZ  <15><12>'WHAT IS YOUR INTERRUPT PRIORITY LEVEL?  '
3603  020026  020124  051511  054440
3604  020034  052517  020122  047111
3605  020042  042524  051122  050125
3606  020050  020124  051120  047511
3607  020056  044522  054524  046040
3608  020064  053105  046105  020077
3609  020072  000040
3610  020074  005015  051120  043517  FOULUP: .ASCII  <15><12>'PROGRAM DEVICE ACTIVE LOCATION SHOWS NO DEVICE ACTIVE'
3611  020102  040522  020115  047504
3612  020110  044526  042503  040440
3613  020116  052103  053111  020105
3614  020124  047514  040503  044524
3615  020132  047117  051440  047510
3616  020140  051527  047040  020117
3617  020146  042504  044526  042503
3618  020154  040440  052103  053111
3619  020162     105
3620  020163     015  051412  052105          .ASCII  <15><12>'SET SWITCH 0 TO A ONE (1) AND'
3621  020170  051440  044527  041524
3622  020176  020110  020060  047524
3623  020204  040440  047440  042516
3624  020212  024040  024461  040440
3625  020220  042116
3626  020222  005015  044510  020124          .ASCIZ  <15><12>'HIT CONTINUE TO GO BACK TO DEVICE  'LECTION AGAIN'
3627  020230  047503  052116  047111
3628  020236  042525  052040  020117
3629  020244  047507  041040  041501
3630  020252  020113  047524  042040
3631  020260  053105  041511  020105
3632  020266  042523  042514  052103
3633  020274  047511  020116  043501
3634  020302  044501  000116
3635  020306  005015  044127  052101  LINTAD: .ASCIZ  <15><12>'WHAT IS THE TRANSMITTER DATA BUFFER ADDRESS?  '
3636  020314  044440  020123  044124
3637  020322  020105  051124  047101
3638  020330  046523  052111  042524
3639  020336  020122  040504  040524
3640  020344  041040  043125  042506
3641  020352  020122  042101  051104
```

```
3642  020360  051505  037523  020040
3643  020366     000
3644  020367     015  053412  040510  SELCAR: .ASCIZ  <15><12>'WHAT IS THE CHARACTER TO BE TRANSMITTED (OCTAL ASCII E.G. A=101
3645  020374  020124  051511  052040
3646  020402  042510  041440  040510
3647  020410  040522  052103  051105
3648  020416  052040  020117  042502
3649  020424  052040  040522  051516
3650  020432  044515  052124  042105
3651  020440  024040  041517  040524
3652  020446  020114  051501  044503
3653  020454  020111  027105  027107
3654  020462  040440  030475  030460
3655  020470  037451  020040     000
3656  020475     015  053412  040510  SELDLY: .ASCIZ  <15><12>'WHAT IS THE DESIRED MSEC. DELAY (OCTAL E.G. 10=8(10))?  '
3657  020502  020124  051511  052040
3658  020510  042510  042040  051505
3659  020516  051111  042105  046440
3660  020524  042523  027103  042040
3661  020532  046105  054501  024040
3662  020540  041517  040524  020114
3663  020546  027105  027107  030440
3664  020554  036460  024070  030061
3665  020562  024451  020077  000040
3666  020570  005015  044127  052101  LINRAD: .ASCIZ  <15><12>'WHAT IS THE RECEIVER DATA BUFFER ADDRESS?  '
3667  020576  044440  020123  044124
3668  020604  020105  042522  042503
3669  020612  053111  051105  042040
3670  020620  052101  020101  052502
3671  020626  043106  051105  040440
3672  020634  042104  042522  051523
3673  020642  020077  000040
3674  020646  005015  051511  040440  RSTALL: .ASCIZ  <15><12>'IS A RANDOM WAIT TIME (MSEC.) DESIRED  1/0=YES/NO?  '
3675  020654  051040  047101  047504
3676  020662  020115  040527  052111
3677  020670  052040  046511  020105
3678  020676  046450  042523  027103
3679  020704  020051  042504  044523
3680  020712  042522  020104  030440
3681  020720  030057  054475  051505
3682  020726  047057  037517  020040
3683  020734     000
3684  020735     015  054412  052517  FAILSA: .ASCII  <15><12>'YOU HAVE SWR8 SET INDICATING LOOP ON TEST'
3685  020742  044040  053101  020105
3686  020750  053523  034122  051440
3687  020756  052105  044440  042116
3688  020764  041511  052101  047111
3689  020772  020107  047514  050117
3690  021000  047440  020116  042524
3691  021006  052123
3692  021010  005015  040510  042526          .ASCII  <15><12>'HAVE YOU MODIFIED THE PROPER LOCATIONS FOR THE'
3693  021016  054440  052517  046440
3694  021024  042117  043111  042511
3695  021032  020104  044124  020105
3696  021040  051120  050117  051105
3697  021046  046040  041517  052101
```

```
 3698  021054  047511  051516  043040
 3699  021062  051117  052040  042510
 3700  021070  005015  042504  044526          .ASCII  <15><12>'DEVICE THAT YOU WANT TO TEST?'
 3701  021076  042503  052040  040510
 3702  021104  020124  047531  020125
 3703  021112  040527  052116  052040
 3704  021120  020117  042524  052123
 3705  021126     077
 3706  021127     015  044412  020106          .ASCII  <15><12>'IF SO - PRESS THE CONTINUE SWITCH'
 3707  021134  047523  026440  050040
 3708  021142  042522  051523  052040
 3709  021150  042510  041440  047117
 3710  021156  044524  052516  020105
 3711  021164  053523  052111  044103
 3712  021172  005015  043111  047040          .ASCII  <15><12>'IF NOT - MODIFY THE PROPER LOCATIONS, THEN'
 3713  021200  052117  026440  046440
 3714  021206  042117  043111  020131
 3715  021214  044124  020105  051120
 3716  021222  050117  051105  046040
 3717  021230  041517  052101  047511
 3718  021236  051516  020054  044124
 3719  021244  047105
 3720  021246  005015  042522  052123          .ASCIZ  <15><12>'RESTART THE PROGRAM AT ADDRESS 200'
 3721  021254  051101  020124  044124
 3722  021262  020105  051120  043517
 3723  021270  040522  020115  052101
 3724  021276  040440  042104  042522
 3725  021304  051523  031040  030060
 3726  021312     000
 3727
 3728  021313     040  020075  041520          PCMSG:  .ASCII  ' = PC'
 3729  021320  000040                                  .ASCIZ  ' '
 3730
 3731                                  .EVEN
 3732                                  ;512. WORDS RESERVED FOR TWO 256. BYTE INPUT/OUTPUT DATA BUFFERS
 3733
 3734  021322  000400                  DLBUFO: .BLKB   256.            ;RSVD FOR OUTPUT BUFFER
 3735                                                                  ;THIS IS THE DATA BEING SENT OUT
 3736                                                                  ;BY THE TRANSMITTER
 3737  021722  000400                  DLBUFI: .BLKB   256.            ;RSVD FOR INPUT BUFFER
 3738                                                                  ;THIS IS THE DATA THAT WAS PICKED
 3739                                                                  ;UP BY THE RECEIVER (I.E. DATA
 3740                                                                  ;SENT BY THE TRANSMITTER - HOPEFULLY)
 3741  022322  000000                  BUFEND: 0                       ;TAG MARKS END OF BUFFERS
 3742
 3743          000001                  .END
```

```
ACTREG  001274          677#  1032*  1064*  1065*  1075*  2180  2204
BASEAD  001064          663#   984*  1070*  1071   1078*  1086*  2192*  2208  2227*  2229
BASEIV  001270          670#  1004*  2195*  2210   2228*  2230
BEGIN   001446          457    804#
BIT0    = 000001        563#   971    994    1052   1497   1503   1516  1777  1859  1944  2053
BIT00   = 000001        553#   563
BIT01   = 000002        552#   562
BIT02   = 000004        551#   561
BIT03   = 000010        550#   560
BIT04   = 000020        549#   559
BIT05   = 000040        548#   558
BIT06   = 000100        547#   557
BIT07   = 000200        546#   556
BIT08   = 000400        545#   555   2292
BIT09   = 001000        544#   554   2300   2359
BIT1    = 000002        562#  1408   1422   1467
BIT10   = 002000        543#  2344
BIT11   = 004000        542#  2307
BIT12   = 010000        541#
BIT13   = 020000        540#  2351
BIT14   = 040000        539#  2278
BIT15   = 100000        538#  1342   1354   1373   1385   1387   1409  1421  1423  1481
BIT2    = 000004        561#  1223   1229   1293   1300   1313   1341  1353  1905  2007  2116
BIT3    = 000010        560#  1372   1396
BIT4    = 000020        559#
BIT5    = 000040        558#  1442   1448   1466
BIT6    = 000100        557#  1269   1275   1292   1315
BIT7    = 000200        556#
BIT8    = 000400        555#
BIT9    = 001000        554#
BPTVEC= 000014          570#
BUFEND  022322         1538   1595   1662   1709   2977   3141   3198  3741#
BUSERR  015020          897   3277#
CHARL   014720         3234   3243#  3270
CHKDAT  014512         1540   1597   1654   1711   3190#
CLDLBF  014330         3124   3139#
CONQUE  002626         1037   1081   1093#  1134
CR      = 000015        478#  2779   2789
CRLF    = 000200        479#  2750   2789
DATCHK  013736         2021   2129   3054#
DDISP   = 177570        485#   607    831
DEFAUL  017301          942   3543#
DELAY   013524         1825   1910   2986#  3023
DELCNT  013566         2986*  2989   2996*  3001#
DH1     015201          728   3315#
DH10    016244          777   3441#
DH2     015322          735   3334#
DH3     015472          742   3356#
DH4     015664          749   3381#
DH5     016011          756   3400#
DH6     016120          763   3417#
DH7     016172          770   3429#
DISPLA  001142          607#   831*   839*  2321*  2343*
DISPRE  000174          454#   839
DLCOUN  010054          891    982   2147#  2212   2232
DLBASE  001260          657#   890*   978*   981   2147   2149*  2151  2153*  2155  2157*  2159  2208*  2219*
```

```
DLBUFI  021722          2952    3123    3150    3168    3184    3195    3237    3737#
DLBUFO  021332          3122    3139    3147    3157*   3166    3177    3192    3240    3734#
DLPRI   001302           693#    851*   1107*   1113*   1114*   1115*   1116*   1117*   1118    1241    1286    1461    3110
                        3112
DLRCSR  001410           785#    919*   1125*   1143    1202    1268    1290    1339    1342    1354    1369    1373    1387
                        1405    1409    1423    1440    1465    1546*   1553    1562    1571    1603*   1610    1619    1628
                        1660*   1667    1676    1685    1717*   1723    1731    1739    2147*   2217*   2949*   2962    2975*
                        3083    3097    3118*   3130*   3191*
DLRDBR  001412           786#    920     921     1126    1127    1173    1323    1572    1629    1686    1740    2151*   2218
                        2219    2968    2972    2979    3128    3129
DLVECT  001420           799#    893*   1001*   1003    1004    1239    1284    1459    2210*   2230*   3108
DLXCSR  001414           787#    918*   1124*   1158    1212    1222    1244    1496    1515    1547*   1554    1604*   1611
                        1661*   1668    1718*   1724    2155*   2216*   2944    2950*   2974*   3117*   3131*   3190*
DLXDBR  001416           788#   1198    1324    2159*   2954*
DSWR    = 177570         484#    606     830
DT1     015260           729    3324#
DT10    016312           778    3449#
DT2     015420           736    3346#
DT3     015570           743    3368#
DT4     015722           750    3387#
DT5     016100           757    3411#
DT6     016160           764    3424#
DT7     016232           771    3436#
DUT     013734          3031*   3033*   3037*   3044    3049#
EMTVEC= 000030           573#    815*    816*   1762*   1763*   1844*   1845*   1929*   1930*   2038*   2039*
EM1     015132           727     762    3308#
EM2     015276           734    3330#
EM3     015442           741     776    3352#
EM4     015612           748    3277*   3278*   3279*   3284*   3285*   3286*   3374#
EM5     015734           755     769    3392#
ERRVEC= 000004           566#    828     829*    840*    897*    898*   1141    1142*   1150*   1156    1157*   1165*   1171
                        1172*   1180*   1186    1187*   1195*   1791    1792*   1801*   1803*   1872    1873*   1882*   1884*
                        1957    1958*   1967*   1969*   2066    2067*   2076*   2078*   2283    2284*   2286*   2289*
FAILSA  020735           874    3684#
FIRSTD  002200           960#   1012
FLUSH   002724           947    1124#
FOULUP  020074          2182    3610#
GNS     = ****** U       453    2882    2883    2884    2885    2886    2889    2890    2891    2892
GO      002100           917     931#   1009
HOWMAN  002376          1005    1016#
HT      = 000011         476#   2748    2789
INTFLG  001444           799#   1238*   1246    1256*   1287*   1295    1316*   1463*   1468    1479*
IOTVEC= 000020           571#    813*    814*
IPTR    001434           795#   1538    1595    1652    1709    2977    2979*   2980*   3123*
KEEPAD  001262           659#    981*    984    1078    1086    2227
KEEPIV  001266           666#   1003*   2228
LASTAD  001300           689#   1059*   1071
LASTD   002446          1031    1038#   1131
LDOUT   001436           796#   1530*   1587*   1644*   1701*   3125
LDOUT1  014346          1530    3147#
LDOUT2  014370          1587    3156#
LDOUT3  014410          1644    3165#
LDOUT4  014444          1701    3177#
LENGTH  017217           931    1387    2082    3534#
LESS1   001304           695#   1118*   1120*   1122*
LF      = 000012         477#   2783    2789
```

```
LINRAD  020570          1948   3666#
LINTAD  020306          1766   1933   2042   3635#
MASKIN  014626          3193   3220#
MFIRST  017406           960   3556#
MLASTD  017637          1041   3582#
MRANGE  017735          1088   3593#
MULDEV  017552          1016   3573#
MULTD   001272           673#   842*  1023*  1026*  1029   2177   2187*
MVECT   017474           985   3565#
NUMONE  013656          3006   3010*  3017   3026#
NUMTWO  013660          3009   3013   3016*  3027#
ONCE    002054           878    907    918#
OPTR    001432           794#  2952   2954   2955*  3122*
PC     =000007           497#   891*   982*  1146*  1161*  1176*  1191*  1205*  1215*  1226*  1232*  1251*  1261*
                        1272*  1278*  1306*  1321*  1330*  1344*  1350*  1356*  1360*  1375*  1381*  1389*  1396*
                        1411*  1418*  1425*  1431*  1445*  1451*  1472*  1483*  1487*  1500*  1506*  1520*  1528*
                        1531*  1540*  1557*  1566*  1574*  1585*  1588*  1597*  1614*  1623*  1631*  1642*  1645*
                        1654*  1671*  1680*  1688*  1699*  1702*  1711*  1727*  1735*  1742*  1797*  1825*  1878*
                        1910*  1963*  2002*  2003*  2011*  2021*  2072*  2111*  2112*  2120*  2129*  2161*  2212*
                        2232*  2239*  2242*  2252*  2257   2353*  2405*  2757*  2764*  2771*  2785*  2797*  2930
                        3000*  3023*  3025*  3048*  3066*  3076*  3093*  3104*  3113*  3124*  3125*  3132*  3143*
                        3152*  3161*  3172*  3186*  3193*  3200*  3207*  3241*  3273*
PCMSG   021313           876   3728#
PIRQ   = 177772          483#
PIRQVE= 000240           577#
PLEVEL  020021          1099   3602#
PRG2    006330           458   1759#
PRG2A   006370          1766#  1831
PRG2B   006510          1808#
PRG3    006570           459   1841#
PRG3A   006630          1848#  1916
PRG3B   006750          1885#
PRG4    007040           460   1926#
PRG4A   007100          1933#  2025
PRG4B   007232          1979#
PRG4C   007244          1987#  2028
PRG5    007432           461   2035#
PRG5A   007472          2042#  2134
PRG5B   007662          2101#  2131
PRG5C   007612          2082#  2137
PRIME   014220          1531   1588   1645   1702   3117#
PROG2M  016664          1764   3494#
PROG3M  016730          1846   3500#
PROG4M  016774          1931   3506#
PROG5M  017040          2040   3512#
PR0    = 000000           500#
PR1    = 000040           501#
PR2    = 000100           502#
PR3    = 000140           503#
PR4    = 000200           504#
PR5    = 000240           505#
PR6    = 000300           506#
PR7    = 000340           507#
PS     = 177776           480#   481
PSW    = 177776           481#  1299   2946   2964   2970   3082   3095   3100   3201
PWRVEC= 000024           572#   819*   820*  2898*  2899*  2909*  2914*  2926*  2927*
```

```
RDB     017153          2013    2121    3525#
RDCHR = 104406          2611    2889#
RDDEC = 104411           933    1989    2084    2892#
RDLIN = 104407          2684    2809    2890#
RDOCT = 104410           944     963     986    1018    1044    1090    1100    1769    1810    1918    1851    1891    1899
                        1936    1975    1981    2045    2094    2891#
REDO1   006560          1774    1776    1779    1786    1802    1830#
REDO2   007030          1856    1858    1861    1868    1883    1915#
REDO3   007412          1941    1943    1946    1953    1968    2024#
REDO3A  007422          1993    1995    2026#
REDO4   010034          2050    2052    2055    2062    2077    2133#
REDO4A  010044          2088    2090    2135#
RESTRT  001772           896#    2189    2258    2931    3298
RESVEC= 000010           567#     899*    900#
RETRY   002346           937     939    1007#
RETRY0  002356           968     970     973     977    1010#
RETRY1  002366           991     993     996    1000    1013#
RETRY2  002750          1049    1051    1054    1058    1129#
RETRY3  002760          1104    1106    1132#
RFLG0   001424           791#    1534    1591    1648    1705    2966*   3120*
RFLG1   001426           792#    1536    1593    1650    1707    2973*   3121*
RINT    013402          2962#    3109
ROTADD  001276           684#    1035*   1063*   1065    1068*   1075    1076*   2199*   2204    2225*
RSTALL  020646          1971     2093    3674#
RSVERR  015044           899     3284#
RTRY    001430           793#    1529*   1577*   1578    1586*   1634*   1635    1643*   1691*   1692    1700*   1745*   1746
R0      =%000000         488#     935*    936     938     940     948*    951*    953*    966*    967     969     971     975
                         978     989*    990     992     994     998    1001    1020*   1021    1047*   1048    1050    1052
                        1056    1059    1102*   1103    1105    1107    1991*   1992    1994    1996    2086*   2087    2089
                        2091    2249*   2252    2377    2378*   2379*   2386*   2387*   2388*   2389*   2390*   2391    2396
                        2401*   2403*   2407    2409    2507    2517*   2521    2537    2538    2551*   2681    2685*   2686
                        2689    2709*   2712*   2740    2741*   2742    2745*   2806    2810*   2811    2814    2816    2817
                        2841*   2845*   2859    2860*   2861    2862*   2863*   2864*   2865*   2900    2925*   3006*   3007*
                        3008*   3009*   3010    3011*   3012*   3013*   3014*   3015*   3016    3017*   3018*   3022    3087*
                        3088    3096*   3098    3292*   3293
R1      =%000001         489#     949*    951     952*    954*    955    1303*   1323*   1553*   1562*   1563    1571*   1610*
                        1619*   1620    1628*   1667*   1676*   1677    1685*   1723*   1731*   1732    1739*   2508    2521*
                        2522    2526    2550*   2682    2687*   2695*   2697*   2699*   2702*   2705    2708*   2807    2817*
                        2819    2821    2831*   2832    2840*   2901    2924*   3083*   3089    3097*   3099    3195*   3196
                        3198    3205    3211*
R2      =%000002         490#     950*    953     954    1143*   1144    1158*   1159    1173*   1174    1188*   1189    1202*
                        1203    1212*   1213    1222*   1223*   1224    1229*   1230    1244*   1245*   1257*   1259    1268*
                        1269*   1270    1275*   1276    1290*   1291*   1292*   1293*   1294*   1300*   1301*   1303    1304
                        1309    1313*   1314    1315*   1319    1324*   1339*   1340*   1341*   1348    1353*   1353    1369*
                        1370*   1372*   1379    1386*   1394    1405*   1406*   1408*   1416    1422*   1429    1440*   1442*
                        1443    1448*   1449    1465*   1466*   1467*   1473*   1478*   1481    1485    1496*   1497*   1498
                        1503*   1504    1515*   1516*   1518    1554*   1555    1563*   1564    1572*   1611*   1612    1620*
                        1621    1629*   1668*   1669    1677*   1678    1686*   1724*   1725    1732*   1733    1740*   1772*
                        1773    1775    1777    1781    1787    1793    1854*   1855    1857    1859    1863    1969    1874
                        1902*   1905*   1939*   1940    1942    1944    1948    1954    1959    2048*   2049    2051    2053
                        2057    2063    2068    2509    2520*   2524*   2527    2534*   2535*   2536    2541*   2549*   2683
                        2688*   2696*   2698*   2700*   2706    2707*   2809    2813*   2816*   2835    2839*   2902    2923*
                        3084    3090    3103    3192*   3196    3203    3210*   3236*   3237    3240*
R3      =%000003         491#    1304*   1325*   1326*   1328    1555*    564*   1573*   1612*   1621*   1630*   1669*   1679*
                        1687*   1725*   1733*   1741*   1781*   1782*    .794   1863*   1864*   1866    1948*   1949*   1951
                        2057*   2058*   2060    2449    2458*   2464*   2465*   2468*   2473*   2474*   2475    2484*   2510
```

```
                    2518*  2519*  2533*  2536*  2545*  2546*  2548*  2606   2608*  2609   2612*  2613   2620*
                    2621   2623   2631   2635   2637*  2643   2645   2647*  2650*  2903   2922*  3084*  3091
                    3205*  3206*
R4      =%000004     492#  1201*  1203   1211*  1213   1221*  1224   1228*  1230   1243*  1250*  1259   1267*
                    1270   1274*  1276   1305*  1318*  1327*  1328   1338*  1347*  1348   1352*  1358   1371*
                    1378*  1379   1385*  1393*  1394   1407*  1415*  1416   1421*  1428*  1429   1441*  1443
                    1447*  1449   1464*  1485   1495*  1498   1502*  1504   1514*  1518   1556*  1565*  1613*
                    1622*  1670*  1679*  1726*  1734*  2450   2452*  2453*  2454*  2455   2456*  2470   2472*
                    2480*  2483*  2904   2921*  3092   3203*  3204*
R5      =%000005     493#  1239*  1240*  1241*  1242*  1248*  1284*  1285*  1286*  1288*  1297*  1459*  1460*
                    1461*  1462*  1470*  2451   2457*  2459*  2461*  2462*  2463*  2464   2482*  2511   2513*
                    2515*  2522*  2526*  2541   2547*  2905   2920*  3108*  3109*  3110*  3111*  3112*  3139*
                    3140*  3141   3147*  3148*  3149*  3150   3156*  3157*  3158*  3159   3166*  3167*  3169
                    3177*  3179*  3181*  3184   3220*  3224*  3228*  3232*  3234*  3235*  3236   3256*  3260*
                    3264*  3268*  3270*  3271*  3272
R6      =%000006     494#   807*   808*   809    2986   2988*  3046*
R7      =%000007     495#
SELCAR   020367     1808   1889   1979   3644#
SELDLY   020475     1815   1896   3656#
SP      =%000006     496#   811*   828*   836*   840    896*   935    945    966    989    1020   1047   1102
                    1141*  1149   1150   1156*  1164   1165   1171*  1179   1180   1186*  1194   1195   1302
                    1759*  1772   1791*  1800   1801   1803   1812   1820   1841*  1854   1872*  1881   1882
                    1884   1893   1901   1926*  1939   1957*  1966   1967   1969   1977   1983   1991   2035*
                    2048   2066*  2075   2076   2078   2086   2096   2246*  2283*  2286   2288   2289   2317
                    2318   2322*  2348   2361*  2364*  2377*  2382*  2403   2407*  2441*  2442   2443   2444*
                    2449*  2450*  2451*  2457   2482   2483   2484   2485*  2486*  2507*  2508*  2509*  2510*
                    2511*  2512*  2513   2516*  2529   2531*  2533   2543   2545   2547   2548   2549   2550
                    2551   2553*  2554*  2578*  2579*  2582*  2583*  2584   2588*  2589*  2590   2593   2595
                    2597*  2606*  2607*  2612   2615   2619*  2626   2630*  2649   2650   2651*  2652*  2653*
                    2679*  2680*  2681*  2682*  2683*  2685   2689*  2691   2693   2701*  2702   2704   2705*
                    2707   2708   2709   2711   2740*  2741   2742*  2744   2745   2746*  2748   2750   2752
                    2758   2760*  2762*  2770*  2774   2778   2779   2783   2804*  2805*  2806*  2807*  2808*
                    2810   2812*  2823   2825*  2826*  2827*  2828*  2829*  2832*  2837*  2838*  2839   2840
                    2841   2844   2859*  2860   2870*  2871*  2900*  2901*  2902*  2903*  2904*  2905*  2906*
                    2907   2915*  2919   2920   2921   2922   2923   2924   2925   2930*  2947   2965   2971
                    2991*  2992*  2995   3085   3101   3202   3291   3294   3296
STACK = 001100       471#   811    896    1759   1841   1926   2035
STALL   013570      2002   2111   3006#
STKLMT= 177774       482#
STLMSK  001306       701#  3018
STMES   016635       873    909   3489#
SUERR1  014064      1306   1330   1557   1566   1574   1614   1623   1631   1671   1680   1688   1727   1735
                    1742   3087#  3207
SUERT1  014116      1146   1161   1176   1191   3095#
SUERT2  014144      1797   1878   1963   2072   3100#
SUER2   014036      1205   1215   1226   1232   1251   1261   1272   1278   1321   1344   1350   1356   1360
                    1375   1381   1389   1396   1411   1418   1425   1431   1445   1451   1472   1483   1487
                    1500   1506   1520   3082#
SUVEC   014172      1528   1585   1642   1699   3108#
SWR     001140       606#   809    830*   832    838*   853    913   1336   1367   1403   1438   1457   1493
                    2278   2292   2294   2300   2307   2344   2351   2356   2359   2906   2919*
SWREG   000176       455#   838
SWO   = 000001       535#   913
SWOO  = 000001       525#   535
SWO1  = 000002       524#   534
SWO2  = 000004       523#   533
```

DZDLCA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
SW03   = 000010            522*    532
SW04   = 000020            521*    531
SW05   = 000040            520*    530
SW06   = 000100            519*    529
SW07   = 000200            518*    528
SW08   = 000400            517*    527
SW09   = 001000            516*    526
SW1    = 000002            534*
SW10   = 002000            515*
SW11   = 004000            514*
SW12   = 010000            513*   1336    1367    1403    1438    1457    1493
SW13   = 020000            512*
SW14   = 040000            511*
SW15   = 100000            510*
SW2    = 000004            533*
SW3    = 000010            532*
SW4    = 000020            531*
SW5    = 000040            530*
SW6    = 000100            529*
SW7    = 000200            528*
SW8    = 000400            527*    953
SW9    = 001000            526*
TABFLG  001256             653*    844*    894*    906     911*   2188*
TBITVE= 000014             568*
TCONT   013706            3035    3039*
TIMERX  013662            2011    2120    3031*
TIMETX  013700            2003    2112    3037*
TIMR1   001440             797*   1542*   1599*   1656*   1713*   3126*
TIMR2   001442             798*   1544*   1601*   1658*   1715*   3127*
TKVEC  = 000060            575*
TPVEC  = 000064            576*
TRAPVE= 000034             574*    817*    818*   1760*   1761*   1842*   1843*   1927*   1928*   2036*   2037*
TRPCOM  015070            3280    3287    3291*
TRTVEC= 000014             569*
TST1    002770             922    1128    1140*   2220
TST10   003340            1231    1237*
TST11   003464            1255    1260    1266*
TST12   003534            1277    1283*
TST13   004002            1312    1329    1335*
TST14   004132            1337    1359    1366*
TST15   004260            1368    1395    1402*
TST16   004406            1404    1430    1437*
TST17   004466            1439    1450    1456*
TST2    003036            1155*
TST20   004630            1458    1477    1486    1492*
TST21   004712            1494    1505    1513*
TST22   004744            1519    1526*
TST23   005236            1541    1552    1561    1570    1583*
TST24   005530            1598    1609    1618    1627    1640*
TST25   006022            1655    1666    1675    1684    1697*
TST3    003104            1170*
TST4    003152            1185*
TST5    003220            1200*
TST6    003242            1204    1210*
TST7    003266            1214    1220*
TXPDS  = 104405            2247    2886*
```

```
TYPE   = 104401          873   874   876   903   931   942   960   985  1007  1010  1013  1016  1054
                        1088  1099  1129  1132  1548  1605  1662  1719  1764  1768  1808  1818  1834
                        1846  1848  1889  1896  1915  1931  1933  1971  1979  1967  2005  2013  2034
                        2026  2040  2042  2082  2093  2114  2121  2133  2135  2139  2146  2148  2346
                        2354  2376  2393  2395  2398  2400  2404  2411  2476  2552  2618  2624  2682
                        2633  2638  2639  2641  2644  2648  2713  2715  2753  2848  2848  2882  2888
TYPOC  = 104402         2384  2408  2883#
TYPON  = 104404         2885#
TYPOS  = 104403         2884#
UPMASK   014730         3066  3254#
VECT     002264          985  1015
XDB      017104         2005  2114  3518#
XFLGD    001422          790  1532  1589  1646  1703  2948  3119
XINT     013314         2944  3111
XMSG1    016326         1548  3453#
XMSG2    016405         1605  3461#
XMSG3    016466         1662  3470#
XMSG4    016551         1719  3479#
$AUTOB   001134          603#
$BDADR   001122          598#
$BDDAT   001126          600#
$BELL    001246          644  2346  2367
$CHARC   012670         2755  2765  2772  2781  2796#
$CKSWR=  ******  U      2889
$CMTAG   001100          586#   806   807   821   822
$CM1   = 000010          618#  619#  620#  621#  622#  623#  624#  625#  626#
$CM2   = 000020          618#  619#  620#  621#  622#  623#  624#  625#  626#
$CM3   = 000010          616#  618
$CM4   = 000020          626#  627#  628#  629#  630#  631#  632#  633#  634#  635#  636#  637#  639#
                         639#  640#  641#  642#
$CNTLG   012265         2659#
$CNTLU   012260         2633  2658#
$CRLF    001253          646#  2354  2367  2376  2395  2400  2404  2638  2658  2718  2754  2789  2850
$DBLK    011664         2518  2552  2560#
$DDAGN   010436         2241  2250  2256#
$DTBL    011654         2521  2556#
$ENDAD   010426         2252#
$ENDCT   010374         2243#
$ENDMG   010445         2245  2260#
$ENULL   010442         2248  2259#
$EOP     010130         1712  1722  1730  1738  1748  2171#
$EOPCT   010366         2240#  2244
$ERFLG   001103          589#  2268  2296  2298  2304#  2325  2341#  2367
$ERMAX   001115          595#   823#  2298  2320#  2325
$ERROR   010732          815  1762  1844  1929  2038  2340#
$ERRPC   001116          596#  2348#  2349#  2350  2367  2382  3324  3346  3368  3411  3424  3436  3449
$ERRTB   001310          723#  2390
$ERRTY   011066         2353  2375#
$ERTTL   001112          593#  2347#  2367
$ESCAP   001244          643#   822#  1147#  1162#  1177#  1192#  1252#  1307#  1474#  1549#  1558#  1567#  1575#
                        1606#  1615#  1624#  1632#  1663#  1672#  1681#  1689#  1720#  1729#  1736#  1743#  1798#
                        1879#  1964#  2073#  2319#  2362  2364  2367  3208#  3295#
$FILLC   001156          614#  2758  2789
$FILLS   001155          613#  2789
$GDADR   001120          597#
$GDDAT   001124          599#
```

```
$GET42  010416          2249#
$GTSWR= ****** U        2898
$HD     = 000001         427     428
$HIOCT  012452          2706*   2717#
$ICNT   001104           590#   2311*   2312    2314*   2324
$ILLUP  013276          2898    2914    2933#
$INTAG  001135           604#
$ITEMB  001114           594#   2350*   2367    2379
$LF     001254           647#   2367    2648    2658    2718    2789    2850
$LPADR  001106           591#    824*   2302*   2317*   2322    2324
$LPERR  001110           592#    825*   2302    2318*   2324    2361
$MAIL = ****** U         842    2317    2356    2742
$MNEW   012303          2662#
$MSWR   012272          2660#
$MXCNT  010730          2315    2324#
$NULL   001154           612#   2760    2789
$NWTST= 000001          1137#   1152#   1167#   1182#   1197#   1207#   1217#   1234#   1263#   1280#   1332#   1363#   1399#
                        1434#   1453#   1489#   1510#   1523#   1580#   1637#   1694#
$OCNT   011444          2448*   2477*   2490#
$OMODE  011446          2443*   2447*   2452    2455*   2466*   2492#
$OVER   010714          2279    2295    2303    2313    2321#
$PASS   001100           587#   2237*   2238*   2246    2259    2309    2325
$POWER  013304          2929    2936#
$PWRAD  013272          2931#
$PWRDN  013132           819    2898#   2926
$PWRMG  013266          2929#
$PWRUP  013204          2908    2914#
$QUES   001252           645#   1007    1010    1013    1129    1132    1830    1915    2024    2026    2133    2135    2367
                        2641    2658    2715    2718    2789    2848    2850
$RDCHR  011674          2578#   2889
$RDDEC  012674          2804#   2892
$RDLIN  012014          2606#   2890
$RDOCT  012314          2679#   2891
$RDSZ = 000010          2599#
$REGAD  001160           616#
$REG0   001162           618#   3088*   3098*   3293*   3324    3346    3368    3387    3411
$REG1   001164           619#   3058*   3060*   3089*   3099*   3324    3346    3368    3411    3436    3449
$REG2   001166           620#   3056*   3058    3090*   3103*   3324    3346    3368    3411    3424    3436    3449
$REG3   001170           621#   3062*   3091*   3346    3368    3411    3436    3449
$REG4   001172           622#   3071*   3092*   3346    3368    3449
$REG5   001174           623#
$REG6   001176           624#   1302*   2947*   2965*   2971*   3085*   3086*   3101*   3102*   3202*   3291*   3324    3346
                        3368    3387    3411    3424
$REG7   001200           625#   1361*   3296*   3387
$RTNAD  010440          2258#
$R2A  = ****** U        2893
$SAVRE= ****** U        2893
$SAVR6  013302          2907*   2915    2916*   2917*   2935#
$SCOPE  010462           813    2277#
$SETUP= 000017           803#    812     813     817     817     819     821     822     824    2235    2278    2341    2359
                        2366    2567    2664
$STUP = 177777           803#
$SVLAD  010666          2287    2316#
$SWR  = 167400           410#    427     435     436     437     438     439     440     441     442     642     643     644
                         821     822     824     825    1141    1156    1171    1186    1201    1211    1221    1239    1267
                        1284    1336    1367    1403    1438    1457    1493    1514    1527    1584    1641    1698    2168
```

```
COMMEN    578#
ENDCOM    578#
EOPBEG    430#  2172
ERROR     472#   875  1148  1163  1178  1193  1206  1216  1227  1233  1253  1262  1273  1279  1308
         1322  1331  1345  1351  1357  1362  1376  1382  1390  1397  1412  1419  1426  1432  1446
         1452  1475  1484  1488  1501  1507  1521  1550  1559  1568  1576  1607  1616  1625  1633
         1664  1673  1682  1690  1721  1729  1737  1744  1799  1880  1965  2006  2014  2074  2115
         2122  3073  3075  3209  3297
ESCAPE    578#
GETPRI    578#
GETSWR    578#
MORETA    429#   649
MULT      579#
NEWTST    578#  1137  1152  1167  1182  1197  1207  1217  1234  1263  1280  1332  1363  1399  1434
         1453  1489  1510  1523  1580  1637  1694
POP       414#   578#  2547  2707  2839  2919  2920
PUSH      414#   578#  2506  2681  2806  2900  2906
REPORT    578#
SCOPE     473#  1140  1155  1170  1185  1200  1210  1220  1237  1266  1283  1335  1366  1402  1437
         1456  1492  1513  1526  1583  1640  1697  2176
SETPRI    578#
SETTRA   2874#  2883  2884  2885  2886  2889  2890  2891  2892
SETUP     412#   578#   805
SKIP      578#  1145  1160  1175  1190  1204  1214  1225  1231  1254  1260  1271  1277  1312  1320
         1329  1337  1343  1349  1355  1359  1368  1374  1380  1388  1395  1404  1410  1417  1424
         1430  1439  1444  1450  1458  1476  1482  1486  1494  1499  1505  1519  1541  1551  1560
         1569  1598  1608  1617  1626  1655  1665  1674  1683
SLASH     578#
SPACE     578#
STARS     414#   578#   581   648   781   783  1137  1139  1152  1154  1167  1169  1182  1184  1197
         1199  1207  1209  1217  1219  1234  1236  1263  1265  1280  1282  1332  1334  1363  1365
         1399  1401  1434  1436  1453  1455  1489  1491  1510  1512  1523  1525  1580  1582  1637
         1639  1694  1696  2165  2234  2265  2326  2370  2418  2496  2564  2570  2599  2667  2721
         2792  2853  2896  2912  2940  2942  2958  2960  3078  3080  3302  3304
SWRSU     578#   826#
TRMTRP   2874#
TYPBIN    578#
TYPDEC    578#  2246
TYPNAM    578#
TYPNUM    578#
TYPOCS    578#
TYPOCT    578#  2382  2406
TYPTXT    578#
$$CMRE    579#   618   619   620   621   622   623   624   625
$$CMTM    579#   626   627   628   629   630   631   632   633   634   635   636   637   638   639
         640   641
$$ESCA    578#
$$NEWT    578#  1137  1152  1167  1182  1197  1207  1217  1234  1263  1280  1332  1363  1399  1434
         1453  1489  1510  1523  1580  1637  1694
$$SET    2874#  2883  2884  2885  2886  2889  2890  2891  2892
$$SKIP    578#  1204  1214  1231  1255  1260  1277  1312  1329  1337  1359  1368  1395  1404  1430
         1439  1450  1458  1477  1486  1494  1505  1519  1541  1552  1561  1570  1598  1609  1618
         1627  1655  1666  1675  1684
.EQUAT    412#   468
.HEADE    411#   417
.SETUP    411#   803
```

L07


```
.SWRHI    412#    431
.SWRLO    412#    443#    444
.$CATC    412#    447
.$CMTA    411#    579
.$EOP     412#   2163
.$ERRO    411#   2326
.$ERRT    413#   2368
.$POWE    413#   2894
.$RDDE    414#   2790
.$RDOC    414#   2665
.$READ    414#   2562
.$SCOP    413#   2263
.$TRAP    413#   2851
.$TYPD    413#   2494
.$TYPE    413#   2719
.$TYPO    413#   2416
```

```
ADD      953  954 1070 1902 2149 2153 2157 2192 2195 2390 2444 2454 2526 2702 2746
        2829 2832 2988 3009 3013 3046 3086 3102 3224 3228 3232 3260 3264 3268
ASL     1113 1114 1115 1116 1117 2387 2388 2389 2695 2697 2699 2825 2827 2828 2863
ASLB    2531
BCC     2532
BCS     1069 2201
BEQ      854 1204 1214 1225 1231 1260 1271 1277 1320 1329 1349 1355 1359 1380 1395
        1417 1430 1444 1450 1486 1499 1505 1519 2001 2110 2178 2206 2250 2293 2295
        2297 2301 2310 2342 2345 2360 2363 2392 2397 2410 2471 2627 2690 2749 2784
        2818 2836 2953 2978 2990 3021 3041 3070 3169 3223 3227 3231 3259 3263 3267
BGE     2313
BGT     2241 2478 2540 2596 2692 2820
BHI      937  968  991 1049 1072 1104 1774 1856 1941 1993 2050 2088 2299
BIC     1122 1229 1257 1275 1300 1301 1313 1315 1326 1353 1386 1422 1448 1503 1546
        1547 1603 1604 1660 1661 1717 1718 2103 2238 2468 2583 2589 2597 2701 2949
        2950 2974 2975 3018 3190 3191 3204 3206
BICB    1782 1864 1949 2058 3236 3272
BIS     1065 1075 1223 1245 1269 1292 1293 1341 1372 1408 1442 1466 1467 1497 1516
        1905 2007 2116 2473 2474 2534 2535 3130 3131
BISB    2379
BIT      853  913  975  998 1056 1336 1342 1354 1367 1373 1387 1403 1409 1423 1438
        1457 1481 1493 2204 2278 2292 2300 2307 2344 2351 2359 2823 3064
BITB     971  994 1052 1777 1859 1944 2053 2772
BLO      939  970  993 1051 1106 1776 1858 1943  995 2052 2090 2622
BLOS    2610
BLT      956 2479 2523 2539 2594 2694 2763 2822
BMI     1031 2530 2945 2963
BNE      810  833  907  917  946  973  977  996 1000 1022 1054 1058 1247 1249 1296
        1298 1337 1343 1368 1374 1388 1404 1410 1424 1439 1458 1469 1471 1482 1494
        1533 1535 1537 1539 1543 1545 1579 1590 1592 1594 1596 1600 1602 1636 1647
        1649 1651 1653 1657 1659 1693 1704 1706 1708 1710 1714 1716 1747 1779 1786
        1861 1868 1946 1953 2055 2062 2181 2279 2308 2352 2380 2402 2469 2528 2585
        2591 2614 2616 2632 2636 2646 2743 2751 2759 2773 2780 2815 2824 2918 2993
        2998 3065 3142 3151 3160 3185 3197 3199 3239
BPL     2357 2467 2514 2544 2581 2587 2737 2777 2969 3045
BR       835  878 1005 1025 1145 1160 1175 1190 1255 1312 1477 1541 1552 1561 1570
        1598 1609 1618 1627 1655 1666 1675 1684 1796 1802 1829 1877 1883 1914 1962
        1968 2023 2071 2077 2131 2281 2287 2290 2303 2306 2385 2412 2445 2460 2481
        2525 2542 2592 2625 2634 2640 2642 2703 2716 2739 2756 2766 2775 2782 2834
        2849 2910 2934 2951 2967 2976 3035 3074 3171 3212 3280 3287
BVS     2830 2833
CLC     1066 2198
CLR      808  821  822  842  844  894  918  919  952 1023 1032 1035 1064 1124 1125
        1201 1238 1242 1274 1287 1288 1291 1318 1340 1352 1370 1393 1406 1428 1447
        1462 1463 1464 1473 1473 1529 1586 1643 1700 2187 2188 2214 2216 2217 2235
        2236 2305 2319 2378 2458 2517 2520 2607 2630 2687 2688 2812 2813 2916 3039
        3117 3118 3119 3120 3121 3126 3140 3156 3178 3220 3256
CLRB    1289 2304 2546 2637 2647 2712 2755 2781 2845 3148
CMP      809  832  936  938  955  967  969  990  992 1048 1050 1071 1103 1105 1149
        1164 1179 1194 1203 1213 1224 1230 1259 1270 1276 1328 1348 1358 1373 1394
        1416 1429 1443 1449 1485 1498 1504 1518 1538 1578 1595 1635 1652 1692 1709
        1746 1773 1775 1800 1855 1857 1881 1940 1942 1966 1992 1994 2049 2051 2075
        2087 2089 2288 2312 2538 2584 2590 2593 2595 2609 2621 2952 2977 3068 3141
        3150 3159 3168 3184 3198 3222 3226 3230 3237 3258 3262 3266
CMPB    1784 1866 1951 2060 2294 2298 2613 2631 2635 2645 2691 2693 2748 2750 2758
        2779 2783 2814 2819 2821 3196
```

```
COM     1256  1316  1479  2948  2966  2973  3235  3271
COMB     911  1026  3180
DEC     1248  1297  1470  1542  1544  1599  1601  1656  1658  1713  1715  2239  2386  2620  2992
        2996
DECB    2466  2477  2762  2765  3170
EMT      472
HALT     453   877  2186  2358  2738  2909  2933
INC     1577  1634  1691  1745  2102  2237  2311  2347  2472  2480  2524  2917  2955  2980  3040
        3158  3210  3211
INCB    2316  2341  2785  3182
IOT      473
JMP      457   458   459   460   461   922   947  1009  1012  1015  1037  1081  1092  1128  1131
        1134  1712  1722  1730  1738  1748  1831  1916  2025  2028  2134  2137  2189  2220  2257
        3298
JSR      891   982  1146  1161  1176  1191  1205  1215  1226  1232  1251  1261  1272  1278  1306
        1321  1330  1344  1350  1356  1360  1375  1381  1389  1396  1411  1418  1425  1431  1445
        1451  1472  1483  1487  1500  1506  1520  1528  1531  1540  1557  1566  1574  1585  1588
        1597  1614  1623  1631  1642  1645  1654  1671  1680  1688  1699  1702  1711  1727  1735
        1742  1797  1825  1878  1910  1963  2002  2003  2011  2021  2072  2111  2112  2120  2129
        2212  2232  2252  2353  2757  2764  2771  3023  3066  3124  3125  3193  3207
MOV      807   811   813   814   815   816   817   818   819   820   824   825   828   829   830
         831   836   838   839   840   845   851   890   893   896   897   898   899   900   935
         940   948   949   950   951   966   978   981   984   989  1001  1003  1004  1020  1047
        1059  1063  1076  1078  1086  1102  1107  1118  1141  1142  1143  1147  1150  1156  1157
        1158  1162  1165  1171  1172  1173  1177  1180  1186  1187  1188  1192  1195  1202  1211
        1212  1221  1222  1228  1239  1240  1241  1243  1244  1250  1252  1267  1268  1284  1285
        1286  1290  1299  1302  1303  1304  1305  1307  1323  1324  1325  1327  1338  1339  1347
        1361  1369  1371  1378  1385  1405  1407  1415  1421  1440  1441  1459  1460  1461  1465
        1474  1495  1496  1502  1514  1515  1527  1530  1549  1553  1554  1555  1556  1558  1562
        1563  1564  1565  1567  1571  1572  1573  1575  1584  1587  1606  1610  1611  1612  1613
        1615  1619  1620  1621  1622  1624  1628  1629  1630  1632  1641  1644  1663  1667  1668
        1669  1670  1672  1676  1677  1678  1679  1681  1685  1686  1687  1689  1698  1701  1720
        1723  1724  1725  1726  1728  1731  1732  1733  1734  1736  1739  1740  1741  1743  1759
        1760  1761  1762  1763  1772  1781  1787  1791  1792  1798  1801  1803  1812  1820  1841
        1842  1843  1844  1845  1854  1863  1869  1872  1873  1879  1882  1884  1893  1901  1936
        1927  1928  1929  1930  1939  1948  1954  1957  1958  1964  1967  1969  1977  1983  1991
        1996  1998  2009  2015  2019  2035  2036  2037  2038  2039  2048  2057  2063  2066  2067
        2073  2076  2078  2086  2091  2096  2100  2101  2104  2105  2107  2118  2123  2127  2147
        2151  2155  2159  2208  2210  2225  2227  2228  2229  2230  2242  2246  2249  2283  2284
        2286  2289  2302  2314  2315  2317  2318  2321  2322  2343  2348  2361  2364  2377  2382
        2391  2396  2401  2403  2407  2441  2449  2450  2451  2457  2464  2482  2483  2484  2485
        2486  2507  2508  2509  2510  2511  2512  2513  2518  2521  2541  2547  2548  2549  2550
        2551  2553  2554  2578  2579  2606  2608  2619  2650  2651  2652  2653  2679  2680  2681
        2682  2683  2685  2686  2705  2706  2707  2708  2709  2740  2741  2745  2760  2804  2805
        2806  2807  2808  2810  2811  2826  2838  2839  2840  2841  2859  2860  2864  2870  2871
        2898  2899  2900  2901  2902  2903  2904  2905  2306  2907  2908  2914  2915  2919  2920
        2921  2922  2923  2924  2925  2926  2927  2930  2946  2947  2964  2965  2970  2971  2972
        2986  2991  3006  3010  3016  3017  3022  3031  3037  3054  3056  3058  3062  3071  3082
        3083  3084  3085  3088  3089  3090  3091  3092  3095  3097  3098  3099  3100  3101  3103
        3108  3109  3110  3111  3112  3122  3123  3127  3139  3147  3166  3177  3192  3195  3201
        3202  3208  3234  3240  3254  3270  3291  3293  3294  3295  3296
MOVB     823  1294  1314  1821  1823  1906  1908  2320  2350  2442  2443  2446  2447  2448  2452
        2455  2456  2475  2516  2519  2533  2536  2545  2582  2588  2612  2617  2623  2628  2643
        2689  2742  2770  2778  2816  2817  2862  2954  2979  3087  3096  3149  3157  3165  3167
        3179  3181  3183  3203  3205  3277  3278  3279  3284  3285  3286  3292
NEG     2453  2515  2837
```

B08

```
MAINDEC-11-DZDLC-A      MACY11 27(732)  20-SEP-76  09:19  PAGE 95
DZDLCA.P11       CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

NOP        804   2211   2231   2253   2254   2255
RESET     1517   2251
ROL       1069   2199   2459   2461   2462   2463   2465   2696   2698   2700   3007   3008   3011   3012   3014
          3016
RTI        937   1258   1317   1480   2323   2366   2487   2555   2599   2654   2710   2747   2842   2872   2932
          2956   2991
RTS       2161   2405   2797   2865   3000   3025   3049   3076   3093   3104   3113   3132   3143   3152   3161
          3170   3186   3200   3241   3273
SUB       1120   1999   2017   2108   2125   2349   2522   2831   3033   3060
TRAP      2874   2883   2884   2885   2886   2889   2890   2891   2892
TST        920    921    945   1021   1126   1127   1144   1159   1174   1189   1246   1295   1309   1319   1468
          1532   1534   1536   1599   1591   1593   1646   1648   1650   1703   1705   1707   1733   1974   1959
          2000   2068   2109   2180   2218   2219   2285   2309   2356   2362   2409   2470   2527   2537   2615
          2626   2649   2704   2711   2744   2752   2774   2835   2844   2861   2968   2989   2995   3128   3129
TSTB       906   1029   2177   2296   2529   2543   2580   2586   2736   2776   2944   2962   3044
.ASCII     643    646   3543   3610   3620   3684   3692   3700   3706   3712   3728
.ASCIZ     644    647   2260   2413   2658   2659   2660   2662   2936   3308   3315   3330   3334   3352   3356
          3374   3381   3392   3400   3417   3429   3441   3453   3461   3470   3479   3489   3494   3500   3506
          3513   3519   3525   3534   3550   3556   3565   3573   3582   3593   3602   3626   3635   3644   3656
          3666   3674   3720   3729
.BLKB     2657   3734   3737
.BLKW     2560
.BYTE      588    589    594    595    603    604    612    613    614    615   2259   2488   2489   2490   2491
          2655   2656
.DSABL    2557
.ENABL     415   2565
.END      3743
.ENDC      422    439    441    442    443    458    472    564    578    582    586    588    616    626    642
           643    644    645    649    709    782    784    803    811    912    915    917    819    921    822
           824    826    842   1138   1139   1140   1141   1146   1153   1154   1155   1156   1161   1168   1169
          1170   1171   1175   1183   1184   1185   1186   1191   1198   1199   1200   1201   1205   1208   1209
          1210   1211   1215   1218   1219   1220   1221   1226   1232   1235   1236   1237   1238   1256   1261
          1264   1265   1266   1267   1272   1278   1281   1282   1283   1284   1313   1321   1330   1333   1334
          1335   1336   1338   1344   1350   1356   1360   1364   1365   1366   1367   1369   1375   1381   1399
          1396   1400   1401   1402   1403   1405   1411   1419   1425   1431   1435   1436   1437   1438   1440
          1445   1451   1454   1455   1456   1457   1459   1478   1483   1487   1490   1491   1492   1493   1495
          1500   1506   1511   1512   1513   1514   1520   1524   1525   1526   1527   1528   1542   1553   1562
          1571   1581   1582   1583   1584   1585   1599   1610   1619   1629   1638   1639   1640   1641   1642
          1656   1667   1676   1685   1695   1696   1697   1698   1699   2166   2167   2168   2170   2235   2241
          2244   2245   2249   2251   2257   2259   2260   2263   2266   2269   2274   2278   2280   2291   2294
          2295   2296   2298   2300   2307   2311   2316   2317   2321   2324   2325   2329   2332   2341   2348
          2353   2354   2355   2356   2366   2367   2371   2386   2415   2419   2497   2565   2566   2567   2571
          2599   2600   2608   2610   2613   2641   2658   2664   2669   2674   2718   2722   2742   2793   2954
          2860   2863   2882   2883   2884   2885   2886   2887   2888   2899   2890   2991   2992   2893   2897
          2906   2907   2913   2919   2920   2930   2932   2939   2941   2943   2959   2961   3079   3081   3303
          3305
.EQUIV     472    473    481    526    527    528    529    530    531    532    533    534    535    554    555
           556    557    558    559    560    561    562    563
.EVEN     2414   2938   3323   3345   3367   3386   3410   3423   3435   3449   3731
.IF        418    438    440    441    442    443    456    470    536    564    581    595    597    616    626
           642    643    644    648    649    781    783    803    806    811    913    915    917    819    821
           822    824    842   1137   1139   1141   1145   1152   1154   1156   1160   1167   1169   1171   1175
          1182   1184   1186   1190   1197   1199   1201   1204   1207   1209   1211   1214   1217   1219   1221
          1225   1231   1234   1236   1238   1255   1260   1263   1265   1267   1271   1277   1280   1282   1294
          1312   1320   1329   1332   1334   1336   1337   1343   1349   1355   1359   1363   1365   1367   1369
          1374   1380   1388   1395   1399   1401   1403   1404   1410   1417   1424   1430   1434   1436   1438
```

```
                1439    1444    1450    1453    1455    1457    1458    1477    1482    1486    1489    1491    1493    1494    1499
                1505    1510    1512    1514    1519    1523    1525    1527    1528    1541    1552    1561    1570    1580    1582
                1594    1595    1598    1609    1618    1627    1637    1639    1641    1642    1655    1666    1675    1684    1694
                1696    1698    1699    2165    2166    2167    2168    2169    2170    2172    2234    2240    2243    2245    2249
                2261    2259    2259    2260    2265    2268    2273    2278    2290    2292    2293    2294    2296    2297    2298
                2304    2309    2317    2318    2323    2324    2325    2328    2331    2341    2344    2351    2353    2354    2356
                2359    2366    2367    2370    2385    2401    2418    2496    2564    2566    2567    2570    2571    2599    2607
                2609    2613    2614    2657    2658    2664    2667    2670    2686    2721    2742    2792    2853    2859    2863
                2874    2893    2894    2885    2886    2887    2888    2889    2890    2891    2892    2893    2896    2906    2907
                2912    2919    2920    2928    2930    2932    2936    2940    2942    2958    2960    3078    3080    3302    3304
.IFF             438     441     442     443     470     582     585     587     616     649     782     784     811    1138    1139
                1140    1141    1145    1153    1154    1155    1156    1160    1168    1169    1170    1171    1175    1183    1184
                1185    1186    1190    1198    1199    1200    1201    1205    1208    1209    1210    1211    1215    1218    1219
                1220    1221    1225    1232    1235    1236    1237    1238    1256    1261    1264    1265    1266    1267    1271
                1278    1281    1282    1283    1284    1313    1320    1330    1333    1334    1335    1336    1338    1343    1349
                1355    1360    1364    1365    1366    1367    1369    1374    1380    1388    1396    1400    1401    1402    1403
                1405    1410    1417    1424    1431    1435    1436    1437    1438    1440    1444    1451    1454    1455    1456
                1457    1459    1478    1482    1487    1490    1491    1492    1493    1495    1499    1506    1511    1512    1513
                1514    1520    1524    1525    1526    1527    1528    1542    1553    1562    1571    1581    1582    1583    1584
                1585    1599    1610    1619    1628    1638    1639    1640    1641    1642    1656    1667    1676    1695    1695
                1696    1697    1698    1699    2166    2169    2172    2235    2241    2244    2259    2266    2291    2294    2295
                2298    2324    2329    2331    2344    2366    2367    2371    2386    2415    2419    2497    2565    2567    2571
                2572    2578    2599    2600    2610    2641    2658    2668    2722    2793    2854    2860    2897    2913    2930
                2941    2943    2959    2961    3079    3081    3303    3305
.IFT            2306    2354    2573    2578    2691    2711    2718
.IFTF           2304    2353    2567    2571    2574    2687    2695    2717
.IIF             417     422     427     428     435     436     437     439     442     443     444     453     648     812     915
                 921     822     824     825    2167    2235    2236    2247    2259    2263    2269    2270    2271    2272    2273
                2274    2278    2305    2306    2321    2324    2325    2332    2333    2334    2335    2336    2341    2359    2366
                2367    2383    2408    2565    2649    2658    2664    2718    2789    2950    2882    2883    2884    2885    2896
                2889    2890    2891    2892
.IRP             649     903    1137    1152    1167    1182    1197    1207    1217    1234    1263    1280    1332    1363    1399
                1434    1453    1489    1510    1523    1580    1637    1694    2172    2507    2547    2681    2707    2806    2839
                2900    2906    2919    2920
.LIST            409     442     453     578     616     618     619     620     621     622     623     624     625     626     627
                 628     629     630     631     632     633     634     635     636     637     638     639     640     641     642
                 803     826    1137    1141    1152    1156    1167    1171    1182    1186    1197    1201    1207    1211    1217
                1221    1234    1238    1263    1267    1280    1284    1332    1336    1363    1367    1399    1403    1434    1438
                1453    1457    1489    1493    1510    1514    1523    1527    1580    1584    1637    1641    1694    1698    2235
                2251    2273    2366    2599    2874    2882    2883    2884    2885    2886    2887    2889    2890    2891    2892
                2893
.MACRO           429     430     443     579    2874
.MCALL           411     412     413     414     578     826
.NLIST           408     442     453     578     616     618     619     620     621     622     623     624     625     626     627
                 628     629     630     631     632     633     634     635     636     637     638     539     640     641     642
                 803     826    1137    1141    1152    1156    1167    1171    1182    1186    1197    1201    1207    1211    1217
                1221    1234    1238    1263    1267    1280    1284    1332    1336    1363    1367    1399    1403    1434    1438
                1453    1457    1489    1493    1510    1514    1523    1527    1580    1584    1637    1641    1694    1698    2235
                2251    2273    2366    2599    2874    2882    2883    2884    2885    2886    2887    2889    2890    2891    2892
                2893
.PAGE            579     709    1523    1580    1637    1694
.REM               4
.REPT            453     618     626
.SBTTL           431     447     456     468     579     709     805    1137    1152    1167    1182    1197    1207    1217    1234
                1263    1280    1332    1363    1399    1434    1453    1489    1510    1523    1580    1637    1694    2161    2263
                2326    2368    2416    2494    2562    2665    2719    2790    2851    2874    2894
```

```
MAINDEC-11-DZDLC-A      MACY11 27(732)  20-SEP-76  09:19  PAGE 97
DZDLCA.P11      CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

.TITLE    417
.WORD     453     454     455     464     587     590     591     592     593     596     597     598     599     600     601
          602     605     606     607     616     618     619     620     621     622     623     624     625     626     627
          628     629     630     631     632     633     634     635     636     637     638     639     640     641     653
          657     659     663     666     670     673     677     684     689     693     695    1928    1913    2240    2243
         2258    2394    2399    2492    2714    2717    2786    2847    2881    2929    2931    3001    3024    3049    3243
         3244    3245    3246    3324    3346    3368    3387    3411    3424    3436    3449


ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*.DZDLCA.SEQ/SOL/CRF/PAGNUM=DZDLCA
RUN-TIME:  60 36 6 SECONDS
RUN-TIME RATIO: 189/103=1.8
CORE USED:  26K  (51 PAGES)
```

Spooler runtime 16 Seconds, 68 KCS, 413 disk reads, 3 disk writes, 94 pages