LK11A

PUSH BUTTON MOD DIAG

MD-11-DZLKA-A

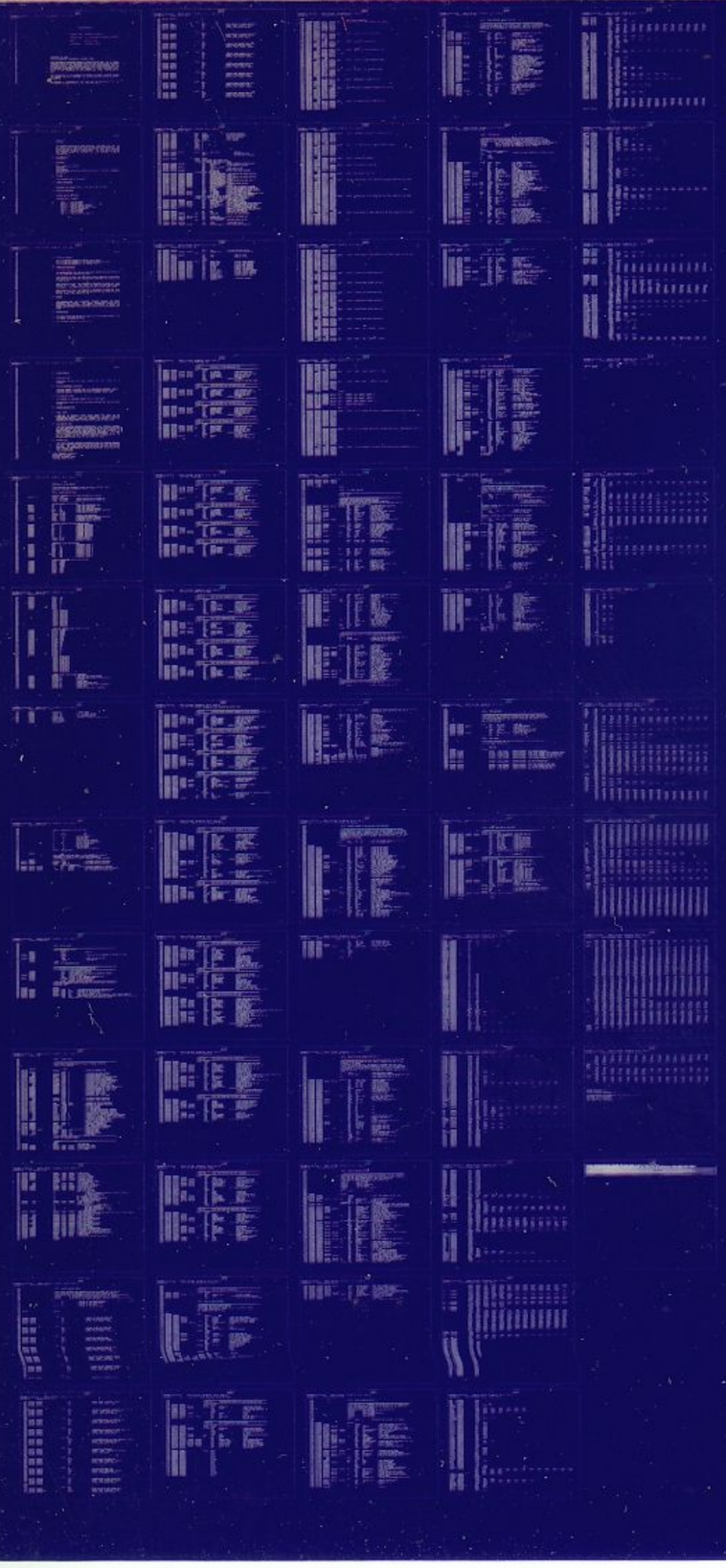
EP-DZLKA-A-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA



```
                        .REM            %
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZLKA-A
PRODUCT NAME: LK11A PUSH BUTTON MODULE DIAGNOSTIC
DATE: AUG. 21, 1976
MAINTAINER: DIAGNOSTIC GROUP

1.0 ABSTRACT

THIS PROGRAM CONSISTS OF THREE SECTIONS. THE FIRST IS A SET OF 25 TESTS WHICH CHECK THE LOGIC OF THE PUSH BUTTON MODULE. THE SECOND SECTION IS ONE TEST WHICH CHECKS THE LAMP DATA REGISTER AND REQUIRES OPERATOR INTERVENTION. THE THIRD SECTION IS ONE TEST WHICH CHECKS THE ENCODER AND ALSO REQUIRES OPERATOR INTERVENTION.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 WITH 4K OF MEMORY
TELETYPE
LK11A MODULE
CABLE AND PUSH BUTTON BOX NEEDED FOR TESTS AT STARTING ADDRESSES 204 AND 210.

2.2 STORAGE

THIS PROGRAM USES 4K OF MEMORY.

3.0 LOADING PROCEDURE

PROCEDURE FOR NORMAL BINARY TYPES SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

STANDARD PDP-11 FORMAT

```
SW15 = 1  HALT ON ERROR
SW14 = 1  LOOP ON TEST
SW13 = 1  INHIBIT ERROR TYPEOUTS
SW12 = 1  HALT AT END OF LOGIC TESTS
SW11 = 1  INHIBIT ITERATIONS
SW10 = 1  BELL ON ERROR
SW9  = 1  LOOP ON ERROR
SW8  = 1  LOOP ON TEST IN SWR<7:0>
```

4.2 STARTING ADDRESS

200 IS STARTING ADDRESS OF THE 25 LOGIC TESTS.
204 IS STARTING ADDRESS OF THE MANUAL INTERVENTION TEST.
210 IS STARTING ADDRESS OF THE ENCODER TEST.

5.0 OPERATING PROCEDURE

THE OPERATOR MUST RUN THE 25 LOGIC TESTS BEFORE RUNNING EITHER OF THE TWO OPERATOR INTERVENTION TESTS.

WHEN STARTING AT 200, THE TEST RUNS WITHOUT OPERATOR INTERVENTION AND PRINTS AN END-OF-PASS MESSAGE WHEN COMPLETED, OR IT WILL HALT IF SW12 IS UP.

WHEN STARTING AT 204, THE OPERATOR IS REQUIRED TO PRESS ONE PUSH BUTTON AT A TIME. THE LIGHT ON THAT BUTTON ONLY SHOULD COMPLEMENT. NO CHANGE SHOULD OCCUR IN ANY OF THE OTHER LIGHTS.

WHEN STARTING AT 210, THE OPERATOR IS REQUIRED TO PRESS ONE PUSH BUTTON AT A TIME. THE LIGHT ON THAT BUTTON ONLY SHOULD TURN ON AND ALL OTHER LIGHTS SHOULD GO OFF.

6.0 ERRORS

THIS PROGRAM USES THE DIAGNOSTIC "SYSMAC" PACKAGE FOR ERROR REPORTING AND TYPEOUT. THE ERROR INFORMATION CONSISTS OF A DESCRIPTIVE MESSAGE ABOUT THE ERROR, THE LOCATION AT WHICH THE ERROR WAS DETECTED, THE ACTUAL DATA VALUE, AND THE EXPECTED DATA VALUE.

7.0 RESTRICTIONS

7.1 THE LOGIC TESTS MUST BE RUN AT LEAST ONE TIME BEFORE RUNNING THE TWO MANUAL INTERVENTION TESTS.

7.2 THIS PROGRAM SUPPORTS ONLY ONE LK11A.

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

EXECUTION TIME FOR ONE PASS THROUGH THE LOGIC TESTS IS 30 SECONDS.

8.2 DEVICE ADDRESSES LOCATIONS

LOCATION "$BASE" CONTAINS THE LK11A BASE DEVICE ADDRESS ‹160060›.
LOCATION "$VECT1" CONTAINS THE LK11A INTERRUPT VECTOR ‹360›.
LOCATION "$VECT1" CONTAINS THE LK11A INTERRUPT LEVEL ‹200›.

8.3 ACT/XXDP/APT NOTES

THE PROGRAM IS CHAINABLE UNDER "ACT-11" AND "XXDP".

THE SOFTWARE HOOKS FOR "APT" HAVE BEEN PROVIDED BUT HAVE NOT BEEN TESTED.

9.0 PROGRAM DESCRIPTION

9.1 LOGIC TESTS

THERE ARE 25 SEPARATE TESTS WHICH CHECK THE LOGIC OF THE PUSH-BUTTON MODULE. THEY RUN SEQUENTIALLY WITHOUT OPERATOR INTERVENTION AND AN END-OF-PASS MESSAGE IS PRINTED AFTER THE COMPLETION OF EACH PASS, OR A HALT WILL OCCUR IF SW12 IS UP.

9.2 PUSH-BUTTON TEST

THIS TEST REQUIRES OPERATOR INTERVENTION. ITS STARTING ADDRESS IS 204 AND THE TEST MUST BE RUN ONLY AFTER THE LOGIC TESTS HAVE BEEN COMPLETED. THE TEST ITSELF DEPENDS UPON THE OPERATOR TO PRESS A SINGLE BUTTON AND TAKE NOTICE OF THE RESULTS. THE LIGHT WITHIN THE BUTTON THAT WAS PRESSED SHOULD COMPLEMENT; IF IT WAS ON, IT SHOULD GO OFF AND IF IT WAS OFF, IT SHOULD GO ON. NO CHANGE SHOULD OCCUR IN ANY OF THE OTHER 15 LIGHTS.

9.3 ENCODER TEST

THIS TEST REQUIRES OPERATOR INTERVENTION AND ITS STARTING ADDRESS IS 210. THE TEST MUST ONLY BE RUN AFTER THE LOGIC TESTS HAVE BEEN RUN. THE OPERATOR MUST PUSH ONE BUTTON AND THE LIGHT WITHIN THAT BUTTON ONLY SHOULD BE TURNED ON AND WON'T GO OFF UNTIL THE TEST IS RESTARTED.

```
        %
.TITLE  MAINDEC-11-DZLKA-A
;*COPYRIGHT (C) 1976
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
```

```
                        ;*
                        ;*PROGRAM BY VERA BREUER
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-CO),MAR 21, 1976.
                        ;*
                        .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
                        .EQUIV  R6,SP           ;;STACK POINTER
                        .EQUIV  R7,PC           ;;PROGRAM COUNTER

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

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
```

```
                    000200          SW07=   200
                    000100          SW06=   100
                    000040          SW05=   40
                    000020          SW04=   20
                    000010          SW03=   10
                    000004          SW02=   4
                    000002          SW01=   2
                    000001          SW00=   1
                                    .EQUIV  SW09,SW9
                                    .EQUIV  SW08,SW8
                                    .EQUIV  SW07,SW7
                                    .EQUIV  SW06,SW6
                                    .EQUIV  SW05,SW5
                                    .EQUIV  SW04,SW4
                                    .EQUIV  SW03,SW3
                                    .EQUIV  SW02,SW2
                                    .EQUIV  SW01,SW1
                                    .EQUIV  SW00,SW0

                                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                    100000          BIT15=  100000
                    040000          BIT14=  40000
                    020000          BIT13=  20000
                    010000          BIT12=  10000
                    004000          BIT11=  4000
                    002000          BIT10=  2000
                    001000          BIT09=  1000
                    000400          BIT08=  400
                    000200          BIT07=  200
                    000100          BIT06=  100
                    000040          BIT05=  40
                    000020          BIT04=  20
                    000010          BIT03=  10
                    000004          BIT02=  4
                    000002          BIT01=  2
                    000001          BIT00=  1
                                    .EQUIV  BIT09,BIT9
                                    .EQUIV  BIT08,BIT8
                                    .EQUIV  BIT07,BIT7
                                    .EQUIV  BIT06,BIT6
                                    .EQUIV  BIT05,BIT5
                                    .EQUIV  BIT04,BIT4
                                    .EQUIV  BIT03,BIT3
                                    .EQUIV  BIT02,BIT2
                                    .EQUIV  BIT01,BIT1
                                    .EQUIV  BIT00,BIT0

                                    ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                    000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                    000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                    000014          TBITVEC=14              ;;"T" BIT
                    000014          TRTVEC= 14              ;;TRACE TRAP
                    000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                    000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                    000024          PWRVEC= 24              ;;POWER FAIL
                    000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
```

```
   299          000034                  TRAPVEC=34              ;;"TRAP" TRAP
   300          000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
   301          000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
   302          000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
   303          160060                  ABASE=160060
   304          100360                  AVECT1=100360
   305          000200                  APRIOR=200
```

```
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        12            HALT AT END OF DIAGNOSTIC
                                ;*        11            INHIBIT ITERATIONS
                                ;*        10            BELL ON ERROR
                                ;*         9            LOOP ON ERROR
                                ;*         8            LOOP ON TEST IN SWR<7:0>
                                .SBTTL  TRAP CATCHER

                                        .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                                        .=174
000174  000000                  DISPREG:  .WORD  0              ;;SOFTWARE DISPLAY REGISTER
000176  000000                  SWREG:    .WORD  0              ;;SOFTWARE SWITCH REGISTER
                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  001620                  JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  004654                  JMP     BEGIN1          ;TEST PBR TURNS LDR LIGHT ON AND OFF
000210  000137  004710                  JMP     BEGIN2          ;TEST ENCODER WITH PUSH BUTTON
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
000214                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  004616                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000214                                  .=$SVPC                 ;; RESTORE PC
001000                          .=1000
                                .SBTTL  APT PARAMETER BLOCK

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

001000                          $APTHD:
001000  000000                  $HIBTS: .WORD   0        ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001202                  $MBADR: .WORD   $MAIL    ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000074                  $TSTM:  .WORD   60.      ;;RUN TIM OF LONGEST TEST
001006  000170                  $PASTM: .WORD   120.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000264                  $UNITM: .WORD   180.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000031                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  .WORD   0
001102  000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          $WR:    .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          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REGO) WAS OBTAINED
001162  000000          $REGO:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000          $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001166  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001170  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001172  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001176  077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
001177  015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001200  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;**********************************************************************
                                .EVEN
001202                  $MAIL:                          ;;APT MAILBOX
001202  000000          $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001204  000000          $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001206  000000          $TESTN: .WORD   ATESTN          ;;TEST NUMBER
```

```
001210  000000          $PASS:   .WORD   APASS    ;;PASS COUNT
001212  000000          $DEVCT:  .WORD   ADEVCT   ;;DEVICE COUNT
001214  000000          $UNIT:   .WORD   AUNIT    ;;I/O UNIT NUMBER
001216  000000          $MSGAD:  .WORD   AMSGAD   ;;MESSAGE ADDRESS
001220  000000          $MSGLG:  .WORD   AMSGLG   ;;MESSAGE LENGTH
001222                  $ETABLE:                  ;;APT ENVIRONMENT TABLE
001222     000          $ENV:    .BYTE   AENV     ;;ENVIRONMENT BYTE
001223     000          $ENVM:   .BYTE   AENVM    ;;ENVIRONMENT MODE BITS
001224  000000          $SWREG:  .WORD   ASWREG   ;;APT SWITCH REGISTER
001226  000000          $USWR:   .WORD   AUSWR    ;;USER SWITCHES
001230  000000          $CPUOP:  .WORD   ACPUOP   ;;CPU TYPE,OPTIONS
                        ;*                        BITS 15-11=CPU TYPE
                        ;*                                11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                        ;*                                11/70=06,PDQ=07,Q=10
                        ;*                        BIT 10=REAL TIME CLOCK
                        ;*                        BIT  9=FLOATING POINT PROCESSOR
                        ;*                        BIT  8=MEMORY MANAGEMENT
001232     000          $MAMS1:  .BYTE   AMAMS1   ;;HIGH ADDRESS,M.S. BYTE
001233     000          $MTYP1:  .BYTE   AMTYP1   ;;MEM. TYPE,BLK#1
                        ;*                        MEM.TYPE BYTE   --  (HIGH BYTE)
                        ;*                               900 NSEC CORE=001
                        ;*                               300 NSEC BIPOLAR=002
                        ;*                               500 NSEC MOS=003
001234  000000          $MADR1:  .WORD   AMADR1   ;;HIGH ADDRESS,BLK#1
                        ;*                          MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001236     000          $MAMS2:  .BYTE   AMAMS2   ;;HIGH ADDRESS,M.S. BYTE
001237     000          $MTYP2:  .BYTE   AMTYP2   ;;MEM.TYPE,BLK#2
001240  000000          $MADR2:  .WORD   AMADR2   ;;MEM.LAST ADDRESS,BLK#2
001242     000          $MAMS3:  .BYTE   AMAMS3   ;;HIGH ADDRESS,M.S.BYTE
001243     000          $MTYP3:  .BYTE   AMTYP3   ;;MEM.TYPE,BLK#3
001244  000000          $MADR3:  .WORD   AMADR3   ;;MEM.LAST ADDRESS,BLK#3
001246     000          $MAMS4:  .BYTE   AMAMS4   ;;HIGH ADDRESS,M.S.BYTE
001247     000          $MTYP4:  .BYTE   AMTYP4   ;;MEM.TYPE,BLK#4
001250  000000          $MADR4:  .WORD   AMADR4   ;;MEM.LAST ADDRESS,BLK#4
001252  100360          $VECT1:  .WORD   AVECT1   ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001254  000000          $VECT2:  .WORD   AVECT2   ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001256  160060          $BASE:   .WORD   ABASE    ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001260  000000          $DEVM:   .WORD   ADEVM    ;;DEVICE MAP
001262  000000          $CDW1:   .WORD   ACDW1    ;;CONTROLLER DESCRIPTION WORD#1
001264                  $ETEND:
                        .MEXIT
```

```
                                .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

001264                          $ERRTB:


                                 ;ITEM  1
001264  005036                          EM1                     ;FAILED TO LOAD AND READ LDR
001266  006565                          DH1                     ;ERRPC LDR EXPECTED ACTUAL
001270  006772                          DT1                     ;$ERRPC, LDR, $GDDAT, $BDDAT, 0
001272  007040                          DF1

                                ;ITEM   2
001274  005104                          EM2                     ;FAILED TO SET OR CLEAR PBR
001276  006625                          DH2                     ;ERRPC PBR EXPECTED ACTUAL
001300  007004                          DT2                     ;$ERRPC, PBR, $GDDAT, $BDDAT, 0
001302  007040                          DF1

                                ;ITEM   3
001304  005137                          EM3                     ;FAILED TO CLEAR PB FLAG
001306  006665                          DH3                     ;ERRPC PBSR EXPECTED ACTUAL
001310  007016                          DT3                     ;$ERRPC, PBSR, $GDDAT, $BDDAT, 0
001312  007040                          DF1

                                ;ITEM   4
001314  005167                          EM4                     ;FAILED TO LOAD AND READ PB FLAG
001316  006665                          DH3                     ;ERRPC PBSR EXPECTED ACTUAL
001320  007016                          DT3                     ;$ERRPC, PBSR, $GDDAT, $BDDAT, 0
001322  007040                          DF1

                                ;ITEM   5
001324  005227                          EM5                     ;FAILED TO CLEAR INTERRUPT ENABLE
001326  006665                          DH3                     ;ERRPC   PBSR    EXPECTED   ACTUAL
001330  007016                          DT3                     ;$ERRPC,PBSR,$GDDAT,$BDDAT
001332  007040                          DF1

                                ;ITEM   6
001334  005270                          EM6                     ;FAILED TO SET INTERRUPT ENABLE
001336  006665                          DH3                     ;ERRPC  PBSR  EXPECTED  ACTUAL
001340  007016                          DT3                     ;$ERRPC,PBSR,$GDDAT,$BDDAT
001342  007040                          DF1


                                ;ITEM   7
001344  005327                          EM7                     ;ERRONEOUSLY CLEARED PB FLAG
```

```
519  001346  006665                    DH3          ;ERRPC  PBSR  EXPECTED  ACTUAL
520  001350  007016                    DT3          ;$ERRPC,PBSR,$GDDAT,$BDDAT
521  001352  007040                    DF1
522
523                          ;ITEM   10
524  001354  005363                    EM10         ;FAILED TO GENERATE INTERRUPT WITH PB FLAG
525  001356  006665                    DH3          ;ERRPC   PBSR  EXPECTED  ACTUAL
526  001360  007016                    DT3          ;$ERRPC,PBSR,$GDDAT,$BDDAT
527  001362  007040                    DF1
528
529                          ;ITEM   11
530  001364  005435                    EM11         ;CLR @PBSR FAILED TO CLEAR PB FLAG
531  001366  006665                    DH3          ;ERRPC  PBSR  EXPECTED  ACTUAL
532  001370  007016                    DT3          ;$ERRPC,PBSR,$GDDAT,$BDDAT
533  001372  007040                    DF1
534
535                          ;ITEM   12
536  001374  005476                    EM12         ;INTERRUPT OK FAILED TO CLEAR INTR FLOP
537  001376  006665                    DH3          ;ERRPC  PBSR  EXPECTED  ACTUAL
538  001400  007016                    DT3          ;$ERRPC,PBSR,$GDDAT,$BDDAT
539  001402  007040                    DF1
540
541                          ;ITEM   13
542  001404  005545                    EM13         ;CLR @PBSR FAILED TO CLEAR INTERRUPT ENABLE
543  001406  006665                    DH3          ;ERRPC  PBSR  EXPECTED  ACTUAL
544  001410  007016                    DT3          ;$ERRPC,PBSR,$GDDAT,$BDDAT
545  001412  007040                    DF1
546
547                          ;ITEM   14
548  001414  005617                    EM14         ;UNEXPECTED INTERRUPT
549  001416  006726                    DH4          ;ERRPC   PBSR   LOCATION OF FAILURE
550  001420  007030                    DT4          ;$ERRPC,PBSR,$GDDAT
551  001422  007040                    DF1
552
553                          ;ITEM   15
554  001424  005644                    EM15         ;BIC FAILED TO CLEAR LDR
555  001426  006665                    DH3          ;ERRPC  PBSR  EXPECTED   ACTUAL
556  001430  007016                    DT3          ;$ERRPC,PBSR,$GDDAT,$BDDAT
557  001432  007040                    DF1
558
559                          ;ITEM   16
560  001434  005675                    EM16         ;BIS FAILED TO SET LDR
561  001436  006665                    DH3          ;ERRPC   PBSR  EXPECTED  ACTUAL
562  001440  007016                    DT3          ;$ERRPC,PBSR,$GDDAT,$BDDAT
563  001442  007040                    DF1
564
565                          ;ITEM   17
566  001444  005723                    EM17         ;BISB FAILED TO SET LOWER BYTE
567  001446  006665                    DH3          ;ERRPC  PBSR  EXPECTED  ACTUAL
568  001450  007016                    DT3          ;$ERRPC,PBSR,$GDDAT,$BDDAT
569  001452  007040                    DF1
570
571                          ;ITEM   20
572  001454  006016                    EM20         ;BISB FAILED TO SET UPPER BYTE
573  001456  006665                    DH3          ;ERRPC  PBSR  EXPECTED  ACTUAL
574  001460  007016                    DT3          ;$ERRPC,PBSR,$GDDAT,$BDDAT
```

```
575  001462  007040                          DF1
576
577                                  ;ITEM   21
578  001464  006111                          EM21            ;BICB FAILED TO CLEAR LOWER BYTE
579  001466  006665                          DH3             ;ERRPC  PBSR  EXPCTED  ACTUAL
580  001470  007016                          DT3             ;$ERRPC,PBSR,$GDDAT,$BDDAT
581  001472  007040                          DF1
582
583                                  ;ITEM   22
584  001474  006206                          EM22            ;BICB FAILED TO CLEAR UPPER BYTE
585  001476  006665                          DH3             ;ERRPC  PBSR  EXPECTED  ACTUAL
586  001500  007016                          DT3             ;$ERRPC,PBSR,$GDDAT,$BDDAT
587  001502  007040                          DF1
588
589                                  ;ITEM   23
590  001504  006303                          EM23            ;LOADING LDR CHANGED PBSR DATA
591  001506  006665                          DH3             ;ERRPC PBSR EXPECTED ACTUAL
592  001510  007016                          DT3             ;$ERRPC, PBSR, $GDDAT, $BDDAT
593  001512  007040                          DF1
594
595                                  ;ITEM   24
596  001514  006341                          EM24            ;LOADING PBSR CHANGED LDR DATA
597  001516  006565                          DH1             ;ERRPC LDR EXPECTED ACTUAL
598  001520  006772                          DT1             ;$ERRPC, LDR, $GDDAT, $GDDAT
599  001522  007040                          DF1
600
601                                  ;ITEM   25
602  001524  006377                          EM25            ;LOADING LDR CHANGED PBR DATA
603  001526  006625                          DH2             ;ERRPC PBR EXPECTED ACTUAL
604  001530  006772                          DT1             ;$ERRPC, PBSR, $GDDAT, $BDDAT
605  001532  007040                          DF1
606
607                                  ;ITEM   26
608  001534  006434                          EM26            ;LOADING PBR CHANGED LDR DATA
609  001536  006565                          DH1             ;ERRPC LDR EXPECTED ACTUAL
610  001540  006772                          DT1             ;$ERRPC, LDR, $GDDAT, $BDDAT
611  001542  007040                          DF1
612
613                                  ;ITEM   27
614  001544  006471                          EM27            ;LOADING PBSR CHANGED PBR DATA
615  001546  006625                          DH2             ;ERRPC PBR EXPECTED ACTUAL
616  001550  007004                          DT2             ;$ERRPC, PBR, $GDDAT, $BDDAT
617  001552  007040                          DF1
618
619                                  ;ITEM   30
620  001554  006527                          EM30            ;LOADING PBR CHANGED PBSR DATA
621  001556  006665                          DH3             ;ERRPC PBSR EXPECTED ACTUAL
622  001560  007016                          DT3             ;$ERRPC, PBSR, $GDDAT, $BDDAT
623  001562  007040                          DF1
```

```
                                PBSR:   ABASE                   ;PB STATUS REGISTER
                                PBR:    ABASE+2                 ;PUSH BUTTON REGISTER
                                LDR:    ABASE+4                 ;LAMP DATA REISTER
                                VECT:   AVECT1                  ;INTR. VECTOR
                                VECTR1: AVECT1+2                ;VECT ADDRESS
                                TEMP:   0
                                TEMP1:  0
                                TEMP2:  0
                                TEMPL:  0
                                BITSAV: 0

                                UNEXP:  MOV     (SP),$GDDAT             ;POP PC
                                        ERROR   14                      ;UNEXPECTED INTR.
                                        RTI                             ;RETURN FROM INTERRUPT

                                BEGIN:  RESET
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
                                        MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
                                        CLR     (R6)+           ;;CLEAR MEMORY LOCATION
                                        CMP     #SWR,R6 ;;DONE?
                                        BNE     .-6             ;;LOOP BACK IF NO
                                        MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
                                        MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
                                        MOV     #340,@#IOTVEC+2 ;;LEVEL 7
                                        MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
                                        MOV     #340,@#EMTVEC+2 ;;LEVEL 7
                                        MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
                                        MOV     #340,@#TRAPVEC+2;LEVEL 7
                                        MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
                                        MOV     #340,@#PWRVEC+2 ;;LEVEL 7
                                        MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
                                        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     $PASS           ;;CLEAR PASS COUNT
                                        BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
```

```
                                                BEQ     67$                     ;;YES,USE NON-APT SWITCH
                                                MOV     #$SWREG,SWR             ;;NO,USE APT SWITCH REGISTER
                                        67$:
                                                TST     @#42                    ;TEST IF XXDP
                                                BNE     INIT                    ;BRANCH IF YES
                                                TYPE
                                                HEAD1
                                        INIT:   MOV     $BASE,PBSR              ;INITIAL BASE ADDRESS
                                                MOV     $BASE,LDR               ;INITIAL LDR ADDRESS
                                                MOV     $BASE,PBR               ;INITIAL PBR ADDRESS
                                                ADD     #2,PBR                  ;CORRECT PBR ADDRESS
                                                ADD     #4,LDR                  ;CORRECT LDR ADDRESS
                                                MOV     $VECT1,VECT             ;INITIAL VECTER0 ADDRESS
                                                MOV     $VECT1,VECTR1           ;INITIAL VECTR1 ADDRESS
                                                ADD     #2,VECTR1               ;CORRECT VECTR1 ADDRESS
                                                BIC     #160000,VECT            ;CLEAR PRIORITY BITS
                                                BIC     #160000,VECTR1
                                                MOV     #UNEXP,@VECT            ;INTERRUPT ADDRESS
                                                CLR     @VECTR1
```

```
                                ;;*****************************************************************
                                ;*TEST 1         LOAD ALL 1'S AND READ BACK LDR
                                ;;*****************************************************************
002212  000004                  TST1:   SCOPE
002214  012737  177777  001124          MOV     #-1,$GDDAT              ;LOAD EXPECTED DATA
002222  013777  001124  177340          MOV     $GDDAT,@LDR             ;LOAD ALL ONES INTO LDR
002230  017737  177334  001126          MOV     @LDR,$BDDAT             ;READ ACTUAL DATA
002236  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002244  001401                          BEQ     TST2            ;;BRANCH IF EQUAL
002246  104001                          ERROR   1                       ;FAILED TO LOAD ALL ONES

                                ;;*****************************************************************
                                ;*TEST 2         LOAD AND READ '25' PATTERN INTO LDR
                                ;;*****************************************************************
002250  000004                  TST2:   SCOPE
002252  012737  125252  001124          MOV     #125252,$GDDAT          ;LOAD EXPECTED DATA
002260  013777  001124  177302          MOV     $GDDAT,@LDR             ;LOAD PATTERN INTO LDR
002266  017737  177276  001126          MOV     @LDR,$BDDAT             ;READ ACTUAL DATA
002274  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002302  001401                          BEQ     TST3            ;;BRANCH IF EQUAL
002304  104001                          ERROR   1                       ;FAILED TO LOAD PATTERN

                                ;;*****************************************************************
                                ;*TEST 3         LOAD AND READ '52' PATTERN INTO LDR
                                ;;*****************************************************************
002306  000004                  TST3:   SCOPE
002310  012737  052525  001124          MOV     #52525,$GDDAT           ;LOAD EXPECTED DATA
002316  013777  001124  177244          MOV     $GDDAT,@LDR             ;LOAD PATTERN INTO LDR
002324  017737  177240  001126          MOV     @LDR,$BDDAT             ;READ ACTUAL DATA
002332  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002340  001401                          BEQ     TST4            ;;BRANCH IF EQUAL
002342  104001                          ERROR   1                       ;FAILED TO LOAD PATTERN

                                ;;*****************************************************************
                                ;*TEST 4         FLOAT A ONE THROUGH LDR
                                ;;*****************************************************************
002344  000004                  TST4:   SCOPE
002346  012737  000001  001124          MOV     #BIT0,$GDDAT            ;LOAD EXPECTED DATA
002354  013777  001124  177206  2$:     MOV     $GDDAT,@LDR             ;LOAD LDR BIT
002362  017737  177202  001126          MOV     @LDR,$BDDAT             ;READ ACTUAL DATA
002370  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002376  001401                          BEQ     1$              ;;BRANCH IF EQUAL
002400  104001                          ERROR   1                       ;FAILED TO SET LDR BIT
002402  006137  001124          1$:     ROL     $GDDAT                  ;TEST NEXT BIT
002406  103362                          BCC     2$                      ;CONTINUE THROUGH LDR
```

```
                                ;;*****************************************************************
                                ;*TEST 5         FLOAT ZERO THROUGH LDR
                                ;;*****************************************************************
000410  000004                  TST5:   SCOPE
002412  012737  177776  001124          MOV     #177776,$GDDAT          ;LOAD EXPECTED DATA
002420  013777  001124  177142  2$:     MOV     $GDDAT,@LDR             ;LOAD LDR BIT
002426  017737  177136  001126          MOV     @LDR,$BDDAT             ;READ ACTUAL DATA
002434  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002442  001401                        BEQ     1$              ;;BRANCH IF EQUAL
002444  104001                        ERROR   1                       ;FAILED TO CLEAR LDR BIT
002446  000261                  1$:     SEC                             ;SET CARRY BIT
002450  006137  001124                ROL     $GDDAT                  ;TEST NEXT BIT
002454  103361                        BCC     2$                      ;CONTINUE THROUGH LDR

                                ;;*****************************************************************
                                ;*TEST 6         LOAD AND READ PBR USING BIT 0
                                ;;*****************************************************************
002456  000004                  TST6:   SCOPE
002460  012737  177777  001124          MOV     #-1,$GDDAT              ;LOAD EXPECTED DATA
002466  012777  000001  177072          MOV     #BIT0,@PBR              ;SET BIT 0
002474  017737  177066  001126          MOV     @PBR,$BDDAT             ;READ ACTUAL DATA
002502  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002510  001401                        BEQ     TST7            ;;BRANCH IF EQUAL
002512  104002                        ERROR   2                       ;FAILED TO SET PBR

                                ;;*****************************************************************
                                ;*TEST 7         CLEAR PBR USING BIT 0
                                ;;*****************************************************************
002514  000004                  TST7:   SCOPE
002516  005037  001124                  CLR     $GDDAT                  ;EXPECTED DATA
002522  005077  177040                  CLR     @PBR                    ;CLEAR BIT 0
002526  017737  177034  001126          MOV     @PBR,$BDDAT             ;READ ACTUAL DATA
002534  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002542  001401                        BEQ     TST10           ;;BRANCH IF EQUAL
002544  104002                        ERROR   2                       ;FAILED TO CLEAR PBR

                                ;;*****************************************************************
                                ;*TEST 10        LOAD AND READ BACK PB FLAG
                                ;;*****************************************************************
002546  000004                  TST10:  SCOPE
002550  005077  177010                  CLR     @PBSR                   ;CLEAR STATUS REGISTER
002554  012737  100000  001124          MOV     #BIT15,$GDDAT           ;LOAD EXPECTED DATA
002562  013777  001124  176774          MOV     $GDDAT,@PBSR            ;LOAD STATUS REG.
002570  017737  176770  001126          MOV     @PBSR,$BDDAT            ;READ ACTUAL DATA
002576  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002604  001401                        BEQ     TST11           ;;BRANCH IF EQUAL
002606  104004                        ERROR   4                       ;FAILED TO SET BIT 15
```

```
                                ;;*****************************************************************
                                ;*TEST 11       TEST PB FLAG CLEARED BY CLR PBSR
                                ;;*****************************************************************
002610  000004                  TST11:  SCOPE
002612  005077  176746                  CLR     @PBSR                   ;CLEAR STATUS REGISTER
002616  005037  001124                  CLR     $GDDAT                  ;EXPECTED DATA
002622  052777  100000  176734          BIS     #BIT15,@PBSR            ;SET PB FLAG
002630  005077  176730                  CLR     @PBSR                   ;CLEAR STATUS REGISTER
002634  017737  176724  001126          MOV     @PBSR,$BDDAT            ;READ ACTUAL DATA
002642  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002650  001401                          BEQ     TST12           ;;BRANCH IF EQUAL
002652  104011                          ERROR   11                      ;FAILED TO CLEAR PB FLAG

                                ;;*****************************************************************
                                ;*TEST 12       TEST INIT CLEARS PB FLAG
                                ;;*****************************************************************
002654  000004                  TST12:  SCOPE
002656  012737  000100  001166          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
002664  005077  176674                  CLR     @PBSR                   ;CLEAR STATUS REG.
002670  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED DATA
002674  012777  100000  176662          MOV     #BIT15,@PBSR            ;SET PB FLAG
002702  000005                          RESET                           ;INITIALIZE
002704  017737  176654  001126          MOV     @PBSR,$BDDAT            ;READ ACTUAL DATA
002712  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002720  001401                          BEQ     TST13           ;;BRANCH IF EQUAL
002722  104003                          ERROR   3                       ;FAILED TO CLEAR PB FLAG

                                ;;*****************************************************************
                                ;*TEST 13       TEST PB FLAG CLEARED WHEN READ PBR H
                                ;;*****************************************************************
002724  000004                  TST13:  SCOPE
002726  005077  176632                  CLR     @PBSR                   ;CLEAR STATUS REGISTER
002732  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED DATA
002736  012777  100000  176620          MOV     #BIT15,@PBSR            ;SET PB FLAG
002744  017737  176616  001576          MOV     @PBR,TEMP               ;READ PB REGISTER
002752  017737  176606  001126          MOV     @PBSR,$BDDAT            ;READ ACTUAL DATA
002760  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
002766  001401                          BEQ     TST14           ;;BRANCH IF EQUAL
002770  104003                          ERROR   3                       ;FAILED TO CLEAR PB FLAG

                                ;;*****************************************************************
                                ;*TEST 14       TEST  PB FLAG IS CLEARED BY READ STATUS WHEN INTR. ENABLE IS CLEARED
                                ;;*****************************************************************
002772  000004                  TST14:  SCOPE
002774  005077  176564                  CLR     @PBSR                   ;CLEAR STATUS REG
003000  005037  001124                  CLR     $GDDAT                  ;EXPECTED DATA
003004  012777  100000  176552          MOV     #BIT15,@PBSR            ;SET PB FLAG
003012  017737  176546  001576          MOV     @PBSR,TEMP              ;READ SR
003020  017737  176540  001126          MOV     @PBSR,$BDDAT            ;READ ACTUAL DATA
003026  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
003034  001401                          BEQ     TST15           ;;BRANCH IF EQUAL
003036  104003                          ERROR   3                       ;READ SR FAILED TO CLEAR PB FLAG
```

```
                                ;;*****************************************************************
                                ;*TEST 15       LOAD AND READ INTERRUPT ENABLE BIT,ALSO UNEXPECTED INTRPT TEST
                                ;;*****************************************************************
003040  000004            TST15:  SCOPE
003042  005077  176516            CLR     @PBSR                   ;CLEAR PB STATUS REGISTER
003046  012737  040000  001124    MOV     #BIT14,$GDDAT           ;LOAD EXPECTED DATA
003054  013777  001124  176502    MOV     $GDDAT,@PBSR            ;LOAD STATUS REGISTER
003062  017737  176476  001126    MOV     @PBSR,$BDDAT            ;READ ACTUAL DATA
003070  023737  001124  001126    CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
003076  001401                    BEQ     TST16           ;;BRANCH IF EQUAL
003100  104006                    ERROR   6                       ;FAILED TO LOAD INTERRUPT ENABLE
                                ;;*****************************************************************
                                ;*TEST 16       TEST INTERRUPT ENABLE CLEARED BY INIT
                                ;;*****************************************************************
003102  000004            TST16:  SCOPE
003104  012737  000100  001166    MOV     #100,$TIMES     ;;DO 100 ITERATIONS
003112  005037  001124            CLR     $GDDAT                  ;LOAD EXPECTED DATA
003116  005077  176442            CLR     @PBSR                   ;ENSURE CLEAR REG
003122  012777  040000  176434    MOV     #BIT14,@PBSR            ;SET BIT 14
003130  000005                    RESET                           ;INITIALIZE
003132  017737  176426  001126    MOV     @PBSR,$BDDAT            ;READ ACTUAL DATA
003140  023737  001124  001126    CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
003146  001401                    BEQ     TST17           ;;BRANCH IF EQUAL
003150  104005                    ERROR   5                       ;FAILED TO CLEAR INTRPT ENABLE
                                ;;*****************************************************************
                                ;*TEST 17       TEST INTERRUPT ENABLE CLEARED BY CLR PBSR
                                ;;*****************************************************************
003152  000004            TST17:  SCOPE
003154  005077  176404            CLR     @PBSR                   ;CLEAR STATUS REGISTER
003160  005037  001124            CLR     $GDDAT                  ;EXPECTED DATA
003164  052777  040000  176372    BIS     #BIT14,@PBSR            ;SET INTERRUPT ENABLE
003172  005077  176366            CLR     @PBSR                   ;CLEAR STATUS REGISTER
003176  017737  176362  001126    MOV     @PBSR,$BDDAT            ;READ ACTUAL DATA
003204  023737  001124  001126    CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
003212  001401                    BEQ     TST20           ;;BRANCH IF EQUAL
003214  104013                    ERROR   13                      ;FAILED TO CLEAR INTERRUPT ENABLE
                                ;;*****************************************************************
                                ;*TEST 20       GENERATE INTERRUPT WITH PB FLAG AND INTRPT EN
                                ;;*****************************************************************
003216  000004            TST20:  SCOPE
003220  005077  176340            CLR     @PBSR                   ;CLEAR STATUS REGISTER
003224  012777  003274  176340    MOV     #2$,@VECT               ;INTERRUPT ADDRESS
003232  012737  140000  001124    MOV     #BIT15!BIT14,$GDDAT     ;EXPECTED DATA
003240  005037  001126            CLR     $BDDAT
003244  012777  040000  176312    MOV     #BIT14,@PBSR            ;SET INTERRUPT ENABLE
003252  012777  140000  176304    MOV     #BIT15!BIT14,@PBSR      ;SET PB FLAG
003260  005037  177776            CLR     PSW                     ;LOWER PSW
003264  000240            1$:     NOP
003266  000240                    NOP
003270  104010                    ERROR   10                      ;PB FLAG FAILED TO GENERATE INTERRUPT
003272  000401                    BR      3$              ;;BRANCH AND CLEAR I.E.
003274  022626            2$:     CMP     (SP)+,(SP)+             ;POP STACK
003276  005077  176262    3$:     CLR     @PBSR                   ;CLEAR I.E.
```

```
                                        ;;*****************************************************************
                                        ;*TEST 21       TEST FOR CORRECTNESS OF  PRIORITY OF INTERRUPT
                                        ;;*****************************************************************
003302  000004                TST21:  SCOPE
003304  005077  176254                CLR     @PBSR                   ;CLEAR STATUS REGISTER
003310  113737  001253  001576        MOVB    $VECT1+1,TEMP           ;GET PRIORITY LEVEL
003316  042737  177437  001576        BIC     #177437,TEMP            ;MASK TO VECTOR BITS
003324  013737  001576  001600        MOV     TEMP,TEMP1                      ;SAVE IT
003332  162737  000040  001600        SUB     #40,TEMP1               ;LOWER PRIORITY
003340  012777  003404  176224        MOV     #1$,@VECT               ;INTERRUPT ADDRESS
003346  012777  000340  176220        MOV     #PR7,@VECTR1            ;LOAD VECTOR ADDRESS PSW
003354  013737  001576  177776        MOV     TEMP,PSW                ;LOWER PSW
003362  012777  040000  176174        MOV     #BIT14,@PBSR            ;SET INTERRUPT ENABLE
003370  012777  140000  176166        MOV     #BIT15!BIT14,@PBSR      ;SET PB FLAG
003376  000240                        NOP
003400  000240                        NOP
003402  000403                        BR      2$                      ;INTERRUPT NOT SERVICED
003404  022626                1$:     CMP     (SP)+,(SP)+             ;POP    STACK
003406  104010                        ERROR   10                      ;INTERRUPT LEVEL NOT CORRECT
003410  000425                        BR      TST22           ;;BRANCH TO NEXT TEST
003412  012777  003450  176152  2$:     MOV     #3$,@VECT               ;INTERRUPT ADDRESS
003420  012777  000340  176146        MOV     #PR7,@VECTR1            ;SET PRIORITY
003426  013737  001600  177776        MOV     TEMP1,PSW               ;LOAD PSW
003434  000240                        NOP
003436  000240                        NOP
003440  005037  001126                CLR     $BDDAT
003444  104010                        ERROR   10                      ;PB FLAG FAILED TO GENERATE INTERRUPT
003446  000406                        BR      TST22           ;;BRANCH TO NEXT TEST
003450  022626                3$:     CMP     (SP)+,(SP)+             ;POP STACK
003452  012777  001610  176112        MOV     #UNEXP,@VECT            ;UNEXPECTED INTERRUPT ADDRESS
003460  005077  176100                CLR     @PBSR                   ;CLEAR OUT PBSR

                                        ;;*****************************************************************
                                        ;*TEST 22       TEST PB FLAG IS NOT CLEARED BY READ STATUS WHEN INTR ENABLE IS SET
                                        ;;*****************************************************************
003464  000004                TST22:  SCOPE
003466  012737  000340  177776        MOV     #340,PSW
003474  005077  176066                CLR     @PBR                    ;CLEAR BUTTON REGISTER
003500  005077  176060                CLR     @PBSR                   ;CLEAR STATUS REGISTER
003504  012737  140000  001124        MOV     #BIT15!BIT14,$GDDAT     ;LOAD EXPECTED DATA
003512  013777  001124  176044        MOV     $GDDAT,@PBSR            ;LOAD STATUS
003520  017737  176040  001126        MOV     @PBSR,$BDDAT            ;READ STATUS REG
003526  017737  176032  001126        MOV     @PBSR,$BDDAT            ;READ STATUS REG AGAIN
003534  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE REGISTERS
003542  001401                        BEQ     TST23           ;;BR IF PB FLAG CLEARED IN ERROR
003544  104007                        ERROR   7                       ;ERRONEOUSLY CLEARED PB FLAG
```

```
                                ;;*****************************************************************
                                ;*TEST 23       TEST BIC INSTRUCTION CLEARS LDR
                                ;;*****************************************************************
003546  000004                  TST23:  SCOPE
003550  005077  176014                  CLR     @LDR                    ;CLEAR LAMP DATA REGISTER
003554  012737  000001  001606          MOV     #BIT0,BITSAV            ;INITIALIZE BITSAV
003562  012777  177777  176000  2$:     MOV     #-1,@LDR                ;SET REIGSTER TO ALL 1'S
003570  012737  177777  001124          MOV     #-1,$GDDAT              ;EXPECTED
003576  043737  001606  001124          BIC     BITSAV,$GDDAT           ;DATA
003604  043777  001606  175756          BIC     BITSAV,@LDR             ;CLEAR PBSR BIT
003612  017737  175752  001126          MOV     @LDR,$BDDAT             ;READ ACTAL DATA
003620  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
003626  001401                          BEQ     1$                      ;BRANCH IF EQUAL
003630  104015                          ERROR   15                      ;FAILED TO CLEAR LDT BIT
003632  006137  001606          1$:     ROL     BITSAV                  ;SHIFT TO TEST NEXT BIT
003636  103351                          BCC     2$                      ;BRANCH IF NOT DONE ALL BITS
                                ;;*****************************************************************
                                ;*TEST 24       TEST BIS INTRUCTION SETS LDR
                                ;;*****************************************************************
003640  000004                  TST24:  SCOPE
003642  012737  000001  001606          MOV     #BIT0,BITSAV            ;INITIAIALIZE BITSAV
003650  005077  175714          2$:     CLR     @LDR                    ;CLEAR LAMP DATA REGISTER
003654  013737  001606  001124          MOV     BITSAV,$GDDAT           ;DATA
003662  053777  001606  175700          BIS     BITSAV,@LDR             ;SET LDT BIT
003670  017737  175674  001126          MOV     @LDR,$BDDAT             ;READ ACTUAL DATA
003676  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
003704  001401                          BEQ     1$                      ;BRANCH IF EQUAL
003706  104016                          ERROR   16                      ;FAILED TO SET LDR BIT
003710  006137  001606          1$:     ROL     BITSAV                  ;SHIFT TO TEST NEXT BIT
003714  103355                          BCC     2$                      ;BRANCH BACK IF NOT DONE ALL BITS

                                ;;*****************************************************************
                                ;*TEST 25       TEST BISB INSTRUCTION SETS LDR
                                ;;*****************************************************************
003716  000004                  TST25:  SCOPE
003720  005077  175644                  CLR     @LDR                    ;CLEAR LAMP DATA REGISTER
003724  012777  052525  175636          MOV     #52525,@LDR             ;LOAD ALTERNATING 1'S IN LDR
003732  012737  052777  001124          MOV     #52777,$GDDAT           ;EXPECTED DATA
003740  152777  177777  175622          BISB    #-1,@LDR                ;SET LOWER BYTE TO ALL 1'S
003746  017737  175616  001126          MOV     @LDR,$BDDAT             ;READ ACTUAL DATA
003754  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
003762  001401                          BEQ     1$                      ;BRANCH IF EQUAL
003764  104017                          ERROR   17                      ;FAILED TO SET LOWER BYTE & LEAVE UPPER BYTE THE
003766  012777  052525  175574  1$:     MOV     #52525,@LDR             ;LOAD ALTERNATING 1'S IN LDR
003774  013737  001570  001604          MOV     LDR,TEMPL               ;GET ADDRESS OF LDR
004002  005237  001604                  INC     TEMPL                   ;GET ADDRESS OF UPPER BYTE OF LDR
004006  012737  177525  001124          MOV     #177525,$GDDAT          ;EXPECTED DATA
004014  152777  177777  175562          BISB    #-1,@TEMPL              ;SET UPPER BYTE TO ALL 1'S
004022  017737  175542  001126          MOV     @LDR,$BDDAT             ;READ ACTUAL DATA
004030  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
004036  001401                          BEQ     TST26           ;;BRANCH IF EQUAL
004040  104020                          ERROR   20                      ;FAILED TO SET UPPER BYTE &
                                                                        ; LEAVE LOWER BYTE THE SAME
```

```
1000                                    ;;*****************************************************************
1001                                    ;*TEST 26       TEST BICB INSTRUCTION CLEARS LDR
1002                                    ;;*****************************************************************
1003  004042  000004                   TST26:  SCOPE
1004  004044  005077  175520                   CLR     @LDR                    ;CLEAR LAMP DATA REGISTER
1005  004050  012777  052525  175512           MOV     #52525,@LDR             ;LOAD ALTERNATING 1'S IN LSR
1006  004056  012737  052400  001124           MOV     #52400,$GDDAT           ;EXPECTED DATA
1007  004064  142777  177777  175476           BICB    #-1,@LDR                ;CLEAR LOWER BYTE OF LDR
1008  004072  017737  175472  001126           MOV     @LDR,$BDDAT             ;READ ACTUAL DTA
1009  004100  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1010  004106  001401                           BEQ     1$                      ;BRANCH IF EQUAL
1011  004110  104021                           ERROR   21                      ;FAILED TO CLEAR LOWER BYTE & LEAVE UPPER BYTE T
1012  004112  012777  052525  175450  1$:     MOV     #52525,@LDR             ;LOAD ALTERNATING 1'S IN LDR
1013  004120  013737  001570  001604           MOV     LDR,TEMPL               ;GET ADDRESS OF LDR
1014  004126  005237  001604                   INC     TEMPL                   ;GET ADDRESS OF UPPER BYTE OF LDR
1015  004132  012737  000125  001124           MOV     #125,$GDDAT             ;EXPECTED DATA
1016  004140  142777  177777  175436           BICB    #-1,@TEMPL              ;CLEAR UPPER BYTE OF LDR
1017  004146  017737  175416  001126           MOV     @LDR,$BDDAT             ;READ ACTUAL DATA
1018  004154  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1019  004162  001401                           BEQ     TST27           ;;BRANCH IF EQUAL
1020  004164  104022                           ERROR   22                      ;FAILED TO CLEAR UPPER BYTE &
1021                                                                            ; LEAVE LOWER BYTE THE SAME
1022
1023                                    ;;*****************************************************************
1024                                    ;*TEST 27       TEST THAT PBSR AND LDR DON'T CHANGE
1025                                    ;;*****************************************************************
1026  004166  000004                   TST27:  SCOPE
1027  004170  005077  175370                   CLR     @PBSR                   ;CLEAR STATUS REGISTER
1028  004174  005037  001124                   CLR     $GDDAT                  ;LOAD EXPECTED DATA
1029  004200  012777  052525  175362           MOV     #52525,@LDR             ;LOAD PATTERN INTO LDR
1030  004206  017737  175352  001126           MOV     @PBSR,$BDDAT            ;READ STATUS REGISTER
1031  004214  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1032  004222  001401                           BEQ     1$              ;;GO TO NEXT SECTION
1033  004224  104023                           ERROR   23
1034  004226  005077  175336           1$:     CLR     @LDR                    ;CLEAR LDR
1035  004232  012777  100000  175324           MOV     #BIT15,@PBSR            ;SET BIT 15 IN PBSR
1036  004240  017737  175324  001126           MOV     @LDR,$BDDAT             ;READ LDR DATA
1037  004246  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1038  004254  001401                           BEQ     TST30           ;;GO TO NEXT TEST
1039  004256  104024                           ERROR   24
1040
```

```
1041                                    ;;*****************************************************************
1042                                    ;*TEST 30       TEST THAT PBR AND LDR DON'T CHANGE
1043                                    ;;*****************************************************************
1044  004260  000004                    TST30:  SCOPE
1045  004262  005077  175300                   CLR     @PBR                    ;CLEAR PBR
1046  004266  012777  000001  175272           MOV     #BIT0,@PBR              ;SET ALL 1'S IN PBR
1047  004274  012737  177777  001124           MOV     #177777,$GDDAT          ;LOAD EXPECTED DATA
1048  004302  012777  052525  175260           MOV     #52525,@LDR             ;LOAD PATTERN INTO LDR
1049  004310  017737  175252  001126           MOV     @PBR,$BDDAT             ;READ PBR
1050  004316  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1051  004324  001401                           BEQ     1$                      ;;GO TO NEXT SECTION
1052  004326  104025                           ERROR   25
1053  004330  005077  175234           1$:     CLR     @LDR                    ;CLEAR LDR
1054  004334  005077  175226                   CLR     @PBR                    ;CLEAR PBR
1055  004340  005037  001124                   CLR     $GDDAT                  ;LOAD EXPECTED DATA
1056  004344  012777  000001  175214           MOV     #BIT0,@PBR              ;SET PBR TO ALL ONES
1057  004352  017737  175212  001126           MOV     @LDR,$BDDAT             ;READ LDR
1058  004360  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;READ LDR
1059  004366  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1060  004374  001401                           BEQ     TST31                   ;;GO TO NEXT TEST
1061  004376  104026                           ERROR   26
1062
1063                                    ;;*****************************************************************
1064                                    ;*TEST 31       TEST THAT PBSR AND PBR DON'T CHANGE
1065                                    ;;*****************************************************************
1066  004400  000004                    TST31:  SCOPE
1067  004402  005077  175160                   CLR     @PBR                    ;CLEAR PBR
1068  004406  005037  001124                   CLR     $GDDAT                  ;LOAD EXPECTED DATA
1069  004412  012777  052525  175144           MOV     #52525,@PBSR            ;LOAD PATTERN INTO PBSR
1070  004420  017737  175142  001126           MOV     @PBR,$BDDAT             ;READ PBR
1071  004426  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1072  004434  001401                           BEQ     1$                      ;;GO TO NEXT SECTION
1073  004436  104027                           ERROR   27
1074  004440  005077  175120           1$:     CLR     @PBSR                   ;CLEAR PBSR
1075  004444  012777  000001  175114           MOV     #BIT0,@PBR              ;SET PBR TO ALL ONES
1076  004452  017737  175106  001126           MOV     @PBSR,$BDDAT            ;READ PBSR
1077  004460  042737  000036  001126           BIC     #36,$BDDAT              ;MASK TO "ENCODER" BITS
1078  004466  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE RESULTS
1079  004474  001401                           BEQ     2$                      ;;BR IF CORRECT
1080  004476  104030                           ERROR   30                      ;PBSR CHANGED IN ERROR
1081  004500  005077  175062           2$:     CLR     @PBR                    ;CLEAR BP REGISTER
1082  004504  005077  175060                   CLR     @LDR                    ;CLEAR LAMPS
1083  004510  005077  175050                   CLR     @PBSR                   ;CLEAR STATUS
1084
```

```
1085
1086                                    ;;**********************************************************************
1087                                    ;*TEST 32       HALT AT END OF DIAGNOSTIC
1088                                    ;;**********************************************************************
1089  004514  000004                    TST32:  SCOPE
1090  004516  032777  010000  174414            BIT     #BIT12,@SWR             ;IS BIT 12 SET?
1091  004524  001401                            BEQ     $EOP                ;;NO-PRINT END-OF-PASS MESSAGE
1092  004526  000000                            HALT                            ;HALT OT END OF LOGIC TESTS
1093
1094                                    .SBTTL  END OF PASS ROUTINE
1095
1096                                    ;;**********************************************************************
1097                                    ;*INCREMENT THE PASS NUMBER ($PASS)
1098                                    ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
1099                                    ;*IF THERES A MONITOR GO TO IT
1100                                    ;*IF THERE ISN'T JUMP TO INIT
1101
1102  004530                            $EOP:
1103  004530  000004                            SCOPE
1104  004532  005037  001102                    CLR     $TSTNM              ;;ZERO THE TEST NUMBER
1105  004536  005037  001166                    CLR     $TIMES              ;;ZERO THE NUMBER OF ITERATIONS
1106  004542  005237  001210                    INC     $PASS               ;;INCREMENT THE PASS NUMBER
1107  004546  042737  100000  001210          BIC     #100000,$PASS       ;;DON'T ALLOW A NEG. NUMBER
1108  004554  005327                          DEC     (PC)+               ;;LOOP?
1109  004556  000001                    $EOPCT: .WORD   1
1110  004560  003022                            BGT     $DOAGN              ;;YES
1111  004562  012737                            MOV     (PC)+,@(PC)+        ;;RESTORE COUNTER
1112  004564  000001                    $ENDCT: .WORD   1
1113  004566  004556                            $EOPCT
1114  004570  104400  004635                    TYPE    ,$ENDMG             ;;TYPE "END PASS #"
1115  004574  013746  001210                    MOV     $PASS,-(SP)         ;;SAVE $PASS FOR TYPEOUT
1116  004600  104404                            TYPDS                       ;;GO TYPE--DECIMAL ASCII WITH SIGN
1117  004602  104400  004632                    TYPE    ,$ENULL             ;;TYPE A NULL CHARACTER
1118  004606  013700  000042          $GET42: MOV     @#42,R0             ;;GET MONITOR ADDRESS
1119  004612  001405                            BEQ     $DOAGN              ;;BRANCH IF NO MONITOR
1120  004614  000005                            RESET                       ;;CLEAR THE WORLD
1121  004616  004710                    $ENDAD: JSR     PC,(R0)             ;;GO TO MONITOR
1122  004620  000240                            NOP                         ;;SAVE ROOM
1123  004622  000240                            NOP                         ;;FOR
1124  004624  000240                            NOP                         ;;ACT11
1125  004626                            $DOAGN:
1126  004626  000137                            JMP     @(PC)+              ;;RETURN
1127  004630  002104                    $RTNAD: .WORD   INIT
1128  004632     377     377     000    $ENULL: .BYTE   -1,-1,0             ;;NULL CHARACTER STRING
1129  004635     015  042412  042116    $ENDMG: .ASCIZ  <15><12>/END PASS #/
1130  004642  050040  051501  020123
1131  004650  000043
1132
```

```
1133                                    ;;*****************************************************************
1134                                    ;*TEST 33        PUSH BUTTON TURNS LDR LIGHT ON AND OFF LOOP
1135                                    ;;*****************************************************************
1136  004652  000004                    TST33:  SCOPE
1137  004654  012706  001100            BEGIN1: MOV     #STACK,SP               ;LOAD STACK POINTER
1138  004660  005077  174700                    CLR     @PBSR                   ;CLEAR STATUS REGISTER
1139  004664  104400                            TYPE                            ;TYPE OUT EXPLANATION
1140  004666  007042                            MESS1                           ;OF TEST
1141  004670  005777  174670            1$:     TST     @PBSR                   ;STALL
1142  004674  100375                            BPL     1$                      ;TIME
1143  004676  017777  174664  174664            MOV     @PBR,@LDR               ;SET OR TURN OFF LIGHT
1144  004704  000771                            BR      1$                      ;BRANCH BACK
1145
1146                                    ;;*****************************************************************
1147                                    ;*TEST 34        TEST ENCODER WITH PUSH BUTTON
1148                                    ;;*****************************************************************
1149  004706  000004                    TST34:  SCOPE
1150  004710  012706  001100            BEGIN2: MOV     #STACK,SP               ;LOAD STACK POINTER
1151  004714  005077  174644                    CLR     @PBSR                   ;CLEAR STATUS REGISTER
1152  004720  104400                            TYPE                            ;TYPE OUT EXPLANATION
1153  004722  007140                            MESS3                           ;OF TEST
1154  004724  005077  174636            1$:     CLR     @PBR                    ;CLEAR PUSH BUTTON REGISTER
1155  004730  005777  174630            2$:     TST     @PBSR                   ;STALL
1156  004734  100375                            BPL     2$                      ;TIME
1157  004736  017737  174622  001600            MOV     @PBSR,TEMP1             ;READ STATUS REGISTER
1158  004744  042737  177741  001600            BIC     #177741,TEMP1           ;GET ENCODER BITS
1159  004752  012737  004776  001602            MOV     #TABLE,TEMP2            ;GET STARTING ADDRESS
1160  004760  063737  001600  001602            ADD     TEMP1,TEMP2             ;ADD OFFSET TO ADDRESS
1161  004766  017777  174610  174574            MOV     @TEMP2,@LDR             ;SET OR TURN OFF LAMP
1162  004774  000753                            BR      1$                      ;BRANCH BACK
1163
1164  004776  000001            TABLE:  BIT0
1165  005000  000002                    BIT1
1166  005002  000004                    BIT2
1167  005004  000010                    BIT3
1168  005006  000020                    BIT4
1169  005010  000040                    BIT5
1170  005012  000100                    BIT6
1171  005014  000200                    BIT7
1172  005016  000400                    BIT8
1173  005020  001000                    BIT9
1174  005022  002000                    BIT10
1175  005024  004000                    BIT11
1176  005026  010000                    BIT12
1177  005030  020000                    BIT13
1178  005032  040000                    BIT14
1179  005034  100000                    BIT15
```

```
1180
1181                                                    .SBTTL  ASCII MESSAGES
1182  005036  040506  046111  042105  EM1:    .ASCIZ  /FAILED TO LOAD AND READ LDR CORRECTLY/
1183  005044  052040  020117  047514
1184  005052  042101  040440  042116
1185  005060  051040  040505  020104
1186  005066  042114  020122  047503
1187  005074  051122  041505  046124
1188  005102  000131
1189  005104  040506  046111  042105  EM2:    .ASCIZ  /FAILED TO SET OR CLEAR PBR/
1190  005112  052040  020117  042523
1191  005120  020124  051117  041440
1192  005126  042514  051101  050040
1193  005134  051102     000
1194  005137     106  044501  042514  EM3:    .ASCIZ  /FAILED TO CLEAR PB FLAG/
1195  005144  020104  047524  041440
1196  005152  042514  051101  050040
1197  005160  020102  046106  043501
1198  005166     000
1199  005167     106  044501  042514  EM4:    .ASCIZ  /FAILED TO LOAD AND READ PB FLAG/
1200  005174  020104  047524  046040
1201  005202  040517  020104  047101
1202  005210  020104  042522  042101
1203  005216  050040  020102  046106
1204  005224  043501     000
1205  005227     106  044501  042514  EM5:    .ASCIZ  /FAILED TO CLEAR INTERRUPT ENABLE/
1206  005234  020104  047524  041440
1207  005242  042514  051101  044440
1208  005250  052116  051105  052522
1209  005256  052120  042440  040516
1210  005264  046102  000105
1211  005270  040506  046111  042105  EM6:    .ASCIZ  /FAILED TO SET INTERRUPT ENABLE/
1212  005276  052040  020117  042523
1213  005304  020124  047111  042524
1214  005312  051122  050125  020124
1215  005320  047105  041101  042514
1216  005326     000
1217  005327     105  051122  047117  EM7:    .ASCIZ  /ERRONEOUSLY CLEARED PB FLAG/
1218  005334  047505  051525  054514
1219  005342  041440  042514  051101
1220  005350  042105  050040  020102                  . .
1221  005356  046106  043501     000
1222  005363     106  044501  042514  EM10:   .ASCIZ  /FAILED TO GENERATE INTERRUPT WITH PB FLAG/
1223  005370  020104  047524  043440
1224  005376  047105  051105  052101
1225  005404  020105  047111  042524
1226  005412  051122  050125  020124
1227  005420  044527  044124  050040
1228  005426  020102  046106  043501
1229  005434     000
1230  005435     103  051114  050040  EM11:   .ASCIZ  /CLR PBSR FAILED TO CLEAR PB FLAG/
1231  005442  051502  020122  040506
1232  005450  046111  042105  052040
1233  005456  020117  046103  040505
1234  005464  020122  041120  043040
1235  005472  040514  000107
```

```
1236  005476  047111  042524  051122  EM12:   .ASCIZ  /INTERRUPT OK FAILED TO CLEAR INTR FLOP/
1237  005504  050125  020124  045517
1238  005512  043040  044501  042514
1239  005520  020104  047524  041440
1240  005526  042514  051101  044440
1241  005534  052116  020122  046106
1242  005542  050117     000
1243  005545     103  051114  050040  EM13:   .ASCIZ  /CLR PBSR FAILED TO CLEAR INTERRUPT ENABLE/
1244  005552  051502  020122  040506
1245  005560  046111  042105  052040
1246  005566  020117  046103  040505
1247  005574  020122  047111  042524
1248  005602  051122  050125  020124
1249  005610  047105  041101  042514
1250  005616     000
1251  005617     125  042516  050130  EM14:   .ASCIZ  /UNEXPECTED INTERRUPT/
1252  005624  041505  042524  020104
1253  005632  047111  042524  051122
1254  005640  050125  000124
1255  005644  044502  020103  040506  EM15:   .ASCIZ  /BIC FAILED TO CLEAR LDR /
1256  005652  046111  042105  052040
1257  005660  020117  046103  040505
1258  005666  020122  042114  020122
1259  005674     000
1260  005675     102  051511  043040  EM16:   .ASCIZ  /BIS FAILED TO SET LDR/
1261  005702  044501  042514  020104
1262  005710  047524  051440  052105
1263  005716  046040  051104     000
1264  005723     102  051511  020102  EM17:   .ASCIZ  /BISB FAILED TO SET LOWER BYTE OR LEAVE UPPER BYTE THE SAME/
1265  005730  040506  046111  042105
1266  005736  052040  020117  042523
1267  005744  020124  047514  042527
1268  005752  020122  054502  042524
1269  005760  047440  020122  042514
1270  005766  053101  020105  050125
1271  005774  042520  020122  054502
1272  006002  042524  052040  042510
1273  006010  051440  046501  000105
1274  006016  044502  041123  043040  EM20:   .ASCIZ  /BISB FAILED TO SET UPPER BYTE OR LEAVE LOWER BYTE THE SAME/
1275  006024  044501  042514  020104
1276  006032  047524  051440  052105
1277  006040  052440  050120  051105
1278  006046  041040  052131  020105
1279  006054  051117  046040  040505
1280  006062  042524  046040  053517
1281  006070  051105  041040  052131
1282  006076  020105  044124  020105
1283  006104  040523  042515     000
1284  006111     102  041511  020102  EM21:   .ASCIZ  /BICB FAILED TO CLEAR LOWER BYTE OR LEAVE UPPER BYTE THE SAME/
1285  006116  040506  046111  042105
1286  006124  052040  020117  046103
1287  006132  040505  020122  047514
1288  006140  042527  020122  054502
1289  006146  042524  047440  020122
1290  006154  042514  053101  020105
1291  006162  050125  042520  020122
```

```
1292  006170  054502  042524  052040
1293  006176  042510  051440  046501
1294  006204  000105
1295  006206  044502  041103  043040  EM22:   .ASCIZ  /BICB FAILED TO CLEAR UPPER BYTE OR LEAVE LOWER BYTE THE SAME/
1296  006214  044501  042514  020104
1297  006222  047524  041440  042514
1298  006230  051101  052440  050120
1299  006236  051105  041040  052131
1300  006244  020105  051117  046040
1301  006252  040505  042526  046040
1302  006260  053517  051105  041040
1303  006266  052131  020105  044124
1304  006274  020105  040523  042515
1305  006302      000
1306  006303      114  040517  044504  EM23:   .ASCIZ  /LOADING LDR CHANGED PBSR DATA/
1307  006310  043516  046040  051104
1308  006316  041440  040510  043516
1309  006324  042105  050040  051502
1310  006332  020122  040504  040524
1311  006340      000
1312  006341      114  040517  044504  EM24:   .ASCIZ  /LOADING PBSR CHANGED LDR DATA/
1313  006346  043516  050040  051502
1314  006354  020122  044103  047101
1315  006362  042507  020104  042114
1316  006370  020122  040504  040524
1317  006376      000
1318  006377      114  040517  044504  EM25:   .ASCIZ  /LOADING LDR CHANGED PBR DATA/
1319  006404  043516  046040  051104
1320  006412  041440  040510  043516
1321  006420  042105  050040  051102
1322  006426  042040  052101  000101
1323  006434  047514  042101  047111  EM26:   .ASCIZ  /LOADING PBR CHANGED LDR DATA/
1324  006442  020107  041120  020122
1325  006450  044103  047101  042507
1326  006456  020104  042114  020122
1327  006464  040504  040524      000
1328  006471      114  040517  044504  EM27:   .ASCIZ  /LOADING PBSR CHANGED PBR DATA/
1329  006476  043516  050040  051502
1330  006504  020122  044103  047101
1331  006512  042507  020104  041120
1332  006520  020122  040504  040524
1333  006526      000
1334  006527      114  040517  044504  EM30:   .ASCIZ  /LOADING PBR CHANGED PBSR DATA/
1335  006534  043516  050040  051102
1336  006542  041440  040510  043516
1337  006550  042105  050040  051502
1338  006556  020122  040504  040524
1339  006564      000
1340  006565      105  051122  041520  DH1:    .ASCIZ  /ERRPC   LDR   EXPECTED   ACTUAL/
1341  006572  020040  046040  051104
1342  006600  020040  042440  050130
1343  006606  041505  042524  020104
1344  006614  020040  041501  052524
1345  006622  046101      000
1346  006625      105  051122  041520  DH2:    .ASCIZ  /ERRPC   PBR   EXPECTED   ACTUAL/
1347  006632  020040  050040  051102
```

```
  1348  006640  020040  042440  050130
  1349  006646  041505  042524  020104
  1350  006654  020040  041501  052524
  1351  006662  046101     000
  1352  006665     105  051122  041520  DH3:    .ASCIZ  /ERRPC   PBSR   EXPECTED   ACTUAL/
  1353  006672  020040  050040  051502
  1354  006700  020122  020040  054105
  1355  006706  042520  052103  042105
  1356  006714  020040  040440  052103
  1357  006722  040525  000114
  1358  006726  051105  050122  020103  DH4:    .ASCIZ  /ERRPC  PBSR    LOCATION OF FAILURE/
  1359  006734  050040  051502  020122
  1360  006742  020040  046040  041517
  1361  006750  052101  047511  020116
  1362  006756  043117  043040  044501
  1363  006764  052514  042522     000
  1364          006772                  .EVEN
  1365
  1366  006772  001116  001570  001124  DT1:    $ERRPC, LDR, $GDDAT, $BDDAT, 0
  1367  007000  001126  000000
  1368  007004  001116  001566  001124  DT2:    $ERRPC, PBR, $GDDAT, $BDDAT, 0
  1369  007012  001126  000000
  1370  007016  001116  001564  001124  DT3:    $ERRPC, PBSR, $GDDAT, $BDDAT, 0
  1371  007024  001126  000000
  1372  007030  001116  001564  001124  DT4:    $ERRPC,PBSR,$GDDAT,0
  1373  007036  000000                  DF1:    0
  1374  007040  000000
  1375
  1376  007042  005015  051120  051505  MESS1:  .ASCIZ  <15><12>/PRESS PUSH BUTTON. CORRESPONDING LIGHT SHOULD COMPLEMENT./<15><
  1377  007050  020123  052520  044123
  1378  007056  041040  052125  047524
  1379  007064  027116  041440  051117
  1380  007072  042522  050123  047117
  1381  007100  044504  043516  046040
  1382  007106  043511  052110  051440
  1383  007114  047510  046125  020104
  1384  007122  047503  050115  042514
  1385  007130  042515  052116  006456
  1386  007136  000012
  1387  007140  005015  051120  051505  MESS3:  .ASCIZ  <15><12>/PRESS PUSH BUTTON. ENCODER SHOULD SET CORRESPONDING LIGHT./<15>
  1388  007146  020123  052520  044123
  1389  007154  041040  052125  047524
  1390  007162  027116  042440  041516
  1391  007170  042117  051105  051440
  1392  007176  047510  046125  020104
  1393  007204  042523  020124  047503
  1394  007212  051122  051505  047520
  1395  007220  042116  047111  020107
  1396  007226  044514  044107  027124
  1397  007234  005015     000
  1398  007237     015  046412  044501  HEAD1:  .ASCIZ  <15><12>/MAINDEC-11-DZLKA-A: PUSH BUTTON OPTION DIAGNOSTIC/<15><12>
  1399  007244  042116  041505  030455
  1400  007252  026461  055104  045514
  1401  007260  026501  035101  050040
  1402  007266  051525  020110  052502
  1403  007274  052124  047117  047440
```

```
 1404  007302  052120  047511  020116
 1405  007310  044504  043501  047516
 1406  007316  052123  041511  005015
 1407  007324     000
 1408          007326                  .EVEN
 1409
 1410                                  .SBTTL  TTY INPUT ROUTINE
 1411
 1412                                  ;;*****************************************************************
 1413                                  .ENABL  LSB
 1414
 1415                                  ;;*****************************************************************
 1416                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 1417                                  ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
 1418                                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
 1419                                  ;*WHEN OPERATING IN TTY FLAG MODE.
 1420  007326  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
 1421  007334  001074                         BNE     15$             ;;BRANCH IF NO
 1422  007336  105777  171602                 TSTB    @$TKS           ;;CHAR THERE?
 1423  007342  100071                         BPL     15$             ;;IF NO, DON'T WAIT AROUND
 1424  007344  117746  171576                 MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
 1425  007350  042716  177600                 BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
 1426  007354  022726  000007                 CMP     #7,(SP)+        ;;IS IT A CONTROL G?
 1427  007360  001062                         BNE     15$             ;;NO, RETURN TO USER
 1428  007362  123727  001134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
 1429  007370  001456                         BEQ     15$             ;;BRANCH IF YES
 1430
 1431  007372  104400  010053                 TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
 1432  007376  104400  010060  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
 1433  007402  013746  000176                 MOV     $SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
 1434  007406  104401                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 1435  007410  104400  010071                 TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
 1436  007414  005046          19$:    CLR     -(SP)           ;;CLEAR COUNTER
 1437  007416  005046                  CLR     -(SP)           ;;THE NEW SWR
 1438  007420  105777  171520          7$:     TSTB    @$TKS           ;;CHAR THERE?
 1439  007424  100375                         BPL     7$              ;;IF NOT TRY AGAIN
 1440
 1441  007426  117746  171514                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
 1442  007432  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 1443
 1444
 1445
 1446  007436  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
 1447  007442  001005                         BNE     10$             ;;BRANCH IF NOT
 1448  007444  104400  010046                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
 1449  007450  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
 1450  007454  000757                         BR      19$             ;;LET'S TRY IT AGAIN
 1451
 1452
 1453  007456  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
 1454  007462  001022                         BNE     16$             ;;BRANCH IF NO
 1455  007464  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
 1456  007470  001403                         BEQ     11$             ;;BRANCH IF YES
 1457  007472  016677  000002  171440        MOV     2(SP),@SWR      ;;SAVE NEW SWR
 1458  007500  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
 1459  007504  104400  001177          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
```

```
1460  007510  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
1461  007516  001003                          BNE     15$             ;;BRANCH IF NOT
1462  007520  012777  000100  171416          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
1463  007526  000002                  15$:    RTI                     ;;RETURN
1464  007530  004737  011516          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
1465  007534  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
1466  007540  002420                          BLT     18$             ;;BRANCH IF YES
1467  007542  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
1468  007546  003015                          BGT     18$             ;;BRANCH IF YES
1469  007550  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
1470  007554  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
1471  007560  001403                          BEQ     17$             ;;BRANCH IF YES
1472  007562  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
1473  007564  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
1474  007566  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
1475  007570  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
1476  007574  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
1477  007600  000707                          BR      7$              ;;GET THE NEXT ONE
1478  007602  104400  001176          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
1479  007606  000720                          BR      20$             ;;SIMULATE CONTROL-U
1480                                .DSABL  LSB
1481
1482
1483                                ;;**************************************************************
1484                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
1485                                ;*CALL:
1486                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
1487                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
1488                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
1489                                ;
1490
1491  007610  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
1492  007612  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
1493  007620  105777  171320          1$:     TSTB    @$TKS           ;;WAIT FOR
1494  007624  100375                          BPL     1$              ;;A CHARACTER
1495  007626  117766  171314  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
1496  007634  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
1497  007642  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
1498  007650  001013                          BNE     3$              ;;BRANCH IF NO
1499  007652  105777  171266          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
1500  007656  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
1501  007660  117746  171262                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
1502  007664  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
1503  007670  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
1504  007674  001366                          BNE     2$              ;;IF NOT DISCARD IT
1505  007676  000750                          BR      1$              ;;YES, RESUME
1506  007700  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
1507  007706  002407                          BLT     4$              ;;BRANCH IF YES
1508  007710  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
1509  007716  003003                          BGT     4$              ;;BRANCH IF YES
1510  007720  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
1511  007726  000002                  4$:     RTI                     ;;GO BACK TO USER
1512                                ;;**************************************************************
1513                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
1514                                ;*CALL:
1515                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
```

```
1516                                    ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
1517                                    ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
1518
1519  007730  010346                    $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
1520  007732  012703  010036            1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
1521  007736  022703  010046            2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
1522  007742  101405                            BLOS    4$              ;;BR IF YES
1523  007744  104407                            RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
1524  007746  112613                            MOVB    (SP)+,(R3)      ;;GET CHARACTER
1525  007750  122713  000177            10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
1526  007754  001003                            BNE     3$              ;;SKIP IF NOT
1527  007756  104400  001176            4$:     TYPE    ,$QUES          ;;TYPE A '?'
1528  007762  000763                            BR      1$              ;;CLEAR THE BUFFER AND LOOP
1529  007764  111337  010034            3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
1530  007770  104400  010034                    TYPE    ,9$
1531  007774  122723  000015                    CMPB    #15,(R3)+       ;;CHECK FOR RETURN
1532  010000  001356                            BNE     2$              ;;LOOP IF NOT RETURN
1533  010002  105063  177777                    CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
1534  010006  104400  001200                    TYPE    ,$LF            ;;TYPE A LINE FEED
1535  010012  012603                            MOV     (SP)+,R3        ;;RESTORE R3
1536  010014  011646                            MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
1537  010016  016666  000004  000002            MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
1538  010024  012766  010036  000004            MOV     #$TTYIN,4(SP)
1539  010032  000002                            RTI                     ;;RETURN
1540  010034     000                    9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
1541  010035     000                            .BYTE   0               ;;TERMINATOR
1542  010036  000010                    $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
1543  010046  052536  005015     000    $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
1544  010053     136  006507  000012    $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
1545  010060  005015  053523  020122    $MSWR:  .ASCIZ  <15><12>/SWR = /
1546  010066  020075     000
1547  010071     040  047040  053505    $MNEW:  .ASCIZ  /  NEW = /
1548  010076  036440  000040
```

```
                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

010102                          $TYPDS:
010102  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
010104  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
010106  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
010110  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
010112  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
010114  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
010120  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
010124  100004                          BPL     1$              ;;BR IF INPUT IS POS.
010126  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
010130  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
010136  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
010140  012703  010316                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
010144  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
010150  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
010152  016001  010306                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
010156  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
010160  002402                          BLT     4$              ;;BR IF DONE
010162  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
010164  000774                          BR      3$
010166  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
010170  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
010172  001002                          BNE     5$              ;;FALL THROUGH IF 0
010174  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
010176  100407                          BMI     7$              ;;BR IF YES
010200  106316                  5$:     ASLB    (SP)            ;;MSD?
010202  103003                          BCC     6$              ;;BR IF NO
010204  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
010212  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
010216  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
010222  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
010224  005720                          TST     (R0)+           ;;JUST INCREMENTING
010226  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
010232  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
010234  003002                          BGT     8$              ;;GO TO EXIT
010236  010502                          MOV     R5,R2           ;;GET THE LSD
010240  000764                          BR      6$              ;;GO CHANGE TO ASCII
010242  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
010244  100003                          BPL     9$              ;;BR IF NO
010246  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
010254  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
010256  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
010260  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
010262  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
 1605  010264  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 1606  010266  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 1607  010270  104400  010316                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 1608  010274  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 1609  010302  012616                          MOV     (SP)+,(SP)
 1610  010304  000002                          RTI                     ;;RETURN TO USER
 1611  010306  023420                  $DTBL:  10000.
 1612  010310  001750                          1000.
 1613  010312  000144                          100.
 1614  010314  000012                          10.
 1615  010316  000004                  $DBLK:  .BLKW   4
```

```
                                .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

010326  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
010330  016666  000004  000002  MOV     4(SP),2(SP)     ;;INPUT NUMBER
010336  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
010340  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
010342  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
010344  104410          1$:     RDLIN                   ;;READ AN ASCIZ LINE
010346  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
010350  010037  010454          MOV     R0,5$           ;;AND SAVE IT
010354  005001                  CLR     R1              ;;CLEAR DATA WORD
010356  005002                  CLR     R2
010360  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
010362  001420                  BEQ     3$              ;;IF ZERO GET OUT
010364  122716  000060          CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
010370  003026                  BGT     4$              ;;IS AN OCTAL DIGIT
010372  122716  000067          CMPB    #'7,(SP)
010376  002423                  BLT     4$
010400  006301                  ASL     R1              ;;*2
010402  006102                  ROL     R2
010404  006301                  ASL     R1              ;;*4
010406  006102                  ROL     R2
010410  006301                  ASL     R1              ;;*8
010412  006102                  ROL     R2
010414  042716  177770          BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
010420  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
010422  000756                  BR      2$              ;;LOOP
010424  005726          3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
010426  010166  000012          MOV     R1,12(SP)       ;;SAVE THE RESULT
010432  010237  010464          MOV     R2,$HIOCT
010436  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
010440  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
010442  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
010444  000002                  RTI                     ;;RETURN
010446  005726          4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
010450  105010                  CLRB    (R0)            ;;SET A TERMINATOR
010452  104400                  TYPE                    ;;TYPE UP THRU THE BAD CHAR.
010454  000000          5$:     .WORD   0
010456  104400  001176          TYPE    ,SQUES          ;;"?" "CR" & "LF"
010462  000730                  BR      1$              ;;TRY AGAIN
010464  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
```

```
                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;;*****************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW11=1         INHIBIT ITERATIONS
                                ;*SW09=1         LOOP ON ERROR
                                ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

010466                          $SCOPE:
010466  104406                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
010470  104406                          CKSWR
010472  032777  040000  170440  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
010500  001114                          BNE     $OVER           ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
010502  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
010504  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
010510  012737  010530  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
010516  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
010522  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
010526  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
010530  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
010532  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
010536  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
010540                          6$:;#####END OF CODE FOR THE XOR TESTER#####
010540  032777  000400  170372          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
010546  001404                          BEQ     2$              ;;BR IF NO
010550  127737  170364  001102          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
010556  001465                          BEQ     $OVER           ;;BR IF YES
010560  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
010564  001421                          BEQ     3$              ;;BR IF NO
010566  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
010574  101015                          BHI     3$              ;;BR IF NO
010576  032777  001000  170334          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
010604  001404                          BEQ     4$              ;;BR IF NO
010606  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
010614  000446                          BR      $OVER
010616  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
010622  005037  001166                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
010626  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
010630  032777  004000  170302  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
010636  001011                          BNE     1$              ;;BR IF YES
010640  005737  001210                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
010644  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
010646  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
010652  023737  001166  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
010660  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
010662  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
010670  013737  010746  001166          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
010676  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
```

```
 1725  010702  113737  001102  001206          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
 1726  010710  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 1727  010714  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 1728  010720  005037  001170                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 1729  010724  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 1730  010732  013777  001102  170202 $OVER:   MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 1731  010740  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 1732  010744  000002                          RTI                     ;;FIXES PS
 1733  010746  003720                 $MXCNT: 2000.                   ;;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

010750                          $ERROR:
010750  104406                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
010752  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
010756  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
010760  013777  001102  170154          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
010766  032777  002000  170144          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
010774  001402                          BEQ     1$              ;;NO - SKIP
010776  104400  001172                  TYPE    ,$BELL          ;;RING BELL
011002  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
011006  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
011012  162737  000002  001116          SUB     #2,$ERRPC
011020  117737  170072  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
011026  032777  020000  170104          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
011034  001004                          BNE     20$             ;;SKIP TYPEOUTS
011036  004737  011150                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
011042  104400  001177                  TYPE    ,$CRLF
011046                          20$:
011046  122737  000001  001222          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
011054  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
011056  113737  001114  011070          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
011064  004737  011604                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
011070     000                  21$:    .BYTE   0
011071     000                          .BYTE   0
011072  000777                  22$:    BR      22$             ;;APT ERROR LOOP
011074  005777  170040          2$:     TST     @SWR            ;;HALT ON ERROR
011100  100002                          BPL     3$              ;;SKIP IF CONTINUE
011102  000000                          HALT                    ;;HALT ON ERROR!
011104  104406                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
011106  032777  001000  170024  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
011114  001402                          BEQ     4$              ;;BR IF NO
011116  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
011122  005737  001170          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
011126  001402                          BEQ     5$              ;;BR IF NONE
011130  013716  001170                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
011134                          5$:
011134  022737  004616  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
011142  001001                          BNE     6$              ;;BRANCH IF NO
011144  000000                          HALT                    ;;YES
011146                          6$:
011146  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.

011150                                  $ERRTYP:
011150  104400  001177                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
011154  010046                          .       MOV     R0,-(SP)        ;;SAVE R0
011156  005000                          .       CLR     R0              ;;PICKUP THE ITEM INDEX
011160  153700  001114                  .       BISB    @#$ITEMB,R0
011164  001004                          .       BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                        ;;TYPE THE PC OF THE ERROR
011166  013746  001116                          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;ERROR ADDRESS
011172  104401                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
011174  000426                                  BR      6$              ;;GET OUT
011176  005300                          1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
011200  006300                                  ASL     R0              ;;      WORK FOR THE ERROR TABLE
011202  006300                                  ASL     R0
011204  006300                                  ASL     R0
011206  062700  001264                          ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
011212  012037  011222                          MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
011216  001404                                  BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
011220  104400                                  TYPE                    ;;TYPE THE "ERROR MESSAGE"
011222  000000                          2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
011224  104400  001177                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
011230  012037  011240                  3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
011234  001404                                  BEQ     5$              ;;SKIP TYPEOUT IF 0
011236  104400                                  TYPE                    ;;TYPE THE "DATA HEADER"
011240  000000                          4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
011242  104400  001177                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
011246  011000                          5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
011250  001004                                  BNE     7$              ;;GO TYPE THE DATA
011252  012600                          6$:     MOV     (SP)+,R0        ;;RESTORE R0
011254  104400  001177                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
011260  000207                                  RTS     PC              ;;RETURN
011262                                  7$:
011262  013046                                  MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
011264  104401                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
011266  005710                                  TST     (R0)            ;;IS THERE ANOTHER NUMBER?
011270  001770                                  BEQ     6$              ;;BR IF NO
011272  104400  011300                          TYPE    ,8$             ;;TYPE TWO(2) SPACES
011276  000771                                  BR      7$              ;;LOOP
011300  020040      000               8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
011304                                  .EVEN
```

```
                                .SBTTL  TYPE ROUTINE

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

011304  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
011310  100002                          BPL     1$              ;;BR IF YES
011312  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
011314  000430                          BR      3$              ;;LEAVE
011316  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
011320  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
011324  122737  000001  001222          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
011332  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
011334  132737  000100  001223          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
011342  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
011344  010037  011354                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
011350  004737  011574                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
011354  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
011356  132737  000040  001223  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
011364  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
011366  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
011370  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
011372  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
011374  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
011376  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
011402  000002                          RTI                     ;;RETURN
011404  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
011410  001430                          BEQ     8$
011412  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
011416  001006                          BNE     5$
011420  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
011422  104400                          TYPE                    ;;TYPE A CR AND LF
011424  001177                          $CRLF
011426  105037  011562                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
011432  000755                          BR      2$              ;;GET NEXT CHARACTER
011434  004737  011516          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
011440  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
011444  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
011446  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
011452  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
011456  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
011460  004737  011516                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
```

```
1892  011464  105337  011562              DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
1893  011470  000770                      BR      7$              ;;LOOP
1894
1895                              ;HORIZONTAL TAB PROCESSOR
1896
1897  011472  112716  000040      8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
1898  011476  004737  011516      9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
1899  011502  132737  000007  011562      BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
1900  011510  001372                      BNE     9$              ;;TAB STOP
1901  011512  005726                      TST     (SP)+           ;;POP SPACE OFF STACK
1902  011514  000724                      BR      2$              ;;GET NEXT CHARACTER
1903  011516  105777  167426      $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
1904  011522  100375                      BPL     $TYPEC
1905  011524  116677  000002  167420      MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
1906  011532  122766  000015  000002      CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
1907  011540  001003                      BNE     1$              ;;BRANCH IF NO
1908  011542  105037  011562              CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
1909  011546  000406                      BR      $TYPEX          ;;EXIT
1910  011550  122766  000012  000002  1$: CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
1911  011556  001402                      BEQ     $TYPEX          ;;BRANCH IF YES
1912  011560  105227                      INCB    (PC)+           ;;COUNT THE CHARACTER
1913  011562  000000      $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
1914  011564  000207      $TYPEX: RTS     PC
1915
```

```
                                        .SBTTL  APT COMMUNICATIONS ROUTINE
                                        ;;**************************************************************************
011566  112737  000001  012032  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
011574  112737  000001  012030  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
011602  000403                  BR      $ATYC
011604  112737  000001  012032  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
011612                          $ATYC:
011612  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011614  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
011616  105737  012030          TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
011622  001450                  BEQ     5$              ;;IF NOT:  BR
011624  122737  000001  001222  CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
011632  001031                  BNE     3$              ;;IF NOT:  BR
011634  132737  000100  001223  BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
011642  001425                  BEQ     3$              ;;IF NOT:  BR
011644  017600  000004          MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
011650  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
011656  005737  001202          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
011662  001375                  BNE     1$              ;;IF NOT:  WAIT
011664  010037  001216          MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
011670  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
011672  001376                  BNE     2$
011674  163700  001216          SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
011700  006200                  ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
011702  010037  001220          MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
011706  012737  000004  001202  MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
011714  000413                  BR      5$
011716  017637  000004  011742  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
011724  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
011732  013746  177776          MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
011736  004737  011304          JSR     PC,$TYPE        ;;CALL TYPE MACRO
011742  000000                  4$:     .WORD   0
011744                          5$:
011744  105737  012032          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
011750  001416                  BEQ     12$             ;;IF NOT:  BR
011752  005737  001222          TST     $ENV            ;;RUNNING UNDER APT?
011756  001413                  BEQ     12$             ;;IF NOT:  BR
011760  005737  001202          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
011764  001375                  BNE     11$             ;;IF NOT:  WAIT
011766  017637  000004  001204  MOV     @4(SP),$FATAL   ;;GET ERROR #
011774  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
012002  005237  001202          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
012006  105037  012032          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
012012  105037  012031          CLRB    $LFLG           ;;CLEAR LOG FLAG
012016  105037  012030          CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
012022  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
012024  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
012026  000207                  RTS     PC              ;;RETURN
012030  000             $MFLG:  .BYTE   0               ;;MESSG. FLAG
012031  000             $LFLG:  .BYTE   0               ;;LOG FLAG
012032  000             $FFLG:  .BYTE   0               ;;FATAL FLAG
012034                          .EVEN
000200                  APTSIZE=200
000001                  APTENV=001
```

```
                        000100                  APTSPOOL=100
                        000040                  APTCSUP=040
                                                .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

 012034  017646  000000        $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 012040  116637  000001 012257          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
 012046  112637  012261                 MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 012052  062716  000002                 ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 012056  000406                         BR      $TYPON
 012060  112737  000001 012257 $TYPOC:  MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
 012066  112737  000006 012261         MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 012074  112737  000005 012256 $TYPON:  MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
 012102  010346                         MOV     R3,-(SP)        ;;SAVE R3
 012104  010446                         MOV     R4,-(SP)        ;;SAVE R4
 012106  010546                         MOV     R5,-(SP)        ;;SAVE R5
 012110  113704  012261                 MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 012114  005404                         NEG     R4
 012116  062704  000006                 ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
 012122  110437  012260                 MOVB    R4,$OMODE       ;;SAVE IT FOR USE
 012126  113704  012257                 MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
 012132  016605  000012                 MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
 012136  005003                         CLR     R3              ;;CLEAR THE OUTPUT WORD
 012140  006105                 1$:     ROL     R5              ;;ROTATE MSB INTO "C"
 012142  000404                         BR      3$              ;;GO DO MSB
 012144  006105                 2$:     ROL     R5              ;;FORM THIS DIGIT
 012146  006105                         ROL     R5
 012150  006105                         ROL     R5
 012152  010503                         MOV     R5,R3
 012154  006103                 3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
 012156  105337  012260                 DECB    $OMODE          ;;TYPE THIS DIGIT?
 012162  100016                         BPL     7$              ;;BR IF NO
 012164  042703  177770                 BIC     #177770,R3      ;;GET RID OF JUNK
 012170  001002                         BNE     4$              ;;TEST FOR 0
```

```
2028  012172  005704                        TST     R4              ;;SUPPRESS THIS 0?
2029  012174  001403                        BEQ     5$              ;;BR IF YES
2030  012176  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2031  012200  052703  000060              BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2032  012204  052703  000040        5$:   BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2033  012210  110337  012254              MOVB    R3,8$           ;;SAVE FOR TYPING
2034  012214  104400  012254              TYPE    ,8$             ;;GO TYPE THIS DIGIT
2035  012220  105337  012256        7$:   DECB    $OCNT           ;;COUNT BY 1
2036  012224  003347                        BGT     2$              ;;BR IF MORE TO DO
2037  012226  002402                        BLT     6$              ;;BR IF DONE
2038  012230  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2039  012232  000744                        BR      2$              ;;GO DO THE LAST DIGIT
2040  012234  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
2041  012236  012604                        MOV     (SP)+,R4        ;;RESTORE R4
2042  012240  012603                        MOV     (SP)+,R3        ;;RESTORE R3
2043  012242  016666  000002  000004       MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2044  012250  012616                        MOV     (SP)+,(SP)
2045  012252  000002                        RTI                     ;;RETURN
2046  012254     000                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2047  012255     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2048  012256     000                $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2049  012257     000                $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2050  012260  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
2051
2052                                    .SBTTL  TRAP DECODER
2053
2054                                    ;;**********************************************************************
2055                                    ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2056                                    ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2057                                    ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2058                                    ;*GO TO THAT ROUTINE.
2059
2060  012262  010046                    $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2061  012264  016600  000002                    MOV     2(SP),R0        ;;GET TRAP ADDRESS
2062  012270  005740                            TST     -(R0)           ;;BACKUP BY 2
2063  012272  111000                            MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2064  012274  006300                            ASL     R0              ;;POSITION FOR INDEXING
2065  012276  016000  012304                    MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2066  012302  000200                            RTS     R0              ;;GO TO ROUTINE
2067
2068                                    .SBTTL  TRAP TABLE
2069
2070                                    ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2071                                    ;*BY THE "TRAP" INSTRUCTION.
2072
2073                                    ;       ROUTINE
2074                                    ;       -------
2075  012304                            $TRPAD:
2076  012304  011304                            $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
2077  012306  012060                            $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2078  012310  012034                            $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2079  012312  012074                            $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2080  012314  010102                            $TYPDS  ;;CALL=TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
2081
2082  012316  007376                            $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
2083
2084  012320  007326                            $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
2085  012322  007610                            $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
2086  012324  007730                            $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
2087  012326  010326                            $RDOCT  ;;CALL=RDOCT    TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES
                                        ;;**********************************************************************
                                        ;POWER DOWN ROUTINE
012330  012737  012470  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
012336  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
012344  010046                        MOV     R0,-(SP)         ;;PUSH R0 ON STACK
012346  010146                        MOV     R1,-(SP)         ;;PUSH R1 ON STACK
012350  010246                        MOV     R2,-(SP)         ;;PUSH R2 ON STACK
012352  010346                        MOV     R3,-(SP)         ;;PUSH R3 ON STACK
012354  010446                        MOV     R4,-(SP)         ;;PUSH R4 ON STACK
012356  010546                        MOV     R5,-(SP)         ;;PUSH R5 ON STACK
012360  017746  166554                MOV     @SWR,-(SP)       ;;PUSH @SWR ON STACK
012364  010637  012474                MOV     SP,$SAVR6        ;;SAVE SP
012370  012737  012402  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
012376  000000                        HALT
012400  000776                        BR      .-2              ;;HANG UP
                                        ;;**********************************************************************
                                        ;POWER UP ROUTINE
012402  012737  012470  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
012410  013706  012474                MOV     $SAVR6,SP        ;;GET SP
012414  005037  012474                CLR     $SAVR6           ;;WAIT LOOP FOR THE TTY
012420  005237  012474          1$:   INC     $SAVR6           ;;WAIT FOR THE INC
012424  001375                        BNE     1$               ;;OF  WORD
012426  012677  166506                MOV     (SP)+,@SWR       ;;POP STACK INTO @SWR
012432  012605                        MOV     (SP)+,R5         ;;POP STACK INTO R5
012434  012604                        MOV     (SP)+,R4         ;;POP STACK INTO R4
012436  012603                        MOV     (SP)+,R3         ;;POP STACK INTO R3
012440  012602                        MOV     (SP)+,R2         ;;POP STACK INTO R2
012442  012601                        MOV     (SP)+,R1         ;;POP STACK INTO R1
012444  012600                        MOV     (SP)+,R0         ;;POP STACK INTO R0
012446  012737  012330  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
012454  012737  000340  000026        MOV     #340,@#PWRVEC+2  ;;PRIO:7
012462  104400                        TYPE                     ;REPORT THE POWER FAILURE
012464  012476          $PWRMG: .WORD   $POWER                   ;;POWER FAIL MESSAGE POINTER
012466  000002                        RTI
012470  000000          $ILLUP: HALT                             ;;THE POWER UP SEQUENCE WAS STARTED
012472  000776                  BR      .-2                      ;;  BEFORE THE POWER DOWN WAS COMPLETE
012474  000000          $SAVR6: 0                                ;;PUT THE SP HERE
012476  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
012504  000122
                                        .EVEN
000001                                  .END
```

```
ABASE = 160060          303#    417     458     624     625     626
ACDW1 = 000000          417     460
ACDW2 = 000000          417
ACPUOP= 000000          417     432
ADDW0 = 000C00          417
ADDW1 = 000000          417
ADDW10= 000000          417
ADDW11= 000000          417
ADDW12= 000000          417
ADDW13= 000000          417
ADDW14= 000000          417
ADDW15= 000000          417
ADDW2 = 000000          417
ADDW3 = 000000          417
ADDW4 = 000000          417
ADDW5 = 000000          417
ADDW6 = 000000          417
ADDW7 = 000000          417
ADDW8 = 000000          417
ADDW9 = 000000          417
ADEVCT= 000000          417     423
ADEVM = 000000          417     459
AENV  = 000000          417     428
AENVM = 000000          417     429
AFATAL= 000000          417     420
AMADR1= 000000          417     445
AMADR2= 000000          417     449
AMADR3= 000000          417     452
AMADR4= 000000          417     455
AMAMS1= 000000          417     439
AMAMS2= 000000          417     447
AMAMS3= 000000          417     450
AMAMS4= 000000          417     453
AMSGAD= 000000          417     425
AMSGLG= 000000          417     426
AMSGTY= 000000          417     419
AMTYP1= 000000          417     440
AMTYP2= 000000          417     448
AMTYP3= 000000          417     451
AMTYP4= 000000          417     454
APASS = 000000          417     422
APRIOR= 000200          305#    417
APTCSU= 000040         1867    1973#
APTENV= 000001         1765    1860    1929    1971#
APTSIZ= 000200          679    1970#
APTSPO= 000100         1862    1931    1972#
ASWREG= 000000          417     430
ATESTN= 000000          417     421
AUNIT = 000000          417     424
AUSWR = 000000          417     431
AVECT1= 100360          304#    417     456     627     628
AVECT2= 000000          417     457
BEGIN   001620          329     639#
BEGIN1  004654          330    1137#
BEGIN2  004710          331    1150#
BITSAV  001606          633#    952*    955     956     961*    967*    969     970     975*
```

```
BIT0   = 000001            288#    737     765     952     967    1046    1056    1075    1164
BIT00  = 000001            278#    288
BIT01  = 000002            277#    287
BIT02  = 000004            276#    286
BIT03  = 000010            275#    285
BIT04  = 000020            274#    284
BIT05  = 000040            273#    283
BIT06  = 000100            272#    282
BIT07  = 000200            271#    281
BIT08  = 000400            270#    280    1700
BIT09  = 001000            269#    279    1708    1776
BIT1   = 000002            287#   1165
BIT10  = 002000            268#   1174    1753
BIT11  = 004000            267#   1175    1715
BIT12  = 010000            266#   1090    1176
BIT13  = 020000            265#   1177    1760
BIT14  = 040000            264#    851     864     876     889     891     892     913     914     940    1178    1686
BIT15  = 100000            263#    787     800     814     827     840     889     892     914     940    1035    1179
BIT2   = 000004            286#   1166
BIT3   = 000010            285#   1167
BIT4   = 000020            284#   1168
BIT5   = 000040            283#   1169
BIT6   = 000100            282#   1170
BIT7   = 000200            281#   1171
BIT8   = 000400            280#   1172
BIT9   = 001000            279#   1173
BPTVEC= 000014             295#
CKSWR  = 104406           1684    1685    1749    1775    2084#
CR     = 000015            203#   1906    1916
CRLF   = 000200            204#   1877    1916
DDISP  = 177570            210#    394     667
DF1      007040            484     490     496     502     508     514     521     527     533     539     545     551     557
                           563     569     575     581     587     593     599     605     611     617     623    1374#
DH1      006565            482     597     609    1340#
DH2      006625            488     603     615    1346#
DH3      006665            494     500     506     512     519     525     531     537     543     555     561     567     573
                           579     585     591     621    1352#
DH4      006726            549    1358#
DISPLA   001142            394#    667*    675*   1730*   1752*
DISPRE   000174            326#    675
DSWR   = 177570            209#    393     666
DT1      006772            483     598     604     610    1366#
DT2      007004            489     616    1368#
DT3      007016            495     501     507     513     520     526     532     538     544     556     562     568     574
                           580     586     592     622    1370#
DT4      007030            550    1372#
EMTVEC= 000030             298#    650*    651*
EM1      005036            481    1182#
EM10     005363            524    1222#
EM11     005435            530    1230#
EM12     005476            536    1236#
EM13     005545            542    1243#
EM14     005617            548    1251#
EM15     005644            554    1255#
EM16     005675            560    1260#
EM17     005723            566    1264#
```

```
EM2     005104            487    1189*
EM20    006016            572    1274*
EM21    006111            578    1284*
EM22    006206            584    1295*
EM23    006303            590    1306*
EM24    006341            596    1312*
EM25    006377            602    1318*
EM26    006434            608    1323*
EM27    006471            614    1328*
EM3     005137            493    1194*
EM30    006527            620    1334*
EM4     005167            499    1199*
EM5     005227            505    1205*
EM6     005270            511    1211*
EM7     005327            518    1217*
ERRVEC= 000004            291*    664     665*    676*   1691    1692*   1694*   1697*
GNS     = ****** U        325    2076    2077    2078    2079    2080    2082    2084    2085    2086    2087
GTSWR   = 104405         2082*
HEAD1   007237            686    1398*
HT      = 000011          201*   1875    1916
INIT    002104            684     687*   1127
IOTVEC= 000020            296*    648*    649*
LDR     001570            626*    688*    691*    705*    706     716*    717     727*    728     738*    739     751*    752
                          951*    953*    956*    957     968*    970*    971     982*    983*    985*    986     990*    991
                           995   1004*   1005*   1007*   1008    1012*   1013    1017    1029*   1034*   1036    1048*   1053*
                          1057   1082*   1143*   1161*   1366
LF      = 000012          202*   1910    1916
MESS1   007042           1140    1376*
MESS3   007140           1153    1387*
PBR     001566            625*    689*    690*    765*    766     776*    777     828     938*   1045*   1046*   1049    1054*
                          1056*   1067*   1070    1075*   1081*   1143    1154*   1368
PBSR    001564            624*    687*    786*    788*    789     798*    800*    801*    802     812*    814*    816     825*
                          827*    829     838*    840*    841     842     850*    852*    853     863*    864*    866     874*
                          876*    877*    878     887*    891*    892*    899*    905*    913*    914*    931*    939*    941*
                           942     943   1027*   1030    1035*   1069*   1074*   1076    1083*   1138*   1141    1151*   1155
                          1157    1370   1372
PC      =%000007          222*   1108*   1111*   1121*   1126    1464*   1762*   1768*   1826*   1865*   1884*   1891*   1898*
                          1912*   1914*   1948*   1965*
PIRQ    = 177772          208*
PIRQVE= 000240            302*
PR0     = 000000          225*
PR1     = 000040          226*
PR2     = 000100          227*
PR3     = 000140          228*
PR4     = 000200          229*
PR5     = 000240          230*
PR6     = 000300          231*
PR7     = 000340          232*    911     922
PS      = 177776          205*    206
PSW     = 177776          206*    893*    912*    923*    937*
PWRVEC= 000024            297*    654*    655*   2093*   2094*   2103*   2109*   2121*   2122*
RDCHR   = 104407         1523    2085*
RDLIN   = 104410         1635    2086*
RDOCT   = 104411         2087*
RESVEC= 000010            292*
R0      =%000000          213*   1118*   1121    1562    1572*   1576    1592    1593    1606*   1632    1636*   1637    1640
```

```
                    1660*  1663*  1798   1799*  1800*  1807*  1808*  1809*  1810*  1811*  1812   1817   1822*
                    1824*  1828   1830   1858   1859*  1864   1869   1872*  1925   1933*  1937   1938   1940*
                    1941*  1942   1964*  2060   2061*  2062   2063*  2064*  2065*  2066*  2095   2120*
R1      =%000001     214#  1563   1576*  1577   1581   1605*  1633   1638*  1646*  1648*  1650*  1653*  1656
                    1659*  1926   1963*  2096   2119*
R2      =%000002     215#  1564   1575*  1579*  1582   1589*  1590*  1591   1596*  1604*  1634   1639*  1647*
                    1649*  1651*  1657   1658*  2097   2118*
R3      =%000003     216#  1519   1520*  1521   1524*  1525   1529   1531   1533*  1535*  1565   1573*  1574*
                    1588*  1591*  1600*  1601*  1603*  2007   2016*  2022*  2023*  2026*  2031*  2032*  2033
                    2042*  2098   2117*
R4      =%000004     217#  2008   2010*  2011*  2012*  2013   2014*  2028   2030*  2038*  2041*  2099   2116*
R5      =%000005     218#  1566   1568*  1570*  1577*  1581*  1596   1602*  2009   2015*  2017*  2019*  2020*
                    2021*  2022   2040*  2100   2115*
R6      =%000006     219#   221    642*   643*   644
R7      =%000007     220#   222
SP      =%000006     221#   635    646*   664*   672*   676    898    918    929    1115*  1137*  1150*  1424*
                    1425*  1426   1433*  1436*  1437*  1441*  1442*  1446   1449*  1453   1455   1457   1458*
                    1465   1467   1469*  1470   1472*  1473*  1474*  1475*  1476*  1491*  1492*  1495*  1496*
                    1497   1501*  1502*  1503   1506   1508   1510*  1519*  1524   1535   1536*  1537*  1538*
                    1562*  1563*  1564*  1565*  1566*  1567*  1568   1571*  1584   1586*  1588   1598   1600
                    1602   1603   1604   1605   1606   1608*  1609*  1630*  1631*  1632*  1633*  1634*  1636
                    1640*  1642   1644   1652*  1653   1655   1656*  1658   1659   1660   1662   1691*  1694
                    1696   1697   1726   1727   1731*  1757   1778*  1781*  1798*  1803*  1824   1828*  1858*
                    1859   1869*  1871   1872   1873*  1875   1877   1879   1885   1887*  1889*  1897*  1901
                    1905   1906   1910   1925*  1926*  1933   1934*  1945   1946*  1947*  1957   1958*  1963
                    1964   1999*  2000   2001   2002*  2007*  2008*  2009*  2015   2040   2041   2042   2043*
                    2044*  2060*  2061   2095*  2096*  2097*  2098*  2099*  2100*  2101*  2102   2110*  2114
                    2115   2116   2117   2118   2119   2120
STACK = 001100       196#   646   1137   1150
STKLMT= 177774       207#
SWR     001140       393#   644    666*   668    674*   681*   1090   1420   1457*  1686   1700   1702   1708
                    1715   1753   1760   1772   1776   2101   2114*
SWREG   000176       327#   674   1420   1433
SW0     = 000001     260#
SW00    = 000001     250#   260
SW01    = 000002     249#   259
SW02    = 000004     248#   258
SW03    = 000010     247#   257
SW04    = 000020     246#   256
SW05    = 000040     245#   255
SW06    = 000100     244#   254
SW07    = 000200     243#   253
SW08    = 000400     242#   252
SW09    = 001000     241#   251
SW1     = 000002     259#
SW10    = 002000     240#
SW11    = 004000     239#
SW12    = 010000     238#
SW13    = 020000     237#
SW14    = 040000     236#
SW15    = 100000     235#
SW2     = 000004     258#
SW3     = 000010     257#
SW4     = 000020     256#
SW5     = 000040     255#
SW6     = 000100     254#
```

```
SW7    = 000200          253#
SW8    = 000400          252#
SW9    = 001000          251#
TABLE    004776         1159    1164#
TBITVE= 000014           293#
TEMP     001576          629#    828*    841*    906*    907*    908     912
TEMPL    001604          632#    991*    992*    994*   1013*   1014*   1016*
TEMP1    001600          630#    908*    909*    923    1157*   1158*   1160
TEMP2    001602          631#   1159*   1160*   1161
TKVEC  = 000060          300#
TPVEC  = 000064          301#
TRAPVE= 000034           299#    652*    653*
TRTVEC= 000014           294#
TST1     002212          703#
TST10    002546          779     785#
TST11    002610          791     797#
TST12    002654          804     810#
TST13    002724          818     824#
TST14    002772          831     837#
TST15    003040          844     849#
TST16    003102          855     860#
TST17    003152          868     873#
TST2     002250          708     714#
TST20    003216          880     886#
TST21    003302          904#
TST22    003464          920     928     936#
TST23    003546          945     950#
TST24    003640          966#
TST25    003716          981#
TST26    004042          997    1003#
TST27    004166         1019    1026#
TST3     002306          719     725#
TST30    004260         1038    1044#
TST31    004400         1060    1066#
TST32    004514         1089#
TST33    004652         1136#
TST34    004706         1149#
TST4     002344          730     736#
TST5     002410          749#
TST6     002456          763#
TST7     002514          768     774#
TYPDS  = 104404         1116    2080#
TYPE   = 104400          685    1114    1117    1139    1152    1431    1432    1435    1448    1459    1478    1527    1530
                        1534    1607    1664    1666    1755    1763    1797    1814    1816    1819    1821    1825    1832
                        1880    2034    2076#   2123
TYPOC  = 104401         1434    1805    1829    2077#
TYPON  = 104403         2079#
TYPOS  = 104402         2078#
UNEXP    001610          635#    697     930
VECT     001572          627#    692*    695*    697*    888*    910*    921*    930*
VECTR1   001574          628#    693*    694*    696*    698*    911*    922*
$APTHD   001000          353     359#
$ASTAT= ****** U        1951    1966
$ATYC    011612         1922    1924#
$ATY1    011566         1920#
$ATY3    011574         1865    1921#
```

```
$ATY4   011604          1768    1923#
$AUTOB  001134           390#   1428    1549
$BASE   001256           458#    687     688     689
$BOADR  001122           385#
$BODAT  001126           387#    706*    707     717*    718     728*    729     739*    740     752*    753     766*    767
                         777*    778     789*    790     802*    803     816*    817     829*    830     842*    843     853*
                         854     866*    867     878*    879     890*    926*    942*    943*    944     957*    958     971*
                         972     986*    987     995*    996    1008*   1009    1017*   1018    1030*   1031    1036*   1037
                        1049*   1050    1057*   1058    1059    1070*   1071    1076*   1077*   1078    1366    1368    1370
$BELL   001172           409#   1755    1788
$CDWI   001262           460#
$CHARC  011562          1882*   1892*   1899    1908*   1913#
$CKSWR  007326          1420#   2084
$CNTAG  001100           373#    641     642     650     656     657     658
$CM1  = 000002           405#    406#    407#
$CM2  = 000004           405#    406#    407#
$CM3  = 000002           403#    405
$CNTLG  010053          1431    1544#
$CNTLU  010046          1448    1543#
$CPUOP  001230           432#
$CRLF   001177           411#   1459    1543    1669    1763    1788    1797    1816    1821    1825    1881    1916
$DBLK   010316          1573    1607    1615#
$DEVCT  001212           423#
$DEVM   001260           459#
$DOAGN  004626          1110    1119    1125#
$DTBL   010306          1576    1611#
$ENDAD  004616           339    1121#   1783
$ENDCT  004564           656    1112#
$ENDMG  004635          1114    1129#
$ENULL  004632          1117    1128#
$ENV    001222           428#   1765    1860    1929    1953
$ENVM   001223           429#    679    1862    1867    1931
$EOP    004530          1091    1102#
$EOPCT  004556           656*   1109#   1113
$ERFLG  001103           376#   1674    1704    1706    1712*   1734    1750*   1788
$ERMAX  001115           382#    659*   1706    1729*   1734
$ERROR  010750           650    1748#
$ERRPC  001116           383#   1366    1368    1370    1372    1757*   1758*   1759    1788    1803
$ERRTB  001264           477#   1811
$ERRTY  011150          1762    1796#
$ERTTL  001112           380#   1756*   1788
$ESCAP  001170           408#    658*   1728*   1779    1781    1788
$ETABL  001222           427#
$ETEND  001264           365     461#
$FATAL  001204           420#   1957*
$FFLG   012032          1920*   1923*   1951    1960*   1968#
$FILLC  001156           401#   1885    1916
$FILLS  001155           400#   1916
$GDADR  001120           384#
$GDDAT  001124           386#    635*    704*    705     707     715*    716     718     726*    727     729     737*    738
                         740     743*    750*    751     753     757*    764*    767     775*    778     787*    788     790
                         799*    803     813*    817     826*    830     839*    843     851*    852     854     862*    867
                         875*    879     889*    940*    941     944     954*    955*    958     969*    972     984*    987
                         993*    996    1006*   1009    1015*   1018    1028*   1031    1037    1047*   1050    1055*   1058
                        1059    1068*   1071    1078    1366    1368    1370    1372
$GET42  004606          1118#
```

```
$GTSWR  007376          1432#   2082
$HD   = 000000           193
$HIBTS  001000           360#
$HIOCT  010464          1657*   1668#
$ICNT   001104           377#   1719*   1720    1722*   1733
$ILLUP  012470          2093    2109    2126#
$INTAG  001135           391#   1460    1549
$ITEMB  001114           381#   1759*   1767    1788    1800
$LF     001200           412#   1534    1543    1669    1788    1916
$LFLG   012031          1961*   1967#
$LPADR  001106           378#    660*   1710*   1726*   1731    1733
$LPERR  001110           379#    661*   1710    1727*   1733    1778
$MADR1  001234           445#
$MADR2  001240           449#
$MADR3  001244           452#
$MADR4  001250           455#
$MAIL   001202           361     365     418#    678     1725    1765    1860
$MAMS1  001232           439#
$MAMS2  001236           447#
$MAMS3  001242           450#
$MAMS4  001246           453#
$MBADR  001002           361#
$MFLG   012030          1921*   1927    1962*   1966#
$MNEW   010071          1435    1547#
$MSGAD  001216           425#   1937*   1940
$MSGLG  001220           426#   1942*
$MSGTY  001202           419#   1935    1943*   1955    1959*
$MSWR   010060          1432    1545#
$MTYP1  001233           440#
$MTYP2  001237           448#
$MTYP3  001243           451#
$MTYP4  001247           454#
$MXCNT  010746          1723    1733#
$NULL   001154           399#   1887    1916
$NWTST= 000001           700#    711#    722#    733#    746#    760#    771#    782#    794#    807#    821#    834#    846#
                         857#    870#    883#    901#    933#    947#    963#    978#   1000#   1023#   1041#   1063#   1086#
                        1133#   1146#
$OCNT   012256          2006*   2035*   2048#
$OMODE  012260          2001*   2005*   2010    2013*   2024*   2050#
$OVER   010732          1687    1703    1711    1721    1730#
$PASS   001210           422#    678*   1106*   1107*   1115    1128    1717    1734
$PASTM  001006           363#
$POWER  012476          2124    2129#
$PWRDN  012330           654    2093#   2121
$PWRMG  012464          2124#
$PWRUP  012402          2103    2109#
$QUES   001176           410#   1478    1527    1543    1666    1669    1788    1916
$ROCHR  007610          1491#   2085
$RODEC= ****** U        2088
$ROLIN  007730          1519#   2086
$RODCT  010326          1630#   2087
$ROSZ   = 000010        1512#
$REGAD  001160           403#
$REGO   001162           405#
$REGI   001164           406#
$RTNAD  004630          1127#
```

```
$R2A    = ****** U       2088
$SAVRE= ****** U         2088
$SAVR6   012474          2102*   2110    2111*   2112*   2128*
$SCOPE   010466           648    1683*
$SETUP= 000137            479*    647     648     650     652     654     656     657     658     660    1104    1415    1549
                         1684    1749    1775    1783
$STUP = 177777            479*
$SVLAD   010676          1695    1724*
$SVPC = 000214            337*    342
$SWR  = 167400            183*    193     311     312     313     314     315     316     317     318     407     408     409
                           657     658     660     661     704     715     726     737     750     764     775     786     798
                           811     825     838     850     861     874     887     905     937     951     967     982    1004
                          1027    1045    1067    1090    1099    1105    1120    1126    1128    1137    1150    1675    1676
                          1677    1678    1679    1686    1698    1700    1701    1704    1705    1706    1713    1714    1715
                          1727    1730    1733    1740    1741    1742    1743    1744    1753    1760    1772    1776    1788
                          2125
$SWREG   001224           430*    681
$SWRMK= 000000            318     319    1679    1680    1702
$TESTN   001206           421*   1725*
$TIMES   001166           407*    657*    811*    861*   1105*   1713*   1720    1723*   1733
$TKB     001146           396*   1413    1424    1441    1495    1501
$TKS     001144           395*   1413    1422    1438   1462*    1493    1499
$TN   = 000035            183*    193     700     704*    708     711     715*    719     722     726*    730     733     737*
                           746     750*    760     764*    768     771     775*    779     782     786*    791     794     798*
                           804     807     811*    818     821     825*    831     834     838*    844     846     850*    855
                           857     861*    868     870     874*    880     883     887*    901     905*    920     928     933
                           937*    945     947     951*    963     967*    978     982*    997    1000    1004*   1019    1023
                          1027*   1038    1041    1045*   1060    1063    1067*   1086    1090*   1133    1137*   1146    1150*
$TPB     001152           398*   1905*   1916
$TPFLG   001157           402*   1854    1916
$TPS     001150           397*   1903    1916
$TRAP    012262           652    2060*
$TRP  = 000012           2068*   2077*   2078*   2079*   2080*   2081*   2082    2083*   2084    2085*   2086*   2087*   2088*
$TRPAD   012304          2065    2075*
$TSTM    001004           362*
$TSTMM   001102           375*   1104*   1674    1702   1724*   1725    1730    1734    1752    1788
$TTYIN   010036          1520    1521    1538   1542*
$TYPBN= ****** U         2081
$TYPDS   010102          1561*   2080
$TYPE    011304          1854*   1948    2068    2076
$TYPEC   011516          1464    1884    1891    1898   1903*   1904
$TYPEX   011564          1909    1911   1914*
$TYPOC   012060          2004*   2077
$TYPON   012074          2003    2006*   2079
$TYPOS   012034          1999*   2078
$UNIT    001214           424*
$UNITM   001010           364*
$USWR    001226           431*
$VECT1   001252           456*    692     693     906
$VECT2   001254           457*
$XTSTR   010502          1689*
$$GET4= 000000           1120*
$DFILL   012257          2000*   2004*   2014   2049*
$40CAT= ****** U         1686    1762
.     = 012506            321*    325*    337     338*    340*    342*    343*    349     350*    352*    354*    372*    413
                          645     660     661    1128    1132    1364*   1408*   1413    1542*   1543    1549    1615*   1669
```

```
COMMEN      1#    303#
ENDCOM      1#    303#
ERROR     197#    636     709     720     731     742     755     769     780     792     805     819     832     845     856
          869     881     896     919     927     946     960     974     989     998    1011    1020    1033    1039    1052
         1061    1073    1080
ESCAPE      1#    303#
GETPRI      1#    303#
GETSWR      1#    303#
MULT        1#    303#
NEWTST      1#    303#    700     711     722     733     746     760     771     782     794     807     821     834     846
          857     870     883     901     933     947     963     978    1000    1023    1041    1063    1086    1133    1146
POP         1#    303#   1602    1658    1963    1964    2114    2115
PUSH        1#    303#   1561    1632    1924    1926    1947    2095    2101
REPORT      1#    303#
SCOPE     198#    703     714     725     736     749     763     774     785     797     810     824     837     849     860
          873     886     904     936     950     966     981    1003    1026    1044    1066    1089    1103    1136    1149
SETPRI      1#    303#
SETTBA   2068#   2077    2078    2079    2080    2082    2084    2085    2086    2087
SETUP       1#    303#    640
SKIP        1#    303#    708     719     730     741     754     768     779     791     804     818     831     844     855
          868     880     897     920     928     945     997    1019    1032    1038    1051    1060    1072    1079    1091
SLASH       1#    303#
SPACE     303#
STARS       1#    303#    335     346     348     355     368     413     416     700     702     711     713     722     724
          733     735     746     748     760     762     771     773     782     784     794     796     807     809     821
          823     834     836     846     848     857     859     870     872     883     885     901     903     933     935
          947     949     963     965     978     980    1000    1002    1023    1025    1041    1043    1063    1065    1086
         1088    1096    1133    1135    1146    1148    1412    1415    1483    1512    1551    1618    1671    1736    1791
         1839    1919    1976    2054    2091    2107
SWRSU       1#    303#    662#
TRNTRP   2068#
TYPBIN      1#    303#
TYPDEC      1#    303#   1115
TYPNAM      1#    303#
TYPNUM      1#    303#
TYPOCS      1#    303#
TYPOCT      1#    303#   1433    1803    1827
TYPTXT      1#    303#
$SCMRE    366#    405     406
$SCMTM    366#
$SESCA      1#    303#
$SNEWT      1#    303#    700     711     722     733     746     760     771     782     794     807     821     834     846
          857     870     883     901     933     947     963     978    1000    1023    1041    1063    1086    1133    1146
$SSET    2068#   2077    2078    2079    2080    2082    2084    2085    2086    2087
$SSETM    678#
$SSKIP      1#    303#    708     719     730     768     779     791     804     818     831     844     855     868     880
          920     928     945     997    1019    1038    1060
.EQUAT      1#    183#    193
.HEADE      1#    183#
.KT11       1#
.SETUP      1#    183#    479
.SWRHI      1#    183#    307
.SWRLO    319#
.$ACT1      1#    183#    333
.$APTB      1#    183#    414#
.$APTH      1#    183#    344
```

DZLKAA.PII CROSS REFERENCE TABLE -- MACRO NAMES

```
.$APTY      1#    183#   1917
.$ASTA      1#
.$CATC      1#    183#    319
.$CMTA      1#    183#    366
.$DB2D      1#
.$DB2O      1#
.$DIV       1#
.$EOP       1#    183#   1094
.$ERRO      1#    183#   1734
.$ERRT      1#    183#   1789
.$MULT      1#
.$PARM    183#
.$POWE      1#    183#   2089
.$RAND      1#
.$RDDE      1#
.$RDOC      1#    183#   1616
.$READ      1#    183#   1410
.$R2AZ      1#
.$SAVE      1#    183#
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#    183#   1669
.$SIZE      1#
.$SPAC    183#
.$SUPR      1#
.$SWDO    183#
.$TRAP      1#    183#   2052
.$TYPB      1#
.$TYPD      1#    183#   1549
.$TYPE      1#    183#   1837
.$TYPO      1#    183#   1974
.$40CA      1#
.1170       1#
```

DZLKAA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
                957   967   969   971   983   984   986   990   991   993   995  1005  1006  1008  1012
               1013  1015  1017  1029  1030  1035  1036  1046  1047  1048  1049  1056  1057  1069  1070
               1075  1076  1111  1115  1118  1137  1143  1150  1157  1159  1161  1433  1457  1462  1491
               1492  1519  1520  1535  1536  1537  1538  1562  1563  1564  1565  1566  1567  1568  1573
               1576  1596  1602  1603  1604  1605  1606  1608  1609  1630  1631  1632  1633  1634  1636
               1637  1656  1657  1658  1659  1660  1691  1692  1694  1697  1710  1722  1723  1726  1727
               1730  1731  1752  1757  1778  1781  1798  1803  1812  1817  1822  1824  1828  1858  1859
               1864  1872  1887  1925  1926  1933  1937  1942  1943  1945  1947  1957  1963  1964  1999
               2007  2008  2009  2015  2022  2040  2041  2042  2043  2044  2060  2061  2065  2093  2094
               2095  2096  2097  2098  2099  2100  2101  2102  2103  2109  2110  2114  2115  2116  2117
               2118  2119  2120  2121  2122
MOVB            659   906  1424  1441  1495  1501  1524  1529  1571  1574  1588  1591  1600  1640  1725
               1729  1759  1767  1869  1897  1905  1920  1921  1923  2000  2001  2004  2005  2006  2010
               2013  2014  2033  2063
NEG            1570  2011
NOP             894   895   915   916   924   925  1122  1123  1124
RESET           639   815   865  1120
ROL             743   757   961   975  1647  1649  1651  2017  2019  2020  2021  2023
RTI             637   673  1463  1511  1539  1610  1661  1732  1787  1874  2045  2125
RTS            1826  1914  1965  2066
SEC             756
SUB             909  1577  1758  1940
TRAP           2068  2077  2078  2079  2080  2082  2084  2085  2086  2087
TST             683  1141  1155  1455  1470  1582  1592  1655  1662  1693  1717  1772  1779  1830  1871
               1879  1901  1935  1953  1955  2028  2062
TSTB           1422  1438  1493  1499  1584  1598  1704  1854  1903  1927  1938  1951
.ASCII          410   411
.ASCIZ          409   412  1129  1182  1189  1194  1199  1205  1211  1217  1222  1230  1236  1243  1251
               1255  1260  1264  1274  1284  1295  1306  1312  1318  1323  1328  1334  1340  1346  1352
               1358  1376  1387  1398  1543  1544  1545  1547  1834  2129
.BLKB          1542
.BLKW          1615
.BYTE           375   376   381   382   390   391   399   400   401   402   428   429   439   440   447
                448   450   451   453   454  1128  1540  1541  1769  1770  1966  1967  1968  2046  2047
               2048  2049
.DSABL         1480
.ENABL            1   183  1413
.END           2132
.ENDC           188   197   289   303   315   317   318   319   330   336   340   342   347   349   356
                369   373   375   403   407   408   409   410   414   417   439   447   450   453   456
                457   458   459   460   463   479   646   647   650   652   654   656   657   658   660
                662   683   701   702   703   704   709   712   713   714   715   720   723   724   725
                726   731   734   735   736   737   742   747   748   749   750   755   761   762   763
                764   769   772   773   774   775   780   783   784   785   786   792   795   796   797
                798   805   808   809   810   811   812   819   822   823   824   825   832   835   836
                837   838   845   847   848   849   850   856   858   859   860   861   862   869   871
                872   873   874   881   884   885   886   887   898   902   903   904   905   921   929
                934   935   936   937   946   948   949   950   951   964   965   966   967   979   980
                981   982   998  1001  1002  1003  1004  1020  1024  1025  1026  1027  1033  1039  1042
               1043  1044  1045  1052  1061  1064  1065  1066  1067  1073  1080  1087  1088  1089  1090
               1092  1097  1098  1099  1101  1104  1110  1113  1114  1118  1120  1126  1128  1129  1132
               1134  1135  1136  1137  1147  1148  1149  1150  1413  1414  1416  1444  1480  1484  1512
               1513  1520  1522  1525  1527  1543  1549  1552  1619  1625  1669  1672  1675  1680  1686
               1688  1699  1702  1703  1704  1706  1708  1715  1719  1724  1726  1730  1733  1734  1737
               1740  1750  1757  1762  1763  1764  1772  1783  1787  1788  1792  1807  1836  1840  1869
               1920  1921  1924  1951  1966  1977  2055  2061  2064  2076  2077  2078  2079  2080  2081
```

```
            2082    2083    2084    2085    2086    2087    2088    2092    2101    2102    2108    2114    2115    2125    2132
.EQUIV       197     198     206     221     222     251     252     253     254     255     256     257     258     259     260
             279     280     281     282     283     284     285     286     287     288
.EVEN        417    1364    1408    1835    1969    2131
.IF          184     195     261     289     314     316     317     318     319     328     335     338     340     346     348
             355     368     372     374     403     407     408     409     413     414     416     439     447     450     453
             456     457     458     459     460     461     463     479     641     646     648     650     652     654     656
             657     658     660     678     700     702     704     708     711     713     715     719     722     724     726
             730     733     735     737     741     746     748     750     754     760     762     764     768     771     773
             775     779     782     784     786     791     794     796     798     804     807     809     811     812     818
             821     823     825     831     834     836     838     844     846     848     850     855     857     859     861
             862     868     870     872     874     880     883     885     887     897     901     903     905     920     928
             933     935     937     945     947     949     951     963     965     967     978     980     982     997    1000
            1002    1004    1019    1023    1025    1027    1032    1038    1041    1043    1045    1051    1060    1063    1065
            1067    1072    1079    1086    1088    1090    1091    1096    1097    1098    1099    1100    1101    1103    1109
            1112    1114    1118    1120    1126    1128    1129    1133    1135    1137    1146    1148    1150    1412    1414
            1415    1416    1444    1483    1484    1512    1520    1521    1525    1526    1542    1543    1549    1551    1618
            1621    1637    1671    1674    1679    1685    1686    1698    1700    1701    1702    1704    1705    1706    1715
            1717    1725    1727    1732    1733    1734    1736    1739    1750    1753    1760    1762    1763    1765    1772
            1776    1783    1787    1788    1791    1806    1822    1839    1860    1919    1921    1924    1951    1966    1976
            2054    2060    2064    2068    2077    2078    2079    2080    2081    2082    2084    2085    2086    2087    2088
            2091    2101    2102    2107    2114    2115    2123    2125    2129
.IFF          195     314     317     318     319     336     340     342     347     349     356     369     372     375     403
             414     417     646     701     702     703     704     709     712     713     714     715     720     723     724
             725     726     731     734     735     736     737     741     747     748     749     750     754     761     762
             763     764     769     772     773     774     775     780     783     784     785     786     792     795     796
             797     798     805     808     809     810     811     819     822     823     824     825     832     835     836
             837     838     845     847     848     849     850     856     858     859     860     861     869     871     872
             873     874     881     884     885     886     887     897     902     903     904     905     921     929     934
             935     936     937     946     948     949     950     951     964     965     966     967     979     980     981
             982     998    1001    1002    1003    1004    1020    1024    1025    1026    1027    1032    1039    1042    1043
            1044    1045    1051    1061    1064    1065    1066    1067    1072    1079    1087    1088    1089    1090    1091
            1097    1100    1104    1110    1113    1128    1134    1135    1136    1137    1147    1148    1149    1150    1413
            1416    1484    1486    1491    1512    1513    1522    1526    1543    1552    1619    1672    1699    1702    1703
            1706    1733    1734    1737    1739    1753    1783    1788    1792    1807    1836    1840    1920    1977    2055
            2061    2092    2108    2125
.IFT        1486    1491    1642    1662    1669    1714    1763
.IFTF       1431    1484    1487    1638    1646    1668    1712    1762
.IIF         183     188     193     311     312     313     315     318     319     325     413     417     647     650     656
             657     658     660     661    1098    1104    1105    1116    1128    1132    1413    1434    1535    1543    1549
            1669    1675    1676    1677    1678    1679    1680    1684    1713    1714    1730    1733    1734    1740    1741
            1742    1743    1744    1749    1775    1783    1788    1804    1829    1916    2076    2077    2078    2079    2080
            2082    2084    2085    2086    2087
.IRP         479     700     711     722     733     746     760     771     782     794     807     821     834     846     857
             870     883     901     933     947     963     978    1000    1023    1041    1063    1086    1133    1146    1562
            1602    1632    1658    1685    1925    1926    1947    1963    1964    2095    2101    2114    2115
.LIST           1     183     303     318     325     403     405     406     407     414     417     479     662     700     704
             711     715     722     726     733     737     746     750     760     764     771     775     782     786     794
             798     807     811     821     825     834     838     846     850     857     861     870     874     883     887
             901     905     933     937     947     951     963     967     978     982    1000    1004    1023    1027    1041
            1045    1063    1067    1086    1090    1104    1120    1133    1137    1146    1150    1512    1679    1783    2068
            2076    2077    2078    2079    2080    2081    2082    2083    2084    2085    2086    2087    2088
.MACRO          1     319     366     678    2068
.MCALL        183     303     414     662
.MEXIT        462
.NLIST          1     183     303     318     325     403     405     406     407     414     417     479     662     700     704
```

```
            711     715     722     726     733     737     746     750     760     764     771     775     782     786     794
            798     807     811     821     825     834     838     846     850     857     861     870     874     883     887
            901     905     933     937     947     951     963     967     978     982    1000    1004    1023    1027    1041
           1045    1063    1067    1086    1090    1104    1120    1133    1137    1146    1150    1512    1679    1783    2068
           2076    2077    2078    2079    2080    2081    2082    2083    2084    2085    2086    2087    2088
.PAGE       366     463
.REM          1
.REPT       325     405
.SBTTL      193     307     319     328     333     344     366     414     463     640     700     711     722     733     746
            760     771     782     794     807     821     834     846     857     870     883     901     933     947     963
            978    1000    1023    1041    1063    1086    1094    1133    1146    1181    1410    1549    1616    1669    1734
           1789    1837    1917    1974    2052    2068    2089
.TITLE      183
.WORD       325     326     327     341     360     361     362     363     364     365     374     377     378     379     380
            383     384     385     386     387     388     389     392     393     394     403     405     406     419     420
            421     422     423     424     425     426     430     431     432     445     449     452     455     456     457
            458     459     460    1109    1112    1127    1665    1668    1815    1820    1866    1913    1949    2050    2124


ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*,DZLKAA.SEQ/SOL/CRF/PAGNUM/NL:TOC=DZLKAA.SML,DZLKAA.P11
RUN-TIME: 36 44 5 SECONDS
RUN-TIME RATIO: 156/86=1.8
CORE USED:  33K  (65 PAGES)
```

Spooler runtime 9 Seconds, 43 KCS, 247 disk reads, 3 disk writes, 62 pages