LPS11/DRA

DIAGNOSTIC I/O TEST

MD-11-DZLPI-B

EP-DCFPB-B-DL-A

OCT 1976

FICHE 1 OF 1

digital

Made In U.S.A.

.REM %

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZLPI-B
PRODUCT NAME:	LPS-11 DRA DIGITAL I/O TEST
DATE:	MAY 21, 1976
MAINTAINER:	DIAGNOSTIC GROUP

1. ABSTRACT

THIS PROGRAM IS A LOGIC TEST OF THE "LPS-11-DRA" DIGITAL INPUT OUTPUT CONTROL OPTION. ALL FUNCTIONS OF THE OPTION WILL BE TESTED. DUE TO THE FLEXIBILITY OF THE OPTION, THE OPERATOR MAY BE REQUIRED TO SUPPLY OPTION CHARASTICS.
THE PROGRAM WILL HANDLE ALL CONFIGURATIONS OF SW15 THRU SW00 AND JUMPERS W15 THRU W00. THE FOLLOWING JUMPERS MUST BE INSERTED TO EXECUTE THE LOGIC TEST: W16-W17-W18-W19.

2. REQUIREMENTS

2.1 EQUIPTMENT

PDP-11 FAMILY COMPUTER WITH 4K WORDS OF MEMORY
LPS-11 WITH DRA OPTION INSTALLED
TELETYPE

2.2 STORAGE

THIS PROGRAM USES 4K OF MEMORY.

3. LOADING PROCEDURE

PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

STANDARD PDP-11 FORMAT

```
SW 15 = 1       HALT ON ERROR
SW 14 = 1       LOOP ON TEST
SW 13 = 1       INHIBIT ERROR TYPOUTS
SW 11 = 1       INHIBIT INTERATIONS
SW 10 = 1       DRA INPUT-OUTPUT CABLE NOT CONNECTED
SW 09 = 1       LOOP ON ERROR
SW 08 = 1       LOOP ON TEST IN SWR <7:0>
```

REFER TO 10. FOR SOFTWARE SWITCH REGISTER OPERATION.

4.2 STARTING ADDRESS OR ADDRESSES

```
200     IS THE STARTING ADDRESS OF THE LOGIC TEST.
204     IS THE RESTART ADDRESS OF THE LOGIC TEST.
210     IS THE STARTING ADDRESS OF THE COMBINED FUNCTION LOOP.
```

5. OPERATING PROCEDURE

THE FOLLOWING JUMPERS MUST BE INSTALLED TO EXECUTE THE
LOGIC TEST: W16-W17-W18-W19.
IF THE CUSTOMER HAS SELECTED THE "A" SECTION OF THESE JUMPER
IT MUST BE RETURNED TO THE "FACTORY" POSITION BEFORE RUNNING
THE LOGIC TEST. ** WORSE CASE WILL ONLY BE CHANGING FOUR JUMPERS. **
THE OPERATOR MUST INSERT THE CORRECT INFORMATION IN THE SWITCH
REGISTER WHEN REQUESTED BY THE PROGRAM OR AN ERROR WILL OCCUR.
ONCE STARTED THE TEST WILL RUN IN IT'S NORMAL MANNER
WITHOUT OPERATOR INTERVENTION OR SWITCH SELECTION.

6. ERRORS

THIS PROGRAM USES THE DIAGNOSTIC 'SYSMAC' PACKAGE FOR
ERROR REPOTRING AND TYPEOUT. REFER TO THE "ERROR POINTER TABLE"
FOR TYPE AND DESCRIPTION OF ERRORS.

7. RESTRICTIONS

THE FOLLOWING JUMPERS MUST BE IN THE "FACTORY" POSITION:
W16-W17-W18-W19.
THE OPERATOR MUST SUPPLY THE CORRECT JUMPER AND SWITCH
CONFIGURATION OR AN ERROR WILL OCCUR.

8. MISCELANEOUS

8.1 EXECUTION TIME

THE LOGIC TEST WILL TAKE APPROXIMATELY 60 SECONDS FOR COMPLETION
AND WILL TYPE 'END PASS'.
THE COMBINED FUNCTION LOOP WILL NEVER EXIT.

8.2 DEVICE ADDRESS PROGRAM LOCATIONS

LOCATION "$BASE" CONTAINS THE DRA BASE DEVICE ADDRESS <170410>
LOCATION "$VECT1" CONTAINS THE DRA BASE INTERRUPT VECTOR <350>
LOCATION "$PRIOR" CONTAINS THE DRA BR LEVEL <200><4>

*NOTE: IF THESE LOCATIONS ARE CHANGED, THE OPERATOR MUST START
THE TEST AGAIN AT LOC. 200. THE PROGRAM WILL USE THE BASE
ADDRESS AND VECTOR AND UPDATE THE ACTUAL PROGRAM VALUES.

8.3 ACT/XXDP/APT

THE PROGRAM IS CHAINABLE UNDER XXDP AND ACT. THE HOOKS FOR "APT"
ARE PROVIDED BUT HAVE NOT BEEN TESTED.

9. PROGRAM DESCRIPTION

THE LOGIC TEST IS A TEST OF THE CONTROL AND INPUT/OUTPUT
REGISTERS. ALL JUMPERS AND SWITCHES COMBINATIONS EXCEPT:

W16A-W17A-W19A-W19A CAN BE DIAGNOSED.

THE PROGRAM CHECKS THAT THE DRA CAN INTERRUPT AND THAT
"RESET" WILL WORK CORRECTLY.

THE COMBINED FUNCTION LOOP PROVIDES THE OPERATOR WITH:

1. SCOPE LOOP FOR INVERTED SIGNALS (W16A-W17A-W18A-W19A).
2. SLOW LOOP FOR TESTING RELAY CLOSURE.

10. SOFTWARE SWITCH REGISTER OPERATION

THE PROGRAM SUPPORTS NON-SWITCH REGISTER CPU TYPES.
A CHANGE IN SWR VALUE IS ACCOMPLISHED BY TYPING A "CTRL G".
THE RESPONSE WILL BE "SWR = " AND WAIT FOR A NEW VALUE.
THE OPERATOR NOW INPUTS THE NEW VALUE AND TERMINATES WITH A "CR".

11. TABLE OF CONTENTS

ATTACHED.

```
                                        %
                                .TITLE  MAINDEC-11-DZLPI-B      LPS-11-DRA DIAGNOSTIC
                                ;*COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY RAYMOND SHOOP
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-B2),NOV 21, 1975.
                                ;*

                                .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
```

```
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        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)
                        BIT15=  100000
                        BIT14=  40000
                        BIT13=  20000
                        BIT12=  10000
                        BIT11=  4000
                        BIT10=  2000
                        BIT09=  1000
                        BIT08=  400
                        BIT07=  200
                        BIT06=  100
                        BIT05=  40
                        BIT04=  20
                        BIT03=  10
                        BIT02=  4
                        BIT01=  2
                        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
```

H01


```
                        TBITVEC=14              ;;"T" BIT
                        TRTVEC= 14              ;;TRACE TRAP
                        BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                        IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                        PWRVEC= 24              ;;POWER FAIL
                        EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        TRAPVEC=34              ;;"TRAP" TRAP
                        TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        TPVEC=  64              ;;TTY PRINTER VECTOR
                        PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                                ABASE=170410
                                AVECT1=350
                                APRIOR=200
```

```
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        11            INHIBIT ITERATIONS
                                ;*        10            DRA INPUT OUTPUT CABLE NOT CONNECTED
                                ;*         9            LOOP ON ERROR
                                ;*         8            LOOP ON TEST IN SWR<7:0>


                                .SBTTL  TRAP CATCHER

            000000                      .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
            000174              .=174
000174  000000                 DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                 SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER


                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  001470                  JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  001474                  JMP     @#BEGIN1        ;JUMP TO THE RESTART ADDRESS
000210  000137  010040                  JMP     @#EXTTST        ;JUMP TO THE MISC. TEST
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
000214                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  010240                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000214                                  .=$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  001174                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000012                  $TSTM:  .WORD   10.     ;;RUN TIM OF LONGEST TEST
001006  000074                  $PASTM: .WORD   60.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000074                  $UNITM: .WORD   60.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000052                          .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  000000                          .WORD   0
001136  177570                  SWR:    .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
001140  177570                  DISPLAY: .WORD  DDISP   ;;ADDRESS OF DISPLAY REGISTER
001142  177560                  $TKS:   177560          ;;TTY KBD STATUS
001144  177562                  $TKB:   177562          ;;TTY KBD BUFFER
001146  177564                  $TPS:   177564          ;;TTY PRINTER STATUS REG. ADDRESS
001150  177566                  $TPB:   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
001152  000                     $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
001153  002                     $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001154  012                     $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001155  000                     $TPFLG: .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001156  000000                  $REGAD: .WORD   0       ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REGO) WAS OBTAINED
001160  000000                  $REGO:  .WORD   0       ;;CONTAINS (($REGAD)+0)
001162  000000                  $REG1:  .WORD   0       ;;CONTAINS (($REGAD)+2)
001164  000000                  $TIMES: 0               ;;MAX. NUMBER OF ITERATIONS
001166  000000                  $ESCAPE:0               ;;ESCAPE ON ERROR ADDRESS
001170  077                     $QUES:  .ASCII  /?/     ;;QUESTION MARK
001171  015                     $CRLF:  .ASCII  <15>    ;;CARRIAGE RETURN
001172  000012                  $LF:    .ASCIZ  <12>    ;;LINE FEED
```

```
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE
                                ;;*****************************************************************
                                .EVEN
001174                          $MAIL:                          ;;APT MAILBOX
001174  000000                  $MSGTY: .WORD   AMSGTY   ;;MESSAGE TYPE CODE
001176  000000                  $FATAL: .WORD   AFATAL   ;;FATAL ERROR NUMBER
001200  000000                  $TESTN: .WORD   ATESTN   ;;TEST NUMBER
001202  000000                  $PASS:  .WORD   APASS    ;;PASS COUNT
001204  000000                  $DEVCT: .WORD   ADEVCT   ;;DEVICE COUNT
001206  000000                  $UNIT:  .WORD   AUNIT    ;;I/O UNIT NUMBER
001210  000000                  $MSGAD: .WORD   AMSGAD   ;;MESSAGE ADDRESS
001212  000000                  $MSGLG: .WORD   AMSGLG   ;;MESSAGE LENGTH
001214                          $ETABLE:                        ;;APT ENVIRONMENT TABLE
001214  000                     $ENV:   .BYTE   AENV     ;;ENVIRONMENT BYTE
001215  000                     $ENVM:  .BYTE   AENVM    ;;ENVIRONMENT MODE BITS
001216  000000                  $SWREG: .WORD   ASWREG   ;;APT SWITCH REGISTER
001220  000000                  $USWR:  .WORD   AUSWR    ;;USER SWITCHES
001222  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
001224  000                     $MAMS1: .BYTE   AMAMS1   ;;HIGH ADDRESS,M.S. BYTE
001225  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
001226  000000                  $MADR1: .WORD   AMADR1   ;;HIGH ADDRESS,BLK#1
                                ;*                                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001230  000                     $MAMS2: .BYTE   AMAMS2   ;;HIGH ADDRESS,M.S. BYTE
001231  000                     $MTYP2: .BYTE   AMTYP2   ;;MEM.TYPE,BLK#2
001232  000000                  $MADR2: .WORD   AMADR2   ;;MEM.LAST ADDRESS,BLK#2
001234  000                     $MAMS3: .BYTE   AMAMS3   ;;HIGH ADDRESS,M.S.BYTE
001235  000                     $MTYP3: .BYTE   AMTYP3   ;;MEM.TYPE,BLK#3
001236  000000                  $MADR3: .WORD   AMADR3   ;;MEM.LAST ADDRESS,BLK#3
001240  000                     $MAMS4: .BYTE   AMAMS4   ;;HIGH ADDRESS,M.S.BYTE
001241  000                     $MTYP4: .BYTE   AMTYP4   ;;MEM.TYPE,BLK#4
001242  000000                  $MADR4: .WORD   AMADR4   ;;MEM.LAST ADDRESS,BLK#4
001244  350                     $VECT1: .BYTE   AVECT1   ;;INTERRUPT VECTOR#1
001245  000                     $VECT2: .BYTE   AVECT2   ;;INTERRUPT VECTOR#2
001246  200                     $PRIOR: .BYTE   APRIOR   ;;BUS PRIORITY #1,#2
001247  000                             .BYTE   0        ;;SPARE,NOT USED
                                        .EVEN
001250  170410                  $BASE:  .WORD   ABASE    ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001252  000000                  $DEVM:  .WORD   ADEVM    ;;DEVICE MAP
001254  000000                  $CDW1:  .WORD   ACDW1    ;;CONTROLLER DESCRIPTION WORD#1
001256  000000                  $CDW2:  .WORD   ACDW2    ;;CONTROLLER DESCRIPTION WORD#2
001260  000000                  $DDW0:  .WORD   ADDW0    ;;DEVICE DESCRIPTOR WORD#0
001262  000000                  $DDW1:  .WORD   ADDW1    ;;DEVICE DESCRIPTOR WORD#1
001264  000000                  $DDW2:  .WORD   ADDW2    ;;DEVICE DESCRIPTOR WORD#2
```

```
001266  000000          $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
001270  000000          $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
001272  000000          $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
001274  000000          $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
001276  000000          $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
001300  000000          $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
001302  000000          $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
001304  000000          $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
001306  000000          $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
001310  000000          $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
001312  000000          $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
001314  000000          $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
001316  000000          $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15

001320                  $ETEND:


                        .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


001320                  $ERRTB:

                        ;ITEM   1
001320  010312                  EM1             ;STATUS REGISTER IN ERROR
001322  010732                  DH1             ;ERRPC  STATUS EXPECTED
001324  011116                  DT1             ;$ERRPC $BDDAT $GDDAT
001326  011144                  DF1

                        ;ITEM   2
001330  010343                  EM2             ;INPUT REGISTER IN ERROR
001332  010763                  DH2             ;ERRPC INPUT EXPECTED
001334  011116                  DT1             ;$ERRPC $BDDAT $GDDAT
001336  011144                  DF1

                        ;ITEM 3
001340  010373                  EM3             ;OUTPUT REGISTER IN ERROR
001342  011014                  DH3             ;ERRPC OUTPUT EXPECTED
001344  011116                  DT1             ;$ERRPC $BDDAT $GDDAT
001346  011144                  DF1

                        ;ITEM   4
001350  010424                  EM4             ;INPUT FAILED TO INTERRUPT
001352  011045                  DH4             ;ERRPC
001354  011126                  DT4             ;$ERRPC
```

```
   522  001356  011144                          DF1
   523
   524                                :ITEM   5
   525  001360  010456                          EM5             ;OUTPUT FAILED TO INTERRUPT
   526  001262  011045                          DH4             ;ERRPC
   527  001364  011126                          DT4             ;$ERRPC
   528  001366  011144                          DF1
   529
   530                                :ITEM   6
   531  001370  010511                          EM6             ;UNEXPECTED INTERRUPT
   532  001372  011045                          DH4             ;ERRPC
   533  001374  011126                          DT4             ;$ERRPC
   534  001376  011144                          DF1
   535
```

```
                                        ;ITEM   7
001420  010536                          EM7             ;OPERATOR INTERVENTION ERROR
001422  011045                          DH4             ;ERRPC
001424  011126                          DT4             ;$ERRPC
001426  011144                          DF1

                                        ;ITEM   10
001430  010572                          EM10            ;INTERRUPT INPUT BIT FAILED TO SET INPUT READY
001432  011053                          DH10            ;ERRPC    STATUS   EXPECTED  INPUT BIT
001434  011132                          DT10            ;$ERRPC  $BDDAT  $GDDAT  BRLEV3
001436  011150                          DF10

                                        ;ITEM   11
001420  010655                          EM11            ;NON-INTERRUPTING INPUT BIT SET INPUT READY
001422  011053                          DH10            ;ERRPC  STATUS  EXPECTED  INPUT BIT
001424  011132                          DT10            ;$ERRPC  $BDDAT  $GDDAT  BRLEV3

001426  170410                  DRADD:  170410                  ;DRA STARTING ADDRESS
001430  000350                  DRIV:   350                     ;DRA STARTING INTERRUPT VECTOR
001432  000200                  DRBRL:  200                     ;DRA BR LEVEL

001434  170410                  GRSTAT: 170410          ;DR STATUS
001436  170412                  GRDAI:  170412          ;INPUT REG.
001440  170414                  GRDIO:  170414          ;OUTPUT REG.
001442  170415                  GRBHIO: 170415          ;OUTPUT REG HIGH BYTE
001444  000000                  NOTLCH: 0
001446  000000                  NOTINT: 0
001450  000310                  GRIVA:  310
001452  000312                  GRIVSA: 312
001454  000314                  GRIVB:  314
001456  000316                  GRIVSB: 316
001460  000200                  DIOBRL: 200
001462  000000                  BRLEV1: 0
000004  000000                  BRLEV2: 0
                                BRLEV3: 0
```

```
                                        ;;*****************************************************************
                                        ;DIGITAL I-O LOGIC TEST
                                        ;;*****************************************************************
001470  005000                BEGIN:  CLR     R0
001472  000402                        BR      RBEG                    ;
001474  013700  177777        BEGIN1: MOV     #-1,R0
001500  000005                RBEG:   RESET
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001502  012706  001100                MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001506  005026                        CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001510  022706  001126                CMP     #$BDDAT,R6      ;;DONE?
001514  001374                        BNE     .-6             ;;LOOP BACK IF NO
001516  012706  001100                MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
001522  012737  011246  000020        MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001530  012737  000340  000022        MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001536  012737  011526  000030        MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001544  012737  000340  000032        MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001552  012737  014034  000034        MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001560  012737  000340  000036        MOV     #340,@#TRAPVEC+2;LEVEL 7
001566  012737  013064  000024        MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001574  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001602  013737  010220  010212        MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
001610  005037  001164                CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
001614  005037  001166                CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001620  112737  000001  001115        MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
001626  012737  001626  001106        MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
001634  012737  001634  001110        MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001642  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001646  012737  001704  000004        MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001654  012737  177570  001136        MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001662  012737  177570  001140        MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001670  022777  177777  177240        CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001676  001013                        BNE     65$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
001700  005737  000001                TST     @#1             ;;FORCE A TRAP THROUGH ERRVEC
001704  012737  000176  001136  64$:  MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001712  012737  000174  001140        MOV     #DISPREG,DISPLAY ;;POINT TO SOFTWARE DISPLAY REG
001720  012716  001726                MOV     #65$,(SP)       ;;REPLACE OLD PC WITH NEW
001724  000002                        RTI                     ;;RESTORE PC AND PSW
001726  012637  000004        65$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001732                          $ARG1:
001732  005037  001202                CLR     $PASS           ;;CLEAR PASS COUNT
001736  132737  000200  001215        BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
001744  001403                        BEQ     64$             ;;YES,USE NON-APT SWITCH
001746  012737  001216  001136        MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
001754                          64$:
```

```
001754  013737  001250  001426  DCNOT:  MOV     $BASE,DRADD
001762  013737  001244  001430          MOV     $VECT1,DRIV
001770  013737  001246  001432          MOV     $PRIOR,DRBRL
001776  005700                          TST     R0                      ;TEST R0
002000  001402                          BEQ     2$                      ;BR IF CLEARED
002002  000137  002470          1$:     JMP     3$                      ;JUMP IF SET
002006  005737  000042          2$:     TST     @#42                    ;TEST IF IN "CHAIN MODE"
002012  001373                          BNE     1$                      ;BR IF YES
002014  104400  002022                  TYPE    ,65$                  ;;TYPE ASCIZ STRING
002020  000430                          BR      64$                   ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12><15><12>/LPS-11-DRA DIGITAL INPUT OUTPUT LOGIC TEST/
002122                          64$:
002122  104400  002110                  TYPE    ,67$                  ;;TYPE ASCIZ STRING
002106  000413                          BR      66$                   ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <15><12>/MAINDEC-11-DZLPI-B/
002136                          66$:
002136  104400  002144                  TYPE    ,69$                  ;;TYPE ASCIZ STRING
002142  000435                          BR      68$                   ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <15><12>/SET SWITCH REGISTER BITS EQUAL TO THE NON-LATCHING BITS/
002236                          68$:
002236  004737  011170                  JSR     PC,GETSWR               ;GET THE SWITCH REGISTER VALUE
002242  017737  176670  001444          MOV     @SWR,NOTLCH             ;SAVE SWITCHES
002250  104400  002256                  TYPE    ,71$                  ;;TYPE ASCIZ STRING
002254  000437                          BR      70$                   ;;GET OVER THE ASCIZ
                                ;;71$:  .ASCIZ  <15><12>/SET SWITCH REGISTER BITS EQUAL TO THE NON-INTERRUPTING BITS
002354                          70$:
```

```
 
002354  004737  011170                    JSR     PC,GETSWR                ;GET THE SWITCH REGISTER VALUE
002360  017737  176552  001446            MOV     @SWR,NOTINT              ;SAVE SWITCHES
002366  104400  002374                    TYPE    ,73$             ;;TYPE ASCIZ STRING
002372  000434                            BR      72$              ;;GET OVER THE ASCIZ
                                  ;;73$:  .ASCIZ  <15><12>/SET SWITCH REGISTER WITH THE DESIRED PROGRAM OPTIONS/
002464                            72$:
002464  004737  011170                    JSR     PC,GETSWR                ;GET THE SWITCH REGISTER VALUE
002470  013737  001426  001434    3$:     MOV     DRADD,GRSTAT             ;LOAD INITIAL ADDRESS
002476  013737  001426  001436            MOV     DRADD,GRDAI              ;LOAD 2ND ADDRESS
002504  062737  000002  001436            ADD     #2,GRDAI
002512  013737  001426  001440            MOV     DRADD,GRDIO              ;LOAD 3RD ADDRESS
002520  062737  000004  001440            ADD     #4,GRDIO
002526  013737  001426  001442            MOV     DRADD,GRBHIO             ;LOAD 4TH ADDRESS
002534  062737  000005  001442            ADD     #5,GRBHIO
002542  013737  001430  001450            MOV     DRIV,GRIVA               ;LOAD FIRST VECTOR
002550  013737  001430  001452            MOV     DRIV,GRIVSA
002556  062737  000002  001452            ADD     #2,GRIVSA
002564  013737  001430  001454            MOV     DRIV,GRIVB               ;LOAD 2ND VECTOR
002572  062737  000004  001454            ADD     #4,GRIVB
002600  013737  001430  001456            MOV     DRIV,GRIVSB
002606  062737  000006  001456            ADD     #6,GRIVSB
002614  013737  001432  001460            MOV     DRBRL,DIOBRL             ;LOAD BR LEVEL
002622  000005                    IOTEST: RESET
002624  012706  001100                    MOV     #STACK,SP                ;LOAD STACK
002630  013777  001452  176612            MOV     GRIVSA,@GRIVA            ;RESET INPUT VECTOR
002636  005077  176610                    CLR     @GRIVSA
002642  013777  001456  176604            MOV     GRIVSB,@GRIVB            ;RESET OUTPUT VECTOR
002650  005077  176602                    CLR     @GRIVSB
                                  ;;************************************************************
                                  ;;*TEST 1         TEST FOR NO BUS ERRORS
                                  ;;************************************************************
002654  000004                    TST1:   SCOPE
002656  005077  176552                    CLR     @GRSTAT
002662  005077  176550                    CLR     @GRDAI
002666  005077  176546                    CLR     @GRDIO

                                  ;;************************************************************
                                  ;;*TEST 2         TEST THAT OUTPUT REG. CAN HOLD #-1
                                  ;;************************************************************
002672  000004                    TST2:   SCOPE
002674  012737  177777  001124            MOV     #-1,$GDDAT               ;LOAD EXPECTED
002702  012777  177777  176530            MOV     #-1,@GRDIO               ;ALL ONES TO REGISTER
002710  017737  176524  001126            MOV     @GRDIO,$BDDAT            ;READ OUTPUT REG.
002716  023737  001124  001126            CMP     $GDDAT,$BDDAT            ;COMPARE
002724  001401                            BEQ     TST3             ;;BR IF EQUAL
002726  104003                            ERROR   3                        ;REG WILL NOT HOLD ONES
```

```
                                        ;;*****************************************************************
                                        ;*TEST 3         TEST THAT RESET CLEARS OUTPUT REG.
                                        ;;*****************************************************************
002730  000004                  TST3:   SCOPE
002732  012737  000040  001164          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
002740  005037  001124                  CLR     $GDDAT
002744  012777  177777  176466          MOV     #-1,@GRDIO
002752  000005                          RESET                           ;SET DATA TO ALL ONES
002754  017737  176460  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
002762  001401                          BEQ     TST4            ;;BR IF EQUAL
002764  104003                          ERROR   3                       ;REG FAILED TO CLEAR

                                        ;;*****************************************************************
                                        ;*TEST 4         TEST THAT OUTPUT REG. CAN HOLD #52525
                                        ;;*****************************************************************
002766  000004                  TST4:   SCOPE
002770  012737  052525  001124          MOV     #52525,$GDDAT           ;LOAD EXPECTED VALUE
002776  012777  052525  176434          MOV     #52525,@GRDIO
003004  017737  176430  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
003012  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003020  001401                          BEQ     TST5            ;;BR IF EQUAL
003022  104003                          ERROR   3                       ;DATA NOT=52525

                                        ;;*****************************************************************
                                        ;*TEST 5         TEST THAT OUTPUT REG. CAN HOLD #125252
                                        ;;*****************************************************************
003024  000004                  TST5:   SCOPE
003026  012737  125252  001124          MOV     #125252,$GDDAT          ;LOAD EXPECTED VALUE
003034  012777  125252  176376          MOV     #125252,@GRDIO
003042  017737  176372  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
003050  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003056  001401                          BEQ     TST6            ;;BR IF EQUAL
003060  104003                          ERROR   3                       ;DATA NOT=125252

                                        ;;*****************************************************************
                                        ;*TEST 6         TEST THAT OUTPUT REG. CAN HOLD A COUNT PATTERN
                                        ;;*****************************************************************
003062  000004                  TST6:   SCOPE
003064  012737  000004  001164          MOV     #4,$TIMES       ;;DO 4 ITERATIONS
003072  012737  003104  001110          MOV     #2$,$LPERR              ;LOAD SCOPE ERROR RETURN
003100  005037  001124                  CLR     $GDDAT                  ;CLEAR PATTERN
003104  013777  001124  176326  2$:     MOV     $GDDAT,@GRDIO           ;LOAD THE OUTPUT REG.
003112  017737  176322  001126          MOV     @GRDIO,$BDDAT           ;READ THE REG.
003120  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003126  001401                          BEQ     1$              ;;BR IF EQUAL
003130  104003                          ERROR   3                       ;OUTPUT REG.FAILED TO HOLD COUNT PATTERN
003132  005237  001124          1$:     INC     $GDDAT                  ;UPDATE THE PATTERN
003136  001362                          BNE     2$                      ;TRY AGAIN
```

```
                                ;;*****************************************************************
                                ;*TEST 7         FLOAT A 1 ACROSS THE OUTPUT REGISTER
                                ;;*****************************************************************
003140  000004                  TST7:   SCOPE
003142  012737  003156  001110          MOV     #1$,$LPERR              ;LOAD SCOPE ERROR RETURN
003150  012737  000001  001124          MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE

003156  005077  176256          1$:     CLR     @GRDIO                  ;CLEAR OUTPUT
003162  053777  001124  176250          BIS     $GDDAT,@GRDIO           ;SET THAT BIT
003170  017737  176244  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
003176  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST RESULTS
003204  001401                        BEQ     2$                      ;BR IF EQUAL ?
003206  104003                        ERROR   3

003210  006337  001124          2$:     ASL     $GDDAT                  ;SHIFT EXPECTED DATA
003214  001360                        BNE     1$                      ;BR UNTIL DONE
                                ;;*****************************************************************
                                ;*TEST 10        FLOAT A 0 ACROSS THE OUTPUT REGISTER
                                ;;*****************************************************************
003216  000004                  TST10:  SCOPE
003220  012737  003234  001110          MOV     #1$,$LPERR              ;LOAD SCOPE ERROR RETURN
003226  012737  000001  001124          MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE

003234  012777  177777  176176  1$:     MOV     #-1,@GRDIO              ;LOAD OUTPUT TO A ONE
003242  043777  001124  176170          BIC     $GDDAT,@GRDIO           ;CLEAR A BIT
003250  017737  176164  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
003256  005137  001126                  COM     $BDDAT                  ;COMPLEMENT IT
003262  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;EQUAL ?
003270  001401                        BEQ     2$                      ;BR IF EQUAL
003272  104003                        ERROR   3

003274  006337  001124          2$:     ASL     $GDDAT                  ;SHIFT LEFT
003300  001355                        BNE     1$                      ;BRANCH UNTIL DONE

                                ;;*****************************************************************
                                ;*TEST 11        TEST FOR SLOW OUTPUT GATES WITH #125252
                                ;;*****************************************************************
003302  000004                  TST11:  SCOPE
003304  012737  125252  001124          MOV     #125252,$GDDAT          ;LOAD EXPECTED VALUE
003312  013777  001124  176120          MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT
003320  005177  176114                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003324  005177  176110                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003330  005177  176104                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003334  005177  176100                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003340  005177  176074                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003344  005177  176070                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003350  005177  176064                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003354  005177  176060                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003360  005177  176054                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003364  005177  176050                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003370  000240                        NOP
003372  000240                        NOP
003374  017737  176040  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
003402  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST REGISTER
003410  001401                        BEQ     TST12           ;;BR IF CORRECT
003412  104003                        ERROR   3
```

```
                                ;;*****************************************************************
                                ;*TEST 12        TEST FOR SLOW OUTPUT GATES WITH #52525
                                ;;*****************************************************************
003414  000004          TST12:  SCOPE
003416  012737  052525  001124          MOV     #52525,$GDDAT           ;LOAD EXPECTED VALUE
003424  013777  001124  176006          MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT REG.
003432  005177  176002                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003436  005177  175776                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003442  005177  175772                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003446  005177  175766                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003452  005177  175762                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003456  005177  175756                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003462  005177  175752                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003466  005177  175746                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003472  005177  175742                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003476  005177  175736                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
003502  000240                          NOP
003504  017737  175730  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
003512  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST PATTERN
003520  001401                          BEQ     TST13           ;;BR IF EQUAL
003522  104003                          ERROR   3                       ;OUTPUT REGISTER IN ERROR

                                ;;*****************************************************************
                                ;*TEST 13        TEST THAT OUTPUT CAN HOLD BYTE COUNT PATTERN
                                ;;*****************************************************************
003524  000004          TST13:  SCOPE
003526  012737  000400  001164          MOV     #400,$TIMES     ;;DO 400 ITERATIONS
003534  012737  003546  001110          MOV     #2$,$LPERR              ;LOAD SCOPE ERROR RETURN
003542  005037  001124                  CLR     $GDDAT                  ;CLEAR PATTERN
003546  113777  001124  175664  2$:     MOVB    $GDDAT,@GRDIO           ;LOAD THE OUTPUT
003554  017737  175660  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
003562  123737  001124  001126          CMPB    $GDDAT,$BDDAT
003570  001401                          BEQ     1$                      ;BRANCH IF EQUAL
003572  104003                          ERROR   3                       ;ERROR, LOW BYTE HAS BAD DATA
003574  105237  001124          1$:     INCB    $GDDAT                  ;UPDATE PATTERN
003600  001362                          BNE     2$

                                ;;*****************************************************************
                                ;*TEST 14        TEST THAT OUTPUT CAN HOLD HIGH BYTE COUNT PATTERN
                                ;;*****************************************************************
003602  000004          TST14:  SCOPE
003604  012737  000400  001164          MOV     #400,$TIMES     ;;DO 400 ITERATIONS
003612  012737  003624  001110          MOV     #2$,$LPERR              ;LOAD SCOPE ERROR RETURN
003620  005037  001124                  CLR     $GDDAT                  ;CLEAR PATTERN
003624  113777  001125  175610  2$:     MOVB    $GDDAT+1,@GRBHIO        ;LOAD THE HIGH BYTE
003632  117737  175604  001127          MOVB    @GRBHIO,$BDDAT+1        ;READ HIGH BYTE OUTPUT REG.
003640  123737  001125  001127          CMPB    $GDDAT+1,$BDDAT+1
003646  001401                          BEQ     1$                      ;BRANCH IF EQUAL
003650  104003                          ERROR   3                       ;ERROR, HIGH BYTE IN ERROR
003652  105237  001125          1$:     INCB    $GDDAT+1
003656  001362                          BNE     2$
```

```
                                        ;;*****************************************************************
                                        ;*TEST 15        TEST RELAY #1 STATUS BIT
                                        ;;*****************************************************************
003660  000004                  TST15:  SCOPE
003662  005077  175552                  CLR     @GRDIO
003666  012777  177777  175542          MOV     #-1,@GRDAI
003674  012737  000001  001124          MOV     #BIT0,$GDDAT            ;LOAD EXPECTED
003702  012777  000001  175524          MOV     #BIT0,@GRSTAT           ;SET BIT 0
003710  017737  175520  001126          MOV     @GRSTAT,$BDDAT          ;READ STATUS
003716  033737  001124  001126          BIT     $GDDAT,$BDDAT           ;TEST IT
003724  001001                          BNE     TST16           ;;BR IF SET
003726  104001                          ERROR   1                       ;ERROR, RELAY 1 FAILED TO SET

                                        ;;*****************************************************************
                                        ;*TEST 16        TEST RELAY #2 STATUS BIT
                                        ;;*****************************************************************
003730  000004                  TST16:  SCOPE
003732  012737  000400  001124          MOV     #BIT8,$GDDAT            ;LOAD EXPECT
003740  012777  000400  175466          MOV     #BIT8,@GRSTAT           ;SET RELAY 2
003746  017737  175462  001126          MOV     @GRSTAT,$BDDAT          ;READ STATUS
003754  033737  001124  001126          BIT     $GDDAT,$BDDAT
003762  001001                          BNE     TST17           ;;BR IF SET
003764  104001                          ERROR   1                       ;ERROR, RELAY 2 FAILED TO SET

                                        ;;*****************************************************************
                                        ;*TEST 17        TEST OUTPUT DATA ACCEPT FLAG
                                        ;;*****************************************************************
003766  000004                  TST17:  SCOPE
003770  012737  100000  001124          MOV     #BIT15,$GDDAT           ;LOAD EXPECTED
003776  012777  100000  175430          MOV     #BIT15,@GRSTAT          ;SET BIT 15
004004  017737  175424  001126          MOV     @GRSTAT,$BDDAT          ;READ STATUS
004012  033737  001124  001126          BIT     $GDDAT,$BDDAT
004020  001001                          BNE     TST20           ;;BR IF SET
004022  104001                          ERROR   1                       ;ERROR, BIT 15 FAILED TO SET

                                        ;;*****************************************************************
                                        ;*TEST 20        TEST OUTPUT INTERRUPT ENABLE
                                        ;;*****************************************************************
004024  000004                  TST20:  SCOPE
004026  005077  175402                  CLR     @GRSTAT                 ;CLEAR STATUS
004032  012737  040000  001124          MOV     #BIT14,$GDDAT           ;LOAD EXPECTED
004040  012777  040000  175366          MOV     #BIT14,@GRSTAT          ;LOAD BIT 14
004046  017737  175362  001126          MOV     @GRSTAT,$BDDAT          ;READ STATUS
004054  033737  001124  001126          BIT     $GDDAT,$BDDAT
004062  001001                          BNE     TST21           ;;BR IF SET
004064  104001                          ERROR   1                       ;ERROR BIT 14 FAILED TO SET
```

```
                                        ;;*****************************************************************
                                        ;*TEST 21        TEST INPUT DATA READY FLAG
                                        ;;*****************************************************************
004066  000004                  TST21:  SCOPE
004070  012737  000200  001124          MOV     #BIT7,$GDDAT                    ;LOAD EXPECTED
004076  012777  000200  175330          MOV     #BIT7,@GRSTAT                   ;SET BIT 7
004104  017737  175324  001126          MOV     @GRSTAT,$BDDAT                  ;READ RESULT
004112  023737  001124  001126          CMP     $GDDAT,$BDDAT                   ;COMPARE
004120  001401                          BEQ     TST22           ;;BR IF EQUAL
004122  104001                          ERROR   1                               ;ERROR, BIT 7 FAILED TO SET

                                        ;;*****************************************************************
                                        ;*TEST 22        TEST INPUT INTERRUPT ENABLE
                                        ;;*****************************************************************
004124  000004                  TST22:  SCOPE
004126  005077  175302                  CLR     @GRSTAT                         ;CLEAR STATUS
004132  012737  000100  001124          MOV     #BIT6,$GDDAT                    ;LOAD EXPECTED
004140  012777  000100  175266          MOV     #BIT6,@GRSTAT                   ;SET BIT 6
004146  017737  175262  001126          MOV     @GRSTAT,$BDDAT                  ;READ RESULT
004154  023737  001124  001126          CMP     $GDDAT,$BDDAT                   ;COMPARE
004162  001401                          BEQ     TST23           ;;BR IF EQUAL
004164  104001                          ERROR   1                               ;ERROR, BIT 6 FAILED TO SET

                                        ;;*****************************************************************
                                        ;*TEST 23        TEST THAT RESET CLEARS THE DIGITAL STATUS REGISTER
                                        ;;*****************************************************************
004166  000004                  TST23:  SCOPE
004170  012737  000040  001164          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
004176  005077  175232                  CLR     @GRSTAT                         ;CLEAR STATUS
004202  005037  001124                  CLR     $GDDAT                          ;LOAD EXPECTED
004206  012777  140701  175220          MOV     #BIT15!BIT14!BIT8!BIT7!BIT6!BIT0,@GRSTAT
004214  000005                          RESET
004216  017737  175212  001126          MOV     @GRSTAT,$BDDAT                  ;READ RESULT
004224  001401                          BEQ     TST24           ;;BR IF EQUAL
004226  104001                          ERROR   1                               ;ERROR, RESET FAILED TO CLEAR DIGITAL
                                                                                ;STATUS REGISTER
```

```
                                        ;;*****************************************************************
                                        ;*TEST 24       TEST EXTERNAL TRANSFERS - CABLE MUST BE CONNECTED
                                        ;;*****************************************************************
004230  000004                        TST24:  SCOPE
004232  032777  002000  174676              BIT     #BIT10,@SWR             ;TEST SWITCH BIT
004240  001402                              BEQ     1$                      ;BRANCH IF DOWN
004242  000137  006156                      JMP     DRT21                   ;BYPASS SOME TEST USING THE EXTERNAL CABLE
004246                                  1$:
004246  005077  175166                      CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
004252  012777  177777  175156              MOV     #-1,@GRDAI              ;CLEAR INPUT
004260  005037  001124                      CLR     $GDDAT                  ;CLEAR EXPECTED
004264  012777  000000  175146              MOV     #0,@GRDIO               ;LOAD THE OUTPUT
004272  017737  175140  001126              MOV     @GRDAI,$BDDAT           ;READ THE INPUT
004300  023737  001124  001126              CMP     $GDDAT,$BDDAT           ;COMPARE
004306  001401                              BEQ     TST25           ;;BR IF EQUAL
004310  104002                              ERROR   2                       ;ERROR, INPUT DID NOT EQUAL THE OUTPUT REG.

                                        ;;*****************************************************************
                                        ;*TEST 25       TEST INPUT WITH #-1
                                        ;;*****************************************************************
004312  000004                        TST25:  SCOPE
004314  005077  175120                      CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
004320  012777  177777  175110              MOV     #-1,@GRDAI              ;CLEAR INPUT
004326  012737  177777  001124              MOV     #-1,$GDDAT              ;LOAD EXPECTED
004334  012777  177777  175076              MOV     #-1,@GRDIO              ;LOAD THE OUTPUT
004342  017737  175070  001126              MOV     @GRDAI,$BDDAT           ;READ INPUT
004350  023737  001124  001126              CMP     $GDDAT,$BDDAT           ;COMPARE
004356  001401                              BEQ     TST26           ;;BR IF EQUAL
004360  104002                              ERROR   2                       ;ERROR, INPUT DID NOT EQUAL THE OUTPUT
                                                                        ;IS WRAP-AROUND CABLE CONNECTED ???

                                        ;;*****************************************************************
                                        ;*TEST 26       TEST INPUT WITH #52525
                                        ;;*****************************************************************
004362  000004                        TST26:  SCOPE
004364  005077  175050                      CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
004370  012777  177777  175040              MOV     #-1,@GRDAI              ;CLEAR INPUT
004376  012737  052525  001124              MOV     #52525,$GDDAT           ;LOAD EXPECTED
004404  012777  052525  175026              MOV     #52525,@GRDIO           ;LOAD THE OUTPUT
004412  017737  175020  001126              MOV     @GRDAI,$BDDAT           ;READ INPUT
004420  023737  001124  001126              CMP     $GDDAT,$BDDAT           ;COMPARE
004426  001401                              BEQ     TST27           ;;BR IF EQUAL
004430  104002                              ERROR   2                       ;ERROR, INPUT DID NOT EQUAL OUTPUT
```

```
                                        ;;*****************************************************************
                                        ;*TEST 27       TEST INPUT WITH #125252
                                        ;;*****************************************************************
004432  000004                  TST27:  SCOPE
004434  005077  175000                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
004440  012777  177777  174770          MOV     #-1,@GRDAI              ;CLEAR INPUT
004446  012737  125252  001124          MOV     #125252,$GDDAT          ;LOAD EXPECTED
004454  012777  125252  174756          MOV     #125252,@GRDIO          ;LOAD THE OUTPUT
004462  017737  174750  001126          MOV     @GRDAI,$BDDAT           ;READ INPUT
004470  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004476  001401                          BEQ     TST30           ;;BR IF EQUAL
004500  104002                          ERROR   2                       ;ERROR, INPUT DID NOT EQUAL OUTPUT

                                        ;;*****************************************************************
                                        ;*TEST 30       FLOAT A 1 ACROSS NON-LATCHING INPUT BITS
                                        ;;*****************************************************************
004502  000004                  TST30:  SCOPE
004504  012737  004532  001110          MOV     #2$,$LPERR              ;LOAD ERROR SCOPE RETURN

004512  012737  000001  001464          MOV     #BIT0,BRLEV2            ;LOAD EXPECTED
004520  013737  001444  001466          MOV     NOTLCH,BRLEV3           ;GET NON-LATCH
004526  005137  001466                  COM     BRLEV3                  ;COMPLEMENT
004532  013737  001464  001124  2$:     MOV     BRLEV2,$GDDAT           ;LOAD GOOD
004540  033737  001124  001444          BIT     $GDDAT,NOTLCH           ;TEST FOR NON-LATCH
004546  001436                          BEQ     1$                      ;BR IF LATCHING

004550  013777  001124  174662          MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT
004556  017737  174654  001126          MOV     @GRDAI,$BDDAT           ;READ INPUT
004564  043737  001466  001126          BIC     BRLEV3,$BDDAT           ;MASK TO LATCH BITS
004572  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004600  001401                          BEQ     3$              ;;BR IF EQUAL
004602  104002                          ERROR   2                       ;INPUT REGISTER IN ERROR
                                                                        ;WAS CORRECT LATCH/NON-LATCH SUPPLIED ?

                                ;SUB-TEST CLEAR THE OUTPUT BIT AND TEST THE INPUT DOES NOT LATCH

004604  043777  001124  174626  3$:     BIC     $GDDAT,@GRDIO           ;CLEAR OUTPUT BIT
004612  017737  174620  001126          MOV     @GRDAI,$BDDAT           ;READ INPUT
004620  043737  001466  001126          BIC     BRLEV3,$BDDAT           ;MASK TO LATCH BITS
004626  005037  001124                  CLR     $GDDAT                  ;CLEAR EXPECTED
004632  033737  001464  001126          BIT     BRLEV2,$BDDAT           ;TEST FOR BIT
004640  001401                          BEQ     1$              ;;BR IF CLEARED
004642  104002                          ERROR   2                       ;INPUT BIT LATCHED IN ERROR
                                                                        ;WAS CORRECT LATCH/NON-LATCH SUPPLIED ?

004644  006337  001464          1$:     ASL     BRLEV2                  ;CHANGE PATTERN
004650  001330                          BNE     2$                      ;BR UNTIL DONE
```

```
1033
1034                                    ;;*****************************************************************
1035                                    ;*TEST 31       FLOAT A 1 ACROSS LATCHING INPUT BITS
1036                                    ;;*****************************************************************
1037  004652  000004                    TST31:  SCOPE
1038  004654  012737  004670  001110            MOV     #2$,$LPERR              ;LOAD ERROR SCOPE RETURN
1039  004662  012737  000001  001124            MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE
1040
1041  004670  033737  001124  001444  2$:      BIT     $GDDAT,NOTLCH           ;TEST FOR NON-LATCHING
1042  004676  001022                            BNE     1$                      ;BR IF NON-LATCH
1043
1044  004700  005077  174534                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1045  004704  012777  177777  174524            MOV     #-1,@GRDAI              ;CLEAR INPUT
1046
1047  004712  013777  001124  174520            MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT
1048  004720  005077  174514                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1049  004724  017737  174506  001126            MOV     @GRDAI,$BDDAT           ;READ INPUT REG.
1050  004732  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1051  004740  001401                            BEQ     1$              ;;BR IF EQUAL
1052  004742  104002                            ERROR   2                       ;INPUT REGISTER FAILED TO LATCH DATA
1053                                                                    ;IS THIS REALLY A "LPS-11-DRA" ??       FIRST ERROR
1054                                                                    ;IF RUNNING ON A "LPS-11-DR"  IF "DR" USE MD-11-DZLPD
1055
1056  004744  006337  001124            1$:     ASL     $GDDAT                  ;CHANGE PATTERN
1057  004750  001347                            BNE     2$                      ;BR UNTIL DONE
1058
1059                                    ;;*****************************************************************
1060                                    ;*TEST 32       FLOAT A 0 ACROSS LATCHING INPUT BITS
1061                                    ;;*****************************************************************
1062  004752  000004                    TST32:  SCOPE
1063  004754  012737  004770  001110            MOV     #2$,$LPERR              ;LOAD ERROR SCOPE RETURN
1064  004762  012737  000001  001466            MOV     #BIT0,BRLEV3            ;LOAD EXPECTED
1065
1066  004770  033737  001466  001444  2$:      BIT     BRLEV3,NOTLCH           ;TEST FOR LATCHING
1067  004776  001035                            BNE     1$                      ;BR IF NOT
1068
1069  005000  005077  174434                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1070  005004  012777  177777  174424            MOV     #-1,@GRDAI              ;CLEAR INPUT
1071  005012  012737  177777  001124            MOV     #-1,$GDDAT              ;LOAD
1072  005020  043737  001444  001124            BIC     NOTLCH,$GDDAT
1073  005026  000240                            NOP
1074  005030  000240                            NOP
1075  005032  043737  001466  001124            BIC     BRLEV3,$GDDAT           ;MAKE BRLEV3
1076  005040  013777  001124  174372            MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT
1077  005046  005077  174366                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1078
1079  005052  017737  174360  001126            MOV     @GRDAI,$BDDAT           ;READ INPUT
1080  005060  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1081  005066  001401                            BEQ     1$              ;;BR IF EQUAL
1082  005070  104002                            ERROR   2                       ;INPUT REGISTER FAILED TO LATCH DATA
1083
1084  005072  006337  001466            1$:     ASL     BRLEV3                  ;CHANGE PATTERN
1085  005076  001334                            BNE     2$                      ;BR UNTIL DONE
1086
```

```
1087
1088                                    ;;*****************************************************************
1089                                    ;*TEST 33       TEST FOR SLOW INPUT GATES WITH #125252
1090                                    ;;*****************************************************************
1091  005100  000004                   TST33:  SCOPE
1092
1093  005102  012737  125252  001124           MOV     #125252,$GDDAT          ;LOAD EXPECTED
1094  005110  043737  001444  001124           BIC     NOTLCH,$GDDAT           ;CONVERT
1095  005116  013700  001124                  MOV     $GDDAT,R0               ;LOAD PATTERN
1096  005122  005077  174312                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1097  005126  012777  177777  174302          MOV     #-1,@GRDAI              ;CLEAR INPUT
1098
1099  005134  010077  174300                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1100  005140  005077  174274                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1101  005144  017701  174266                  MOV     @GRDAI,R1               ;READ INPUT
1102  005150  050177  174262                  BIS     R1,@GRDAI               ;CLEAR INPUT
1103  005154  005100                          COM     R0
1104  005156  010077  174256                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1105  005162  005077  174252                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1106  005166  017701  174244                  MOV     @GRDAI,R1               ;READ INPUT
1107  005172  050177  174240                  BIS     R1,@GRDAI               ;CLEAR INPUT
1108  005176  005100                          COM     R0
1109  005200  010077  174234                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1110  005204  005077  174230                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1111  005210  017701  174222                  MOV     @GRDAI,R1               ;READ INPUT
1112  005214  050177  174216                  BIS     R1,@GRDAI               ;CLEAR INPUT
1113  005220  005100                          COM     R0
1114  005222  010077  174212                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1115  005226  005077  174206                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1116  005232  017701  174200                  MOV     @GRDAI,R1               ;READ INPUT
1117  005236  050177  174174                  BIS     R1,@GRDAI               ;CLEAR INPUT
1118  005242  005100                          COM     R0
1119  005244  010077  174170                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1120  005250  005077  174164                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1121  005254  017701  174156                  MOV     @GRDAI,R1               ;READ INPUT
1122  005260  050177  174152                  BIS     R1,@GRDAI               ;CLEAR INPUT
1123  005264  005100                          COM     R0
1124  005266  010077  174146                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1125  005272  005077  174142                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1126  005276  017701  174134                  MOV     @GRDAI,R1               ;READ INPUT
1127  005302  050177  174130                  BIS     R1,@GRDAI               ;CLEAR INPUT
1128  005306  005100                          COM     R0
1129  005310  010077  174124                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1130  005314  005077  174120                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1131  005320  017701  174112                  MOV     @GRDAI,R1               ;READ INPUT
1132  005324  050177  174106                  BIS     R1,@GRDAI               ;CLEAR INPUT
1133  005330  005100                          COM     R0
1134  005332  010077  174102                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1135  005336  005077  174076                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1136  005342  017701  174070                  MOV     @GRDAI,R1               ;READ INPUT
1137  005346  050177  174064                  BIS     R1,@GRDAI               ;CLEAR INPUT
1138  005352  005100                          COM     R0
1139  005354  010077  174060                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1140  005360  005077  174054                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1141  005364  017701  174046                  MOV     @GRDAI,R1               ;READ INPUT
1142  005370  050177  174042                  BIS     R1,@GRDAI               ;CLEAR INPUT
```

```
 1143  005374  005100                          COM     R0
 1144  005376  010077  174036                  MOV     R0,@GRDIO               ;LOAD OUTPUT
 1145  005402  005077  174032                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
 1146  005406  017701  174024                  MOV     @GRDAI,R1               ;READ INPUT
 1147  005412  050177  174020                  BIS     R1,@GRDAI               ;CLEAR INPUT
 1148  005416  005100                          COM     R0
 1149  005420  010077  174014                  MOV     R0,@GRDIO               ;LOAD OUTPUT
 1150  005424  005077  174010                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
 1151  005430  017701  174002                  MOV     @GRDAI,R1               ;READ INPUT
 1152  005434  050177  173776                  BIS     R1,@GRDAI               ;CLEAR INPUT
 1153  005440  005100                          COM     R0
 1154
 1155  005442  010137  001126                  MOV     R1,$BDDAT               ;LOAD READ
 1156  005446  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
 1157  005454  001401                          BEQ     TST34           ;;BR IF EQUAL
 1158  005456  104002                          ERROR   2                       ;INPUT GATE SLOW
 1159
 1160                          ;;*****************************************************************
 1161                          ;*TEST 34       TEST FOR SLOW INPUT GATES WITH #52525
 1162                          ;;*****************************************************************
 1163  005460  000004                  TST34:  SCOPE
 1164
 1165  005462  012737  052525  001124          MOV     #52525,$GDDAT           ;SETUP EXPECTED
 1166  005470  043737  001444  001124          BIC     NOTLCH,$GDDAT           ;CONVERT
 1167  005476  013700  001124                  MOV     $GDDAT,R0
 1168  005502  005077  173732                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
 1169  005506  012777  177777  173722          MOV     #-1,@GRDAI              ;CLEAR INPUT
 1170
 1171  005514  010077  173720                  MOV     R0,@GRDIO               ;LOAD OUTPUT
 1172  005520  005077  173714                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
 1173  005524  017701  173706                  MOV     @GRDAI,R1               ;READ INPUT
 1174  005530  050177  173702                  BIS     R1,@GRDAI               ;CLEAR INPUT
 1175  005534  005100                          COM     R0
 1176  005536  010077  173676                  MOV     R0,@GRDIO               ;LOAD OUTPUT
 1177  005542  005077  173672                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
 1178  005546  017701  173664                  MOV     @GRDAI,R1               ;READ INPUT
 1179  005552  050177  173660                  BIS     R1,@GRDAI               ;CLEAR INPUT
 1180  005556  005100                          COM     R0
 1181  005560  010077  173654                  MOV     R0,@GRDIO               ;LOAD OUTPUT
 1182  005564  005077  173650                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
 1183  005570  017701  173642                  MOV     @GRDAI,R1               ;READ INPUT
 1184  005574  050177  173636                  BIS     R1,@GRDAI               ;CLEAR INPUT
 1185  005600  005100                          COM     R0
 1186  005602  010077  173632                  MOV     R0,@GRDIO               ;LOAD OUTPUT
 1187  005606  005077  173626                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
 1188  005612  017701  173620                  MOV     @GRDAI,R1               ;READ INPUT
 1189  005616  050177  173614                  BIS     R1,@GRDAI               ;CLEAR INPUT
 1190  005622  005100                          COM     R0
 1191  005624  010077  173610                  MOV     R0,@GRDIO               ;LOAD OUTPUT
 1192  005630  005077  173604                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
 1193  005634  017701  173576                  MOV     @GRDAI,R1               ;READ INPUT
 1194  005640  050177  173572                  BIS     R1,@GRDAI               ;CLEAR INPUT
 1195  005644  005100                          COM     R0
 1196  005646  010077  173566                  MOV     R0,@GRDIO               ;LOAD OUTPUT
 1197  005652  005077  173562                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
 1198  005656  017701  173554                  MOV     @GRDAI,R1               ;READ INPUT
```

```
:199  005662  050177  173550          BIS     R1,@GRDAI               ;CLEAR INPUT
:200  005666  005100                  COM     R0
:201  005670  010077  173544          MOV     R0,@GRDIO               ;LOAD OUTPUT
:202  005674  005077  173540          CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
:203  005700  017701  173532          MOV     @GRDAI,R1               ;READ INPUT
:204  005704  050177  173526          BIS     R1,@GRDAI               ;CLEAR INPUT
:205  005710  005100                  COM     R0
:206  005712  010077  173522          MOV     R0,@GRDIO               ;LOAD OUTPUT
:207  005716  005077  173516          CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
:208  005722  017701  173510          MOV     @GRDAI,R1               ;READ INPUT
:209  005726  050177  173504          BIS     R1,@GRDAI               ;CLEAR INPUT
:210  005732  005100                  COM     R0
:211  005734  010077  173500          MOV     R0,@GRDIO               ;LOAD OUTPUT
:212  005740  005077  173474          CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
:213  005744  017701  173466          MOV     @GRDAI,R1               ;READ INPUT
:214  005750  050177  173462          BIS     R1,@GRDAI               ;CLEAR INPUT
:215  005754  005100                  COM     R0
:216  005756  010077  173456          MOV     R0,@GRDIO               ;LOAD OUTPUT
:217  005762  005077  173452          CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
:218  005766  017701  173444          MOV     @GRDAI,R1               ;READ INPUT
:219  005772  050177  173440          BIS     R1,@GRDAI               ;CLEAR INPUT
:220  005776  005100                  COM     R0
:221  006000  010077  173434          MOV     R0,@GRDIO               ;LOAD OUTPUT
:222  006004  005077  173430          CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
:223  006010  017701  173422          MOV     @GRDAI,R1               ;READ INPUT
:224  006014  050177  173416          BIS     R1,@GRDAI               ;CLEAR INPUT
:225  006020  005100                  COM     R0
:226
:227  006022  010137  001126          MOV     R1,$BDDAT               ;LOAD VALUE READ
:228  006026  023737  001124  001126  CMP     $GDDAT,$BDDAT           ;COMPARE
:229  006034  001401                  BEQ     TST35           ;;BR IF EQUAL
:230  006036  104002                  ERROR   2
```

```
                                        ;;*****************************************************************
                                        ;*TEST 35       TEST THAT WHEN OUTPUTTING THE INPUT DATA READY FLAG SETS
                                        ;;*****************************************************************
006040  000004                  ;ST35:  SCOPE
006042  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
006050  005077  173360                  CLR     @GRSTAT
006054  012777  000000  173356          MOV     #0,@GRDIO               ;OUTPUT 0
006062  022727  000000  000000          CMP     #0,#0                   ;DELAY
006070  017737  173340  001126          MOV     @GRSTAT,$BDDAT          ;READ RESULTS
006076  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
006104  001401                        BEQ     ST36            ;;BR IF EQUAL
006106  104001                        ERROR   1                       ;INPUT DATA READY FLAG FAILED TO SET

                                        ;TEST THAT WHEN THE INPUT BUFFER IS READ THE OUTPUT FLAG IS SET

                                        ;;*****************************************************************
                                        ;*TEST 36       TEST THAT WHEN THE INPUT BUFFER IS READ THE OUTPUT FLAG IS SET
                                        ;;*****************************************************************
006110  000004                  ;ST36:  SCOPE
006112  012737  100000  001124          MOV     #BIT15,$GDDAT           ;LOAD EXPECTED
006120  005077  173310                  CLR     @GRSTAT
006124  012777  000000  173306          MOV     #0,@GRDIO
006132  022727  000000  000000          CMP     #0,#0
006140  017700  173272                  MOV     @GRDAI,R0               ;READ INPUT
006144  017737  173264  001126          MOV     @GRSTAT,$BDDAT          ;READ RESULTS
006152  100401                        BMI     ST37            ;;BR IF SET
006154  104001                        ERROR   1                       ;INPUT DATA READY FLAG FAILED TO SET
```

```
                                        ;TEST THAT THE DIGITAL I/O DOES NOT INTERRUPT
                                        ;SET INPUT-OUTPUT FLAGS BUT DO NOT ENABLE INTERRUPTS

006156                                  DRT21:
                                        ;;*****************************************************************
                                        ;*TEST 37        TEST FOR UNEXPECTED INTERRUPT
                                        ;;*****************************************************************
006156  000004                  TST37:  SCOPE
006160  012737  000040  001164          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
006166  000005                          RESET
006170  005077  173240                  CLR     @GRSTAT
006174  005037  177776                  CLR     PSW
006200  012777  006260  173242          MOV     #1$,@GRIVA              ;SET UP INTERRUPT INPUT VECTOR
006206  012777  000340  173236          MOV     #340,@GRIVSA
006214  012777  006264  173232          MOV     #2$,@GRIVB              ;SET UP INTERRUPT OUTPUT VECTOR
006222  012777  000340  173226          MOV     #340,@GRIVSB
006230  012777  000000  173202          MOV     #0,@GRDIO               ;OUTPUT
006236  017700  173174                  MOV     @GRDAI,R0               ;INPUT
006242  000240                          NOP
006244  000240                          NOP
006246  000240                          NOP
006250  000240                          NOP
006252  005077  173156                  CLR     @GRSTAT                 ;CLEAR STATUS
006256  000404                          BR      TST40           ;;

006260  104006                  1$:     ERROR   6                       ;ERROR, DIGITAL INPUT INTERRUPTED
006262  000002                          RTI

006264  104006                  2$:     ERROR   6                       ;ERROR, DIGITAL OUTPUT INTERRUPTED
006266  000002                          RTI
```

```
1290
1291                                    ;;*****************************************************************
1292                                    ;*TEST 40       TEST THAT THE INPUT CAN INT. USING THE MAINT. BIT
1293                                    ;;*****************************************************************
1294  006270  000004                    TST40:  SCOPE
1295  006272  000240                            NOP
1296  006274  000240                            NOP
1297  006276  005077  173132                   CLR     @GRSTAT                 ;CLEAR STATUS
1298  006302  012777  006344  173140           MOV     #1$,@GRIVA              ;LOAD RETURN VECTOR
1299  006310  012777  000340  173134           MOV     #340,@GRIVSA            ;
1300  006316  013777  001456  173130           MOV     GRIVSB,@GRIVB           ;LOAD OUTPUT VECTOR
1301  006324  012777  000000  173124           MOV     #0,@GRIVSB
1302  006332  052777  000040  173074           BIS     #BIT5,@GRSTAT           ;MAINT. INT C
1303  006340  000240                            NOP
1304  006342  104004                            ERROR   4                       ;ERROR, INPUT FAILED TO INTERRUPT
1305
1306                                    ;SUB-TEST, TEST THAT IF PSW IS LOWERED AGAIN NO INTERRUPT WILL OCCUR
1307                                    ; IF INTERRUPT OCCURS 'INT DONE C' FAILED TO CLEAR INT C FLOP
1308
1309  006344  012777  006366  173075    1$:     MOV     #2$,@GRIVA              ;LOAD INPUT VECTOR
1310  006352  022626                            CMP     (SP)+,(SP)+             ;POP THE STACK *4
1311  006354  005037  177776                   CLR     PSW                     ;LOWER PRIOR.
1312  006360  000240                            NOP
1313  006362  000240                            NOP
1314  006364  000404                            BR      TST41           ;;<NEXT TEST>
1315
1316  006366  022626                    2$:     CMP     (SP)+,(SP)+             ;POP THE STACK
1317  006370  104006                            ERROR   6                       ;UNEXPECTED INTERRUPT
1318  006372  005037  177776                   CLR     PSW
```

```
1319
1320                                    ;;*****************************************************************
1321                                    ;*TEST 41       TEST THAT THE INPUT INT. CLEARS INT. ENABLE VIA MAINT. BIT
1322                                    ;;*****************************************************************
1323  006376  000004                    TST41:  SCOPE
1324  006400  000240                            NOP
1325  006402  000240                            NOP
1326  006404  005077  173024                    CLR     @GRSTAT                 ;CLEAR STATUS
1327  006410  012777  006460  173032            MOV     #1$,@GRIVA              ;LOAD RETURN VECTOR
1328  006416  012777  000340  173026            MOV     #340,@GRIVSA            ;
1329  006424  013777  001456  173022            MOV     GRIVSB,@GRIVB           ;LOAD OUTPUT VECTOR
1330  006432  012777  000000  173016            MOV     #0,@GRIVSB
1331  006440  012777  000100  172766            MOV     #BIT6,@GRSTAT           ;LOAD INPUT INT. ENABLE
1332  006446  052777  000040  172760            BIS     #BIT5,@GRSTAT           ;MAINT. INT C
1333  006454  000240                            NOP
1334  006456  104004                            ERROR   4                       ;ERROR, INPUT FAILED TO INTERRUPT
1335
1336  006460  012777  006522  172762  1$:       MOV     #2$,@GRIVA              ;LOAD INPUT VECTOR
1337  006466  022626                            CMP     (SP)+,(SP)+             ;POP THE STACK *4
1338  006470  005037  177776                    CLR     PSW                     ;LOWER PRIOR.
1339  006474  005037  001124                    CLR     $GDDAT                  ;CLEAR EXPECTED
1340  006500  017737  172730  001125            MOV     @GRSTAT,$BDDAT          ;READ STATUS
1341  006506  032737  000100  001126            BIT     #BIT6,$BDDAT            ;TEST BIT 6
1342  006514  001406                            BEQ     TST42           ;;BR IF CLEARED
1343  006516  104001                            ERROR   1                       ;INPUT INT. FAILED TO CLEAR
1344  006520  000404                            BR      TST42           ;;<NEXT TEST>
1345
1346  006522  022626                    2$:     CMP     (SP)+,(SP)+             ;POP THE STACK
1347  006524  104006                            ERROR   6                       ;UNEXPECTED INTERRUPT
1348  006526  005037  177776                    CLR     PSW
```

```
1349
1350
1351
1352                            ;;**************************************************************
1353                            ;*TEST 42       TEST THAT THE OUTPUT CAN INT. USING THE MAINT. BIT
1354                            ;;**************************************************************
1355  006532  000004           TST42:  SCOPE
1356  006534  000240                   NOP
1357  006536  005077  172672           CLR     @GRSTAT                 ;CLEAR STATUS
1358  006542  013777  001452  172700   MOV     GRIVSA,@GRIVA           ;LOAD INPUT VECTOR
1359  006550  012777  000000  172674   MOV     #0,@GRIVSA
1360  006556  012777  006604  172670   MOV     #1$,@GRIVB              ;LOAD OUTPUT VECTOR
1361  006564  012777  000340  172664   MOV     #340,@GRIVSB
1362  006572  052777  020000  172634   BIS     #BIT13,@GRSTAT          ;MAINT. INTERRUPT
1363  006600  000240                   NOP
1364  006602  104005                   ERROR   5                       ;OUTPUT FAILED TO INTERRUPT
1365
1366                            ;SUB-TEST, TEST THAT IF PSW IS LOWERED AGAIN, NO INTERRUPT WILL OCCUR
1367                            ; IF IT DOES, 'INT DONE D HIGH' FAILED TO CLEAR 'INT D' FLOP
1368
1369  006604  012777  006626  172642  1$:     MOV     #2$,@GRIVB              ;LOAD OUTPUT VECTOR
1370  006612  022626                   CMP     (SP)+,(SP)+             ;POP THE STACK *4
1371  006614  005037  177776           CLR     PSW
1372  006620  000240                   NOP
1373  006622  000240                   NOP
1374  006624  000404                   BR      TST43           ;;<NEXT TEST>
1375  006626  022626           2$:     CMP     (SP)+,(SP)+             ;POP THE STACK
1376  006630  104006                   ERROR   6                       ;UNEXPECTED INTERRUPT
1377  006632  005037  177776           CLR     PSW
1378
```

```
1379
1380                                    ;;*****************************************************************
1381                                    ;*TEST 43       TEST FOR INT. FROM DRA INPUT
1382                                    ;;*****************************************************************
1383  006636  000004                    TST43:  SCOPE
1384  006640  000240                            NOP
1385  006642  000240                            NOP
1386  006644  032777  002000  172264            BIT     #BIT10,@SWR             ;TEST SWITCH
1387  006652  001401                            BEQ     1$
1388  006654  000455                            BR      TST44           ;;
1389  006656  005077  172552            1$:     CLR     @GRSTAT
1390  006662  012777  006752  172560            MOV     #2$,@GRIVA              ;IN CASE OF INTERRUPT
1391  006670  012777  000340  172554            MOV     #340,@GRIVSA
1392  006676  013777  001456  172550            MOV     GRIVSB,@GRIVB
1393  006704  005077  172546                    CLR     @GRIVSB
1394  006710  012777  000100  172516            MOV     #BIT6,@GRSTAT           ;ENABLE INPUT INT.
1395  006716  012777  000000  172514            MOV     #0,@GRDIO               ;OUTPUT
1396  006724  017700  172506                    MOV     @GRDAI,R0               ;INPUT
1397  006730  000240                            NOP                             ;LET INTERRUPT OCCUR
1398  006732  000240                            NOP
1399  006734  000240                            NOP
1400  006736  042777  000100  172470            BIC     #BIT6,@GRSTAT
1401  006744  000240                            NOP
1402  006746  104004                            ERROR   4                       ;ERROR, INPUT FAILED TO INTERRUPT
1403  006750  000417                            BR      TST44           ;;
1404  006752  022626                    2$:     CMP     (SP)+,(SP)+
1405  006754  005037  177776                    CLR     PSW
1406  006760  005077  172450                    CLR     @GRSTAT                 ;CLEAR STATUS
1407  006764  013777  001452  172456            MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
1408  006772  005077  172454                    CLR     @GRIVSA
1409  006776  013777  001456  172450            MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
1410  007004  005077  172446                    CLR     @GRIVSB
1411
```

```
                                        ;;*****************************************************************
                                        ;*TEST 44       TEST THAT INTERRUPT INPUT BITS SET INPUT READY F_AG
                                        ;;*****************************************************************
007010  000004                  TST44:  SCOPE
007012  032777  002000  172116          BIT     #BIT10,@SWR             ;TEST CABLE SWITCH
007020  001061                          BNE     TST45           ;;BYPASS IF NO I/O CABLE

007022  012737  007036  001110          MOV     #1$,$LPERR              ;LOAD ERROR SCOPE RETURN
007030  012737  000001  001466          MOV     #BIT0,BRLEV3            ;LOAD INTERRUPT BIT
007036  005037  001126          1$:     CLR     $BDDAT                  ;CLEAR BAD DATA
007042  012737  000200  001124          MOV     #BIT7,$GDDAT            ;LOAD GOOD DATA

007050  033737  001466  001446          BIT     BRLEV3,NOTINT           ;TEST IF SET TO INT
007056  001037                          BNE     3$              ;;NO TRY NEXT BIT
007060  005077  172354                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
007064  012777  177777  172344          MOV     #-1,@GRDAI              ;CLEAR INPUT
007072  005077  172336                  CLR     @GRSTAT                 ;CLEAR STATUS
007076  013777  001466  172334          MOV     BRLEV3,@GRDIO           ;LOAD OUTPUT/INPUT
007104  005077  172324                  CLR     @GRSTAT                 ;CLEAR FLAG FROM DATA READY
                                                                        ;SHOULD REMAIN SET VIA DIRECT SET SIDE
007110  105777  172320                  TSTB    @GRSTAT                 ;TEST READY BIT
007114  100401                          BMI     2$              ;;BR IF SET
007116  104010                          ERROR   10                      ;INPUT INTERRUPT BIT FAILED TO SET INPUT READY
                                                                        ;?? DID OPERATOR GIVE CORRECT
                                                                        ;INPUT INTERRUPT BITS ??

007120  043777  001466  172312  2$:     BIC     BRLEV3,@GRDIO           ;CLEAR OUTPUT
007126  053777  001466  172302          BIS     BRLEV3,@GRDAI           ;CLEAR INPUT
007134  005037  001124                  CLR     $GDDAT                  ;CLEAR EXPECTED
007140  005077  172270                  CLR     @GRSTAT                 ;CLEAR STATUS
007144  117737  172264  001126          MOVB    @GRSTAT,$BDDAT          ;READ STATUS
007152  100001                          BPL     3$              ;;BR IF CLEARED
007154  104001                          ERROR   1                       ;INPUT READY FAILED TO CLEAR

007156  006337  001466          3$:     ASL     BRLEV3                  ;CHANGE BIT
007162  001325                          BNE     1$                      ;BR IF NOT DONE
```

```
1448                                    ;;*****************************************************************
1449                                    ;*TEST 45       TEST THAT NON-INTERRUPT INPUT BITS DO NOT SET INPUT READY FLAG
1450                                    ;;*****************************************************************
1451  007164  000004                    TST45:  SCOPE
1452  007166  032777  002000  171742            BIT     #BIT10,@SWR             ;TEST CABLE SWITCH
1453  007174  001042                            BNE     TST46           ;;BYPASS IF NO I/O CABLE
1454
1455  007176  012737  007212  001110            MOV     #1$,$LPERR              ;LOAD ERROR SCOPE RETURN
1456  007204  012737  000001  001466            MOV     #BIT0,BRLEV3            ;LOAD NON-INTERRUPT BIT
1457  007212  012737  000200  001126  1$:       MOV     #200,$BDDAT             ;LOAD BAD DATA
1458  007220  005037  001124                    CLR     $GDDAT                  ;CLEAR GOOD DATA
1459
1460  007224  033737  001466  001446            BIT     BRLEV3,NOTINT           ;TEST IF SET TO INT
1461  007232  001420                            BEQ     3$              ;;NO SKIP AND TRY NEXT BIT
1462  007234  005077  172200                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1463  007240  012777  177777  172170            MOV     #-1,@GRDAI              ;CLEAR INPUT
1464  007246  005077  172162                    CLR     @GRSTAT                 ;CLEAR STATUS
1465  007252  013777  001466  172160            MOV     BRLEV3,@GRDIO           ;LOAD OUTPUT/INPUT
1466  007260  005077  172150                    CLR     @GRSTAT                 ;CLEAR FLAG FROM DATA READY
1467                                                                            ;SHOULD REMAIN SET VIA DIRECT SET SIDE
1468  007264  105777  172144                    TSTB    @GRSTAT                 ;TEST READY BIT
1469  007270  100001                            BPL     3$              ;;BR IF CLEAR
1470  007272  104011                            ERROR   11                      ;INPUT NON-INTERRUPT BIT SET INPUT READY
1471                                                                            ;?? DID OPERATOR GIVE CORRECT
1472                                                                            ;INPUT INTERRUPT BITS ??
1473
1474
1475  007274  006337  001466            3$:     ASL     BRLEV3                  ;CHANGE BIT
1476  007300  001344                            BNE     1$                      ;BR IF NOT DONE
```

```
1477
1478
1479                                    ;;*****************************************************************
1480                                    ;*TEST 46       TEST FOR INT. FROM DRA OUTPUT
1481                                    ;;*****************************************************************
1482  007302  000004                    TST46:  SCOPE
1483  007304  000240                            NOP
1484  007306  000240                            NOP
1485  007310  032777  002000  171620            BIT     #BIT10,@SWR             ;TEST SWITCH
1486  007316  001401                            BEQ     1$
1487  007320  000455                            BR      TST47           ;;
1488  007322  005077  172106           1$:     CLR     @GRSTAT
1489  007326  012777  007416  172120            MOV     #2$,@GRIVB      ;IN CASE OF INTERRUPT
1490  007334  012777  000340  172114            MOV     #340,@GRIVSB
1491  007342  013777  001452  172100            MOV     GRIVSA,@GRIVA
1492  007350  005077  172076                    CLR     @GRIVSA
1493  007354  012777  040000  172052            MOV     #BIT14,@GRSTAT          ;ENABLE OUTPUT INT.
1494  007362  012777  000000  172050            MOV     #0,@GRDIO               ;OUTPUT
1495  007370  017700  172042                    MOV     @GRDAI,R0               ;INPUT
1496  007374  000240                            NOP                             ;LET INTERRUPT OCCUR
1497  007376  000240                            NOP
1498  007400  000240                            NOP
1499  007402  042777  040000  172024            BIC     #BIT14,@GRSTAT
1500  007410  000240                            NOP
1501  007412  104005                            ERROR   5                       ;ERROR, OUTPUT FAILED TO INTERRUPT
1502  007414  000417                            BR      TST47           ;;
1503  007416  022626                    2$:     CMP     (SP)+,(SP)+
1504  007420  013777  001452  172022            MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
1505  007426  005077  172020                    CLR     @GRIVSA
1506  007432  013777  001456  172014            MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
1507  007440  005077  172012                    CLR     @GRIVSB
1508  007444  005037  177776                    CLR     PSW
1509  007450  005077  171760                    CLR     @GRSTAT                 ;CLEAR STATUS
1510
1511                                    ;PRE INTERRUPT SETUP
1512                                    ;;*****************************************************************
1513                                    ;*TEST 47       PRE-INTERRUPT SETUP
1514                                    ;;*****************************************************************
1515  007454  000004                    TST47:  SCOPE
1516  007456  012737  000001  001164            MOV     #1,$TIMES       ;;DO 1 ITERATION
1517  007464  000005                            RESET
1518  007466  042737  177437  001460            BIC     #177437,DIOBRL
1519  007474  001001                            BNE     1$
1520  007476  104007                            ERROR   7                       ;BR LEVEL INDICATED FOR DIGITAL I/O WAS 0
1521  007500  022737  000340  001460    1$:     CMP     #340,DIOBRL
1522  007506  001001                            BNE     2$
1523  007510  104007                            ERROR   7                       ;BR LEVEL INDICATED FOR DIGITAL I/O WAS 7
1524  007512  013737  001460  001462    2$:     MOV     DIOBRL,BRLEV1
1525  007520  162737  000040  001462            SUB     #40,BRLEV1
1526  007526  013737  001460  001464            MOV     DIOBRL,BRLEV2
1527  007534  013737  001460  001466            MOV     DIOBRL,BRLEV3
1528  007542  062737  000040  001466            ADD     #40,BRLEV3
```

```
1529                                    ;;*************************************************************
1530                                    ;*TEST 50       TEST FOR INT. FROM DRA INPUT ON LEVEL INDICATED -1 VIA MAINT. INT
1531                                    ;;*************************************************************
1532  007550  000004                    T5T50:  SCOPE
1533  007552  005077  171656                    CLR     @GRSTAT                 ;CLEAR ENABLES
1534  007556  012777  007612  171664            MOV     #1$,@GRIVA              ;SET UP VECTORS
1535  007564  013737  001462  177776            MOV     BRLEV1,PSW              ;CHANGE PSW
1536  007572  052777  000040  171634            BIS     #BIT5,@GRSTAT           ;GENERATE INPUT MAINT. INTERRUPT
1537  007600  000240                            NOP
1538  007602  000240                            NOP
1539  007604  000240                            NOP
1540  007606  000240                            NOP
1541  007610  104004                            ERROR   4                       ;ERROR, NO INTERRUPT FROM DIGITAL I/O INPUT
1542  007612  022626                    1$:     CMP     (SP)+,(SP)+
1543  007614  013777  001452  171626            MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
1544  007622  005077  171624                    CLR     @GRIVSA
1545  007626  013777  001456  171620            MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
1546  007634  005077  171616                    CLR     @GRIVSB
1547  007640  005037  177776                    CLR     PSW                     ;LOWER PSW
1548                                    ;;*************************************************************
1549                                    ;*TEST 51       TEST FOR NO INT. FROM DRA INPUT ON LEVEL INDICATED  VIA MAINT. INT
1550                                    ;;*************************************************************
1551  007644  000004                    T5T51:  SCOPE
1552  007646  005077  171562                    CLR     @GRSTAT                 ;CLEAR ENABLES
1553  007652  012777  007724  171570            MOV     #1$,@GRIVA              ;SET UP VECTORS
1554  007660  013737  001464  177776            MOV     BRLEV2,PSW              ;CHANGE PSW
1555  007666  052777  000040  171540            BIS     #BIT5,@GRSTAT           ;GENERATE INPUT MAINT. INTERRUPT
1556  007674  000240                            NOP
1557  007676  000240                            NOP
1558  007700  000240                            NOP
1559                                    :SUB-TEST, NOW LOWER THE PSW AND ALLOW THE INTERRUPT
1560  007702  012777  007736  171540            MOV     #2$,@GRIVA              ;RESET VECTOR
1561  007710  005037  177776                    CLR     PSW                     ;LOWER PSW
1562  007714  000240                            NOP
1563  007716  000240                            NOP
1564  007720  000240                            NOP
1565  007722  000403                            BR      3$                      ;ERROR
1566  007724  022626                    1$:     CMP     (SP)+,(SP)+
1567  007726  005037  177776                    CLR     PSW
1568  007732  104006                    3$:     ERROR   6                       ;UNEXPECTED INTERRUPT
1569  007734  000415                            BR      T5T52           ;;
1570  007736  022626                    2$:     CMP     (SP)+,(SP)+             ;POP THE STACK
1571  007740  013777  001452  171502            MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
1572  007746  005077  171500                    CLR     @GRIVSA
1573  007752  013777  001456  171474            MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
1574  007760  005077  171472                    CLR     @GRIVSB
1575  007764  005037  177776                    CLR     PSW
```

```
                                        ;;*****************************************************************
                                        ;*TEST 52        END OF THE PROGRAM
                                        ;;*****************************************************************
010000 000004                           TST52:  SCOPE
010002 012737 000001 001164                     MOV     #1,$TIMES       ;;DO 1 ITERATION
010006 000005                                   RESET
010010 013777 001452 171440                     MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
010016 005077 171436                            CLR     @GRIVSA
010022 013777 001456 171432                     MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
010030 005077 171430                            CLR     @GRIVSB
010034 005037 177776                            CLR     PSW
010040 000137 010164                            JMP     $EOP


                                        ;;*****************************************************************
                                        ;*TEST 53        MISC. EXTERNAL LOGIC TEST
                                        ;;*****************************************************************
010044 000004                           TST53:  SCOPE
010046 012777 000001 171366             EXTTST: MOV     #BIT0,@GRSTAT           ;SET RELAY #1
010054 004737 010066                            JSR     PC,FLIP                 ;TOGGLE THE INPUT-OUTPUT REG
010060 012777 000400 171354                     MOV     #BIT8,@GRSTAT           ;SET RELAY #2
010066 004737 010066                            JSR     PC,FLIP                 ;TOGGLE THE INPUT-OUTPUT REG
010072 000765                                   BR      EXTTST

010074 012737 040000 010162             FLIP:   MOV     #40000,EXTCNT           ;LOAD COUNT
010102                                  1$:
010102 005077 171340                            CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
010106 012777 125252 171332                     MOV     #125252,@GRDIO          ;LOAD OUTPUT
010114 017737 171324 010160                     MOV     @GRDAI,EXTTMP           ;READ INPUT
010122 013777 010160 171314                     MOV     EXTTMP,@GRDAI           ;CLEAR INPUT
010130 005077 171312                            CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
010134 012777 052525 171304                     MOV     #52525,@GRDIO           ;LOAD OUTPUT
010142 017737 171276 010160                     MOV     @GRDAI,EXTTMP           ;READ INPUT
010150 013777 010160 171266                     MOV     EXTTMP,@GRDAI           ;CLEAR INPUT
010156 005337 010162                            DEC     EXTCNT                  ;FINISHED COUNT
010162 001347                                   BNE     1$
010164 000207                                   RTS     PC                      ;EXIT

010166 000000                           EXTTMP: 0
010170 000000                           EXTCNT: 0
```

```
                                .SBTTL  END OF PASS ROUTINE

                                ;;*****************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                ;*TYPE "END PASS"
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO FULNUL
                                ;*IF IT IS DESIRED TO HAVE A BELL INDICATE THE "END OF PASS" LOCATION
                                ;*SENDMG CAN BE CHANGED TO 7.

010164                          $EOP:
010164  000004                          SCOPE
010166  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
010172  005037  001164                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
010176  005237  001202                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
010202  042737  100000  001202          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
010210  005327                          DEC     (PC)+           ;;LOOP?
010212  000001                  $EOPCT: .WORD   1
010214  003015                          BGT     $DOAGN          ;;YES
010216  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
010220  000001                  $ENDCT: .WORD   1
010222  010212                          $EOPCT
010224  104400  010257                  TYPE    ,$ENDMG         ;;TYPE "END PASS"
010230  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
010234  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
010236  000005                          RESET                   ;;CLEAR THE WORLD
010240  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
010242  000240                          NOP                     ;;SAVE ROOM
010244  000240                          NOP                     ;;FOR
010246  000240                          NOP                     ;;ACT11
010250                          $DOAGN:
010250  000137                          JMP     @(PC)+          ;;RETURN
010252  010272                  $RTNAD: .WORD   FULNUL
010254     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
010257     015  042412  042116  $ENDMG: .ASCIZ  <15><12>"END PASS"
010264  050040  051501  000123
010272  005037  010310          FULNUL: CLR     10$
010276  005337  010310          1$:     DEC     10$
010302  001375                          BNE     1$
010304  000137  002622                  JMP     IOTEST
010310  000000                  10$:    0
```

```
 1661
 1662  010312  052123  052101  051525  EM1:    .ASCIZ  /STATUS REGISTER IN ERROR/
 1663  010320  051040  043505  051511
 1664  010326  042524  020122  047111
 1665  010334  042440  051122  051117
 1666  010342     000
 1667  010343     111  050116  052125  EM2:    .ASCIZ  /INPUT REGISTER IN ERROR/
 1668  010350  051040  043505  051511
 1669  010356  042524  020122  047111
 1670  010364  042440  051122  051117
 1671  010372     000
 1672  010373     117  052125  052520  EM3:    .ASCIZ  /OUTPUT REGISTER IN ERROR/
 1673  010400  020124  042522  044507
 1674  010406  052123  051105  044440
 1675  010414  020116  051105  047522
 1676  010422  000122
 1677  010424  047111  052520  020124  EM4:    .ASCIZ  /INPUT FAILED TO INTERRUPT/
 1678  010432  040506  046111  042105
 1679  010440  052040  020117  047111
 1680  010446  042524  051122  050125
 1681  010454  000124
 1682  010456  052517  050124  052125  EM5:    .ASCIZ  /OUTPUT FAILED TO INTERRUPT/
 1683  010464  043040  044501  042514
 1684  010472  020104  047524  044440
 1685  010500  052116  051105  052522
 1686  010506  052120     000
 1687  010511     125  042516  050130  EM6:    .ASCIZ  /UNEXPECTED INTERRUPT/
 1688  010516  041505  042524  020104
 1689  010524  047111  042524  051122
 1690  010532  050125  000124
 1691  010536  050117  051105  052101  EM7:    .ASCIZ  /OPERATOR INTERVENTION ERROR/
 1692  010544  051117  044440  052116
 1693  010552  051105  042526  052116
 1694  010560  047511  020116  051105
 1695  010566  047522  000122
 1696  010572  047111  042524  051122  EM10:   .ASCIZ  /INTERRUPT INPUT BIT FAILED TO SET INPUT READY FLAG/
 1697  010600  050125  020124  047111
 1698  010606  052520  020124  044502
 1699  010614  020124  040506  046111
 1700  010622  042105  052040  020117
 1701  010630  042523  020124  047111
 1702  010636  052520  020124  042522
 1703  010644  042101  020131  046106
 1704  010652  043501     000
 1705  010655     116  047117  044455  EM11:   .ASCIZ  /NON-INTERRUPT INPUT BIT SET INPUT READY FLAG/
 1706  010662  052116  051105  052522
 1707  010670  052120  044440  050116
 1708  010676  052125  041040  052111
 1709  010704  051440  052105  044440
 1710  010712  050116  052125  051040
 1711  010720  043505  054504  043040
 1712  010726  040514  000107
```

```
 1713
 1714  010732  051105  050122  020103  DH1:    .ASCIZ  /ERRPC   STATUS  EXPECTED/
 1715  010740  020040  052123  052101
 1716  010746  051525  020040  054105
 1717  010754  042520  052103  042105
 1718  010762     000
 1719  010763     105  051122  041520  DH2:    .ASCIZ  /ERRPC   INPUT   EXPECTED/
 1720  010770  020040  044440  050116
 1721  010776  052125  020040  042440
 1722  011004  050130  041505  042524
 1723  011012  000104
 1724  011014  051105  050122  020103  DH3:    .ASCIZ  /ERRPC   OUTPUT  EXPECTED/
 1725  011022  020040  052517  050124
 1726  011030  052125  020040  054105
 1727  011036  042520  052103  042105
 1728  011044     000
 1729  011045     105  051122  041520  DH4:    .ASCIZ  /ERRPC/
 1730  011052     000
 1731  011053     105  051122  041520  DH10:   .ASCIZ  /ERRPC   STATUS  EXPECT  INPUT BIT/
 1732  011060  020040  051440  040524
 1733  011066  052524  020123  042440
 1734  011074  050130  041505  020124
 1735  011102  044440  050116  052125
 1736  011110  041040  052111     000
 1737          011116                  .EVEN
 1738  011116  001116  001126  001124  DT1:    $ERRPC,$BDDAT,$GDDAT,0
 1739  011124  000000
 1740  011126  001116  000000          DT4:    $ERRPC,0
 1741  011132  001116  001126  001124  DT10:   $ERRPC,$BDDAT,$GDDAT,BRLEV3,0
 1742  011140  001466  000000
 1743  011144     000     000     000  DF1:    .BYTE   0,0,0,0
 1744  011147     000
 1745  011150     000     000     000  DF10:   .BYTE   0,0,0,0,0,0
 1746  011153     000     000     000
 1747  011156  005015  053523  020122  MSGSWR: .ASCIZ  <15><12>/SWR = /
 1748  011164  020075     000
 1749          011170                  .EVEN
 1750                                  .SBTTL  GETSWR SUBROUTINE
 1751
 1752  011170  022737  000176  001136  GETSWR: CMP     #SWREG,SWR              ;TEST IF REAL SWR
 1753  011176  001415                  BEQ     1$                      ;BR IF NOT
 1754  011200  104400  011206          TYPE    ,65$                ;;TYPE ASCIZ STRING
 1755  011204  000410                  BR      64$                 ;;GET OVER THE ASCIZ
 1756                                  ;;65$:  .ASCIZ  <15><12>/DEPRESS CONT./
 1757  011226                          64$:
 1758  011226  000000                  HALT                            ;WAIT FOR OPERATOR
 1759  011230  000207                  RTS     PC                      ;EXIT
 1760  011232  104400                  1$:     TYPE
 1761  011234  011156                  MSGSWR
 1762  011236  104407                  RDOCT
 1763  011240  012677  167672          MOV     (SP)+,@SWR              ;SAVE VALUE TYPED
 1764  011244  000207                  RTS     PC                      ;EXIT
```

```
                                        .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

011246                                  $SCOPE:
011246  104404                                  CKSWR
011250  032777  040000  167660          1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
011256  001114                                  BNE     $OVER           ;;YES IF SW14=1
                                        ;*****START OF CODE FOR THE XOR TESTER*****
011260  000416                          $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
011262  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
011266  012737  011306  000004                  MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
011274  005737  177060                          TST     @#177060        ;;TIME OUT ON XOR?
011300  012637  000004                          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
011304  000463                                  BR      $SVLAD          ;;GO TO THE NEXT TEST
011306  022626                          5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
011310  012637  000004                          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
011314  000423                                  BR      7$              ;;LOOP ON THE PRESENT TEST
011316                                  6$::*****END OF CODE FOR THE XOR TESTER*****
011316  032777  000400  167612                  BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
011324  001404                                  BEQ     2$              ;;BR IF NO
011326  127737  167604  001102                  CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
011334  001465                                  BEQ     $OVER           ;;BR IF YES
011336  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
011342  001421                                  BEQ     3$              ;;BR IF NO
011344  123737  001115  001103                  CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
011352  101015                                  BHI     3$              ;;BR IF NO
011354  032777  001000  167554                  BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
011362  001404                                  BEQ     4$              ;;BR IF NO
011364  013737  001110  001106          7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
011372  000446                                  BR      $OVER
011374  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
011400  005037  001164                          CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
011404  000415                                  BR      1$              ;;ESCAPE TO THE NEXT TEST
011406  032777  004000  167522          3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
011414  001011                                  BNE     1$              ;;BR IF YES
011416  005737  001202                          TST     $PASS           ;;IF FIRST PASS OF PROGRAM
011422  001406                                  BEQ     1$              ;;      INHIBIT ITERATIONS
011424  005237  001104                          INC     $ICNT           ;;INCREMENT ITERATION COUNT
011430  023737  001164  001104                  CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
011436  002024                                  BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
011440  012737  000001  001104          1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
011446  013737  011524  001164                  MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
```

```
 1821  011454  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
 1822  011460  113737  001102  001200          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
 1823  011466  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 1824  011472  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 1825  011476  005037  001166                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 1826  011502  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 1827  011510  013777  001102  167422  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 1828  011516  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 1829  011522  000002                          RTI                     ;;FIXES PS
 1830  011524  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
 1831
 1832                                  .SBTTL  ERROR HANDLER ROUTINE
 1833
 1834                                  ;;**************************************************************
 1835                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 1836                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 1837                                  ;*AND GO TO $ERRTYP ON ERROR
 1838                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 1839                                  ;*SW15=1         HALT ON ERROR
 1840                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
 1841                                  ;*SW09=1         LOOP ON ERROR
 1842                                  ;*CALL
 1843                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
 1844
 1845  011526                          $ERROR:
 1846  011526  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
 1847  011532  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
 1848  011534  013777  001102  167376          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
 1849  011542  005237  001112                  INC     $ERTTL          ;;INC THE ERROR COUNT
 1850  011546  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
 1851  011552  162737  000002  001116          SUB     #2,$ERRPC
 1852  011560  117737  167332  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
 1853  011566  032777  020000  167342          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
 1854  011574  001004                          BNE     20$             ;;SKIP TYPEOUTS
 1855  011576  004737  012502                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
 1856  011602  104400  001171                  TYPE    ,$CRLF
 1857  011606                          20$:
 1858  011606  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 1859  011614  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
 1860  011616  113737  001114  011630          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
 1861  011624  004737  013604                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
 1862  011630     000          21$:    .BYTE   0
 1863  011631     000                  .BYTE   0
 1864  011632  000777          22$:    BR      22$             ;;APT ERROR LOOP
 1865  011634  005777  167276          2$:     TST     @SWR            ;;HALT ON ERROR
 1866  011640  100001                          BPL     3$              ;;SKIP IF CONTINUE
 1867  011642  000000                          HALT                    ;;HALT ON ERROR!
 1868  011644  032777  001000  167264  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
 1869  011652  001402                          BEQ     4$              ;;BR IF NO
 1870  011654  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
 1871  011660  005737  001166          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
 1872  011664  001402                          BEQ     5$              ;;BR IF NONE
 1873  011666  013716  001166                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
 1874  011672                          5$:
 1875  011672  022737  012040  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
 1876  011700  001001                          BNE     6$              ;;BRANCH IF NO
```

```
                                        HALT                    ;;YES
                                6$:
                                        RTI                     ;;RETURN

                                .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                ;*CHANGE IT TO BINARY.
                                ;*CALL:
                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

011706  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
011710  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
011716  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011720  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
011722  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
011724  104406                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
011726  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
011730  005001                          CLR     R1              ;;CLEAR DATA WORD
011732  005002                          CLR     R2
011734  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
011736  001412                          BEQ     3$              ;;IF ZERO GET OUT
011740  006301                          ASL     R1              ;;*2
011742  006102                          ROL     R2
011744  006301                          ASL     R1              ;;*4
011746  006102                          ROL     R2
011750  006301                          ASL     R1              ;;*8
011752  006102                          ROL     R2
011754  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
011760  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
011762  000764                          BR      2$              ;;LOOP
011764  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
011766  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
011772  010237  012006                  MOV     R2,$HIOCT
011776  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
012000  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
012002  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
012004  000002                          RTI                     ;;RETURN
012006  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE

                                .SBTTL  TTY INPUT ROUTINE

                                ;;**********************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                ;*WHEN OPERATING IN TTY FLAG MODE.
012010  022737  000176  001136  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
012016  001073                          BNE     14$             ;;BRANCH IF NO
012020  105777  167116                  TSTB    @$TKS           ;;CHAR THERE?
012024  100070                          BPL     14$             ;;IF NO, DON'T WAIT AROUND
012026  117746  167112          2$:     MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
012032  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
```

```
1933  012036  022726  000007            CMP     #7,(SP)+        ;;IS IT A CONTROL G?
1934  012042  001061                    BNE     14$             ;;NO, RETURN TO USER
1935  012044  104403  012453            TYPE    .$CNTLG         ;;YES, ECHO CONTROL G
1936
1937  012050  104400  012460    6$:     TYPE    .$MSWR          ;;TYPE CURRENT CONTENTS
1938  012054  013746  000176            MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
1939  012060  104401                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1940  012062  104400  012471            TYPE    .$MNEW          ;;PROMPT FOR NEW SWR
1941  012066  005046                    CLR     -(SP)           ;;CLEAR COUNTER
1942  012070  005046                    CLR     -(SP)           ;;THE NEW SWR
1943  012072  104405            7$:     RDCHR                   ;;GET NEXT CHAR
1944
1945  012074  022716  000025    8$:     CMP     #25,(SP)        ;;IS IT A CONTROL U?
1946  012100  001005                    BNE     9$              ;;BRANCH IF NO
1947  012102  104400  012446            TYPE    .$CNTLU         ;;YES, ECHO IT
1948  012106  062706  000006            ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
1949  012112  000756                    BR      6$              ;;LET'S TRY IT AGAIN
1950
1951  012114  022716  000015    9$:     CMP     #15,(SP)        ;;IS IT A <CR>?
1952  012120  001011                    BNE     11$             ;;BRANCH IF NO
1953  012122  005766  000004            TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
1954  012126  001403                    BEQ     10$             ;;BRANCH IF YES
1955  012130  016677  000002  167000    MOV     2(SP),@SWR      ;;SAVE NEW SWR
1956  012136  062706  000006    10$:    ADD     #6,SP           ;;CLEAR UP STACK
1957  012142  000417                    BR      13$             ;;RETURN TO USER
1958  012144  022716  000012    11$:    CMP     #12,(SP)        ;;IS IT A <LF>
1959  012150  001017                    BNE     15$             ;;BRANCH IF NO
1960  012152  005766  000004            TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
1961  012156  001403                    BEQ     12$             ;;YES
1962  012160  016677  000002  166750    MOV     2(SP),@SWR      ;;SAVE NEW SWR
1963  012166  062706  000006    12$:    ADD     #6,SP           ;;CLEAR UP STACK
1964  012172  013716  000046            MOV     @#46,(SP)       ;;GET RESTART
1965  012176  062716  000010            ADD     #10,(SP)        ;;ADDRESS
1966  012202  104400  001171    13$:    TYPE    .$CRLF          ;;ECHO <CR> AND <LF>
1967  012206  000002    14$:            RTI                     ;;RETURN
1968  012210  004737  013516    15$:    JSR     PC,$TYPEC       ;;ECHO CHAR
1969  012214  042726  177770            BIC     #177770,(SP)+   ;;RESTRICT TO 0-7
1970  012220  005766  000002            TST     2(SP)           ;;IS THIS THE FIRST CHAR
1971  012224  001403                    BEQ     16$             ;;BRANCH IF YES
1972  012226  006316                    ASL     (SP)            ;;NO, SHIFT PRESENT
1973  012230  006316                    ASL     (SP)            ;;    CHAR OVER TO MAKE
1974  012232  006316                    ASL     (SP)            ;;    ROOM FOR NEW ONE.
1975  012234  005266  000002    16$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
1976  012240  056616  177776            BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
1977  012244  000712                    BR      7$              ;;GET THE NEXT ONE
1978                            ;;**************************************************************
1979                            ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
1980                            ;*CALL:
1981                            ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
1982                            ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
1983                            ;*                              ;;WITH PARITY BIT STRIPPED OFF
1984                            ;
1985
1986  012246  011646            $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
1987  012250  016666  000004  000002    MOV     4(SP),2(SP)     ;;SAVE THE PS
1988  012256  105777  166660    1$:     TSTB    @$TKS           ;;WAIT FOR
```

```
 1989  012262  100375                          BPL     1$                      ;;A CHARACTER
 1990  012264  117766  166654  000004          MOVB    @$TKB,4(SP)             ;;READ THE TTY
 1991  012272  042766  177600  000004          BIC     #^C(177),4(SP)          ;;GET RID OF JUNK IF ANY
 1992  012300  026627  000004  000140          CMP     4(SP),#140              ;;IS IT UPPER CASE?
 1993  012306  002407                          BLT     2$                      ;;BRANCH IF YES
 1994  012310  026627  000004  000175          CMP     4(SP),#175              ;;IS IT A SPECIAL CHAR?
 1995  012316  003003                          BGT     2$                      ;;BRANCH IF YES
 1996  012320  042766  000040  000004          BIC     #40,4(SP)               ;;MAKE IT UPPER CASE
 1997  012326  000002                  2$:     RTI                             ;;GO BACK TO USER
 1998                                          ;;**************************************************************
 1999                                          ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 2000                                          ;*CALL:
 2001                                          ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 2002                                          ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 2003                                          ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 2004
 2005  012330  010346                  $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
 2006  012332  012703  012436          1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
 2007  012336  022703  012446          2$:     CMP     #$TTYIN+8.,R3           ;;BUFFER FULL?
 2008  012342  101405                          BLOS    4$                      ;;BR IF YES
 2009  012344  104405                          RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
 2010  012346  112613                          MOVB    (SP)+,(R3)              ;;GET CHARACTER
 2011  012350  122713  000177          10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
 2012  012354  001003                          BNE     3$                      ;;SKIP IF NOT
 2013  012356  104400  001170          4$:     TYPE    ,$QUES                  ;;TYPE A '?'
 2014  012362  000763                          BR      1$                      ;;CLEAR THE BUFFER AND LOOP
 2015  012364  111337  012434          3$:     MOVB    (R3),9$                 ;;ECHO THE CHARACTER
 2016  012370  104400  012434                  TYPE    ,9$
 2017  012374  122723  000015                  CMPB    #15,(R3)+               ;;CHECK FOR RETURN
 2018  012400  001356                          BNE     2$                      ;;LOOP IF NOT RETURN
 2019  012402  105063  177777                  CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
 2020  012406  104400  001172                  TYPE    ,$LF                    ;;TYPE A LINE FEED
 2021  012412  012603                          MOV     (SP)+,R3                ;;RESTORE R3
 2022  012414  011646                          MOV     (SP),-(SP)              ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 2023  012416  016666  000004  000002          MOV     4(SP),2(SP)             ;;      FIRST ASCII CHARACTER ON IT
 2024  012424  012766  012436  000004          MOV     #$TTYIN,4(SP)
 2025  012432  000002                          RTI                             ;;RETURN
 2026  012434     000                  9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
 2027  012435     000                          .BYTE   0                       ;;TERMINATOR
 2028  012436  000010                  $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
 2029  012446  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>            ;;CONTROL "U"
 2030  012453     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>            ;;CONTROL "G"
 2031  012460  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
 2032  012466  020075     000
 2033  012471     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
 2034  012476  036440  000040
```

```
                                .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.

012502                          $ERRTYP:
012502  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012506  010046                          MOV     R0,-(SP)        ;;SAVE R0
012510  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
012512  153700  001114                  BISB    @#$ITEMB,R0
012516  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                ;;TYPE THE PC OF THE ERROR
012520  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;ERROR ADDRESS
012524  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012526  000426                          BR      6$              ;;GET OUT
012530  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
012532  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
012534  006300                          ASL     R0
012536  006300                          ASL     R0
012540  062700  001320                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
012544  012037  012554                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
012550  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
012552  104400                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
012554  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
012556  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012562  012037  012572          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
012566  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
012570  104400                          TYPE                    ;;TYPE THE "DATA HEADER"
012572  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
012574  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012600  011000                  5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
012602  001004                          BNE     7$              ;;GO TYPE THE DATA
012604  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
012606  104400  001171                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
012612  000207                          RTS     PC              ;;RETURN
012614                          7$:
012614  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
012616  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
012620  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
012622  001770                          BEQ     6$              ;;BR IF NO
012624  104400  012632                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
012630  000771                          BR      7$              ;;LOOP
012632  020040  000             8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
012636                                  .EVEN
```

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

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

012636  017646  000000  $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
012642  116637  000001  013061 MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
012650  112637  013063         MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
012654  062716  000002         ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
012660  000406                 BR      $TYPON
012662  112737  000001  013061 $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
012670  112737  000006  013063         MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
012676  112737  000005  013060 $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
012704  010346                 MOV     R3,-(SP)        ;;SAVE R3
012706  010446                 MOV     R4,-(SP)        ;;SAVE R4
012710  010546                 MOV     R5,-(SP)        ;;SAVE R5
012712  113704  013063         MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
012716  005404                 NEG     R4
012720  062704  000006         ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
012724  110437  013062         MOVB    R4,$OMODE       ;;SAVE IT FOR USE
012730  113704  013061         MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
012734  016605  000012         MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
012740  005003                 CLR     R3              ;;CLEAR THE OUTPUT WORD
012742  006105         1$:     ROL     R5              ;;ROTATE MSB INTO "C"
012744  000404                 BR      3$              ;;GO DO MSB
012746  006105         2$:     ROL     R5              ;;FORM THIS DIGIT
012750  006105                 ROL     R5
012752  006105                 ROL     R5
012754  010503                 MOV     R5,R3
012756  006103         3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
012760  105337  013062         DECB    $OMODE          ;;TYPE THIS DIGIT?
012764  100016                 BPL     7$              ;;BR IF NO
012766  042703  177770         BIC     #177770,R3      ;;GET RID OF JUNK
012772  001002                 BNE     4$              ;;TEST FOR 0
```

```
2140  012774  005704                          TST     R4              ;;SUPPRESS THIS 0?
2141  012776  001403                          BEQ     5$              ;;BR IF YES
2142  013000  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2143  013002  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2144  013006  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2145  013012  110337  013056                  MOVB    R3,8$           ;;SAVE FOR TYPING
2146  013016  104400  013056                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
2147  013022  105337  013060          7$:     DECB    $OCNT           ;;COUNT BY 1
2148  013026  003347                          BGT     2$              ;;BR IF MORE TO DO
2149  013030  002402                          BLT     6$              ;;BR IF DONE
2150  013032  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2151  013034  000744                          BR      2$              ;;GO DO THE LAST DIGIT
2152  013036  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
2153  013040  012604                          MOV     (SP)+,R4        ;;RESTORE R4
2154  013042  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2155  013044  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2156  013052  012616                          MOV     (SP)+,(SP)
2157  013054  000002                          RTI                     ;;RETURN
2158  013056     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2159  013057     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2160  013060     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2161  013061     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2162  013062  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;**************************************************************
                                        ;POWER DOWN ROUTINE
013064  012737  013230  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
013072  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
013100  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
013102  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
013104  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
013106  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
013110  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
013112  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
013114  017746  166016                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
013120  010637  013234                  MOV     SP,$SAVR6       ;;SAVE SP
013124  012737  013136  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
013132  000000                          HALT
013134  000776                          BR      .-2             ;;HANG UP

                                        ;;**************************************************************
                                        ;POWER UP ROUTINE
013136  012737  013230  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
013144  013706  013234                  MOV     $SAVR6,SP       ;;GET SP
013150  005037  013234                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
013154  005237  013234          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
013160  001375                          BNE     1$              ;;OF  WORD
013162  012677  165750                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
013166  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
013170  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
013172  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
013174  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
013176  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
013200  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
013202  012737  013064  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
013210  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
013216  104400                          TYPE                    ;;REPORT THE POWER FAILURE
013220  013236                  $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
013222  012716                          MOV     (PC)+,(SP)      ;;RESTART AT IOTEST
013224  002622                  $PWRAD: .WORD   IOTEST          ;;RESTART ADDRESS
013226  000002                          RTI
013230  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
013232  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
013234  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
013236  005015  042522  052123  PWRMSG: .ASCIZ  <15><12>/RESTARTING AFTER A POWER FAILURE.<15><12>
013244  051101  044524  043516
013252  040440  052106  051105
013260  040440  050040  053517
013266  051105  043040  044501
013274  052514  042522  005015
013302     000
                013304                  .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
                                        ;*

013304  105737  001155          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
013310  100002                          BPL     1$              ;;BR IF YES
013312  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
013314  000430                          BR      3$              ;;LEAVE
013316  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
013320  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
013324  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
013332  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
013334  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
013342  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
013344  010037  013354                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
013350  004737  013574                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
013354  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
013356  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
013364  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
013366  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
013370  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
013372  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
013374  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
013376  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
013402  000002                          RTI                     ;;RETURN
013404  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
013410  001430                          BEQ     8$
013412  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
013416  001006                          BNE     5$
013420  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
013422  104400                          TYPE                    ;;TYPE A CR AND LF
013424  001171                          $CRLF
013426  105037  013562                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
013432  000755                          BR      2$              ;;GET NEXT CHARACTER
013434  004737  013516          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
013440  123726  001154          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
013444  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
013446  013746  001152                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
013452  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
013456  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
```

```
013460  004737  013516                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
013464  105337  013562                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
013470  000770                          BR      7$              ;;LOOP

                                :HORIZONTAL TAB PROCESSOR

013472  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
013476  004737  013516          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
013502  132737  000007  013562          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
013510  001372                          BNE     9$              ;;TAB STOP
013512  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
013514  000724                          BR      2$              ;;GET NEXT CHARACTER
013516  105777  165424          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
013522  100375                          BPL     $TYPEC
013524  116677  000002  165416          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
013532  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
013540  001003                          BNE     1$              ;;BRANCH IF NO
013542  105037  013562                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
013546  000406                          BR      $TYPEX          ;;EXIT
013550  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
013556  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
013560  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
013562  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
013564  000207                  $TYPEX: RTS     PC
```

```
                                        .SBTTL  APT COMMUNICATIONS ROUTINE
                                        ;;**********************************************************************
013566  112737  000001  014033  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
013574  112737  000001  014030  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
013602  000403                        BR      $ATYC
013604  112737  000001  014032  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
013612                          $ATYC:
013612  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
013614  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
013616  105737  014030                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
013622  001450                        BEQ     5$              ;;IF NOT:  BR
013624  122737  000001  001214        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
013632  001031                        BNE     3$              ;;IF NOT:  BR
013634  132737  000100  001215        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
013642  001425                        BEQ     3$              ;;IF NOT:  BR
013644  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
013650  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
013656  005737  001174          1$:   TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
013662  001375                        BNE     1$              ;;IF NOT:  WAIT
013664  013037  001210                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
013670  105720                  2$:   TSTB    (R0)+           ;;FIND END OF MESSAGE
013672  001376                        BNE     2$
013674  163700  001210                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
013700  006200                        ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
013702  010037  001212                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
013706  012737  000004  001174        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
013714  000413                        BR      5$
013716  017637  000004  013742  3$:   MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
013724  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
013732  013746  177776                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
013736  004737  013304                JSR     PC,$TYPE        ;;CALL TYPE MACRO
013742  000000                  4$:   .WORD   0
013744                          5$:
013744  105737  014032          10$:  TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
013750  001416                        BEQ     12$             ;;IF NOT:  BR
013752  005737  001214                TST     $ENV            ;;RUNNING UNDER APT?
013756  001413                        BEQ     12$             ;;IF NOT:  BR
013760  005737  001174          11$:  TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
013764  001375                        BNE     11$             ;;IF NOT:  WAIT
013766  017637  000004  001176        MOV     @4(SP),$FATAL   ;;GET ERROR #
013774  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
014002  005237  001174                INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
014006  105037  014032          12$:  CLRB    $FFLG           ;;CLEAR FATAL FLAG
014012  105037  014031                CLRB    $LFLG           ;;CLEAR LOG FLAG
014016  105037  014030                CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
014022  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
014024  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
014026  000207                        RTS     PC              ;;RETURN
014030  000             $MFLG:  .BYTE   0               ;;MESSG. FLAG
014031  000             $LFLG:  .BYTE   0               ;;LOG FLAG
014032  000             $FFLG:  .BYTE   0               ;;FATAL FLAG
                                .EVEN
014034
000200                  APTSIZE=200
```

```
                    000001                  APTENV=001
                    000100                  APTSPOOL=100
                    000040                  APTCSUP=040


                                            .SBTTL  TRAP DECODER

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

014034  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
014036  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
014042  005740                          TST     -(R0)           ;;BACKUP BY 2
014044  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
014046  006300                          ASL     R0              ;;POSITION FOR INDEXING
014050  016000  014056                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
014054  000200                          RTS     R0              ;;GO TO ROUTINE


                                            .SBTTL  TRAP TABLE

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

                                            ;       ROUTINE
                                            ;       -------
014056                          $TRPAD:
014056  013304                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
014060  012662                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
014062  012636                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
014064  012676                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
014066  012010                          $CKSWR  ;;CALL=CKSWR    TRAP+4(104404)  TEST FOR CHANGE IN SOFT-SWR
014070  012246                          $RDCHR  ;;CALL=RDCHR    TRAP+5(104405)  TTY TYPEIN CHARACTER ROUTINE
014072  012330                          $RDLIN  ;;CALL=RDLIN    TRAP+6(104406)  TTY TYPEIN STRING ROUTINE
014074  011706                          $RDOCT  ;;CALL=RDOCT    TRAP+7(104407)  READ AN OCTAL NUMBER FROM TTY
                    000001                  .END
```

```
ABASE = 170410            299*    415     459
ADDW1 = 000000            415     461
ADDW2 = 000000            415     462
ADPUOP= 000000            415     430
ADDW0 = 000000            415     463
ADDW1 = 000000            415     464
ADDW10= 000000            415     473
ADDW11= 000000            415     474
ADDW12= 000000            415     475
ADDW13= 000000            415     476
ADDW14= 000000            415     477
ADDW15= 000000            415     478
ADDW2 = 000000            415     465
ADDW3 = 000000            415     466
ADDW4 = 000000            415     467
ADDW5 = 000000            415     468
ADDW6 = 000000            415     469
ADDW7 = 000000            415     470
ADDW8 = 000000            415     471
ADDW9 = 000000            415     472
ADEVCT= 000000            415     421
ADEVM = 000006            415     460
AENV  = 000000            415     426
AENVM = 000000            415     427
AFATAL= 000000            415     418
AMADR1= 000000            415     444
AMADR2= 000000            415     447
AMADR3= 000000            415     450
AMADR4= 000000            415     453
AMAMS1= 000000            415     437
AMAMS2= 000000            415     445
AMAMS3= 000000            415     448
AMAMS4= 000000            415     451
AMSGAD= 000000            415     423
AMSGLG= 000000            415     424
AMSGTY= 000000            415     417
AMTYP1= 000000            415     438
AMTYP2= 000000            415     446
AMTYP3= 000000            415     449
AMTYP4= 000000            415     452
APASS = 000000            415     420
APRIOR= 000200            300*    415     456
APTCSU= 000040           2247    2354*
APTENV= 000001           1858    2240    2310    2352*
APTSIZ= 000200            619    2351*
APTSPO= 000100           2242    2312    2353*
ASWREG= 000000            415     428
ATESTN= 000000            415     419
AUNIT = 000000            415     422
A_SWR = 000000            415     429
AVECT1= 000350            299*    415     454
AVECT2= 000000            415     455
BEGIN   001470            326     576*
BEGIN1  001474            327     578*
BITO  = 000001            283*    750     766     859     860     932    1004    1039    1064    1420    1456    1535
BITOO = 000001            273*    283
```

```
BIT01 = 000002           272*   282
BIT02 = 000004           271*   281
BIT03 = 000010           270*   280
BIT04 = 000020           269*   279
BIT05 = 000040           268*   278
BIT06 = 000100           267*   277
BIT07 = 000200           266*   276
BIT08 = 000400           265*   275   1797
BIT09 = 001000           264*   274   1805   1868
BIT1  = 000002           282*
BIT10 = 002000           263*   944   1386   1416   1452   1485
BIT11 = 004000           262*  1812
BIT12 = 010000           261*
BIT13 = 020000           260*  1362   1853
BIT14 = 040000           259*   894    895    932   1493   1499   1783
BIT15 = 100000           258*   882    883    932   1251
BIT2  = 000004           281*
BIT3  = 000010           280*
BIT4  = 000020           279*
BIT5  = 000040           278*  1302   1332   1536   1555
BIT6  = 000100           277*   918    919    932   1331   1341   1394   1400
BIT7  = 000200           276*   906    907    932   1236   1422
BIT8  = 000400           275*   871    872    932   1597
BIT9  = 001000           274*
BPTVEC= 000014           290*
BRLEV1  001462           569*  1524*  1525*  1535
BRLEV2  001464           570*  1004*  1007   1025   1030*  1526*  1554
BRLEV3  001466           571*  1005*  1006*  1013   1023   1064*  1066   1075   1084*  1420*  1424   1429   1439
                        1439   1446*  1456*  1460   1465   1475*  1527*  1529*  1741
CKSWR = 104404          1782   2386*
CR    = 000015           198*  2286   2296
CRLF  = 000200           199*  2257   2296
DCISP = 177570           205*   392    605
DFI     011144           504    510    516    522    528    534    540   1743*
DFIO    011150           546   1745*
DHI     010732           502   1714*
DHIO    011053           544    550   1731*
DHO     010763           508   1719*
DHS     011014           514   1724*
DHT     011045           520    526    532    538   1729*
DICBRL  001460           568*   671*  1518*  1521   1524   1526   1527
DISPLA  001140           392*   606*   612*  1827*  1848*
DISPRE  000174           322*   612
DONOT   001754           623*
DRADD   001426           553*   623*   657    658    660    662
DRBRL   001432           555*   625*   671
DRIV    001430           554*   624*   664    665    667    669
DRT21   006156           946   1263*
DSWR  = 177570           204*   391    605
DTI     011116           503    509    515   1738*
DTIO    011132           545    551   1741*
DTT     011126           521    527    533    539   1740*
EMTVEC= 000030           293*   589*   590*
EMI     010312           501   1662*
EMIO    010572           543   1696*
EMII    010655           549   1705*
```

```
EM2     010343            507    1667#
EM3     010373            513    1672#
EM4     010424            519    1677#
EM5     010456            525    1682#
EM6     010511            531    1687#
EM7     010536            537    1691#
ERRVEC= 000004            286#    603     604#    615#   1788    1789#   1791#   1794#
EXTCNT  010162           1601#   1611#   1616#
EXTTMP  010160           1605#   1606    1609#   1610   1615#
EXTTST  010040            328    1595#   1599
FLIP    010066           1596    1598    1601#
FULNUL  010272           1652    1656#
GETSWR  011170            643     650     656    1752#
GNS     = ****** G        321     633     637     641    647     654    1756    2382    2383    2384    2395    2396    2387
                         2388    2389
GRBMIC  001442            561#    662#    663#    846#   847
GRDAI   001436            559#    658#    659#    683#   858#    949#    952    962#    965    976#    979    990#    993
                         1012    1022    1045#   1049   1070#   1079    1097#  1101   1102#   1106   1107#   1111   1112#
                         1116    1117#   1121    1122#  1126    1127#   1131   1132#  1136    1137#  1141    1142#  1146
                         1147#   1151    1152#   1169#  1173    1174#   1178   1179#  1183    1184#  1188    1199#  1193
                         1194#   1198    1199#   1203   1204#   1208    1209#  1213   1214#   1218   1219#   1223   1224#
                         1255    1277    1396    1427#  1439#   1463#   1495   1605   1606#   1609   1610#
GRDIC   001440            560#    660#    661#    684#   691#    692     703#   705    714#    715    725#    726    738#
                          739     752#    753#    754    768#    769#    770    784#   785#    786#   787#    788#   789#
                          790#    791#    792#    793#   794#    797     807#   808#   809#    810#   911#    812#   813#
                          814#    815#    816#    817#   819     831#    932    957#   948#    951#   961#    964#   975#
                          978#    989#    992#   1011#  1021#   1044#   1047#  1048#  1069#   1076#  1077#   1096#  1099#
                         1100#   1104#   1105#   1109#  1110#   1114#   1115#  1119#  1120#   1124#  1125#   1129#  1130#
                         1134#   1135#   1139#   1140#  1144#   1145#   1149#  1150#  1168#   1171#  1172#   1176#  1177#
                         1181#   1182#   1186#   1187#  1191#   1192#   1196#  1197#  1201#   1202#  1206#   1207#  1211#
                         1212#   1216#   1217#   1221#  1222#   1238#   1253#  1276#  1395#   1426#  1429#   1438#  1462#
                         1465#   1494#   1603#   1604#  1607#   1608#
GRIVA   001450            564#    664#    674#   1272#  1298#   1309#   1327#  1336#  1358#   1390#  1407#   1491#  1504#
                         1534#   1543#   1553#   1560#  1571#   1593#
GRIVB   001454            566#    667#    669#    676#  1274#   1300#   1329#  1360#  1369#   1392#  1409#   1489#  1506#
                         1545#   1573#   1585#
GRIVSA  001452            565#    665#    666#    674    675#    1273#   1299#  1328#  1359    1359#  1391#   1407   1408#
                         1491    1492#   1504    1505#  1543    1544#   1571   1572#  1583    1584#
GRIVSB  001456            567#    669#    670#    676    677#    1275#   1300   1301#  1329    1330#  1361#   1392   1393#
                         1409    1410#   1490#   1506   1507#   1545    1546#  1573   1574#   1585   1586#
GRSTAT  001434            558#    657#    682#    860#   861     872#    873    883#   884     893#   895#    896    907#
                          908     917#    919#    920    930#    932#    934   1237#  1240    1252#  1256    1270#  1282#
                         1297#   1302#   1326#   1331#  1332#   1340    1357#  1362#  1389#   1394#  1400#   1406#  1429#
                         1430#   1432    1441#   1442   1464#   1466#   1468   1498#  1493#   1499#  1509#   1533#  1536#
                         1552#   1555#   1595#   1597#
HT      = 000011          196#   2255    2296
ICTEST  002622            672#   1659    2202
ICTVEC= 000020            291#    587#    588#
LF      = 000012          197#   2290    2296
MSGSWR  011156           1747#   1761
NCTINT  001446            563#    651#   1424    1460
NOTLCH  001444            562#    644#   1005    1008   1041    1066    1072   1094   1166
PC      =%000007          217#    643#    650#    656#  1596#   1598#   1613#  1636#  1639#   1646#  1651    1759#  1764#
                         1855#   1861#   1968#   2074#  2201    2245#   2264#  2271#  2279#   2292#  2294#   2329#  2346#
PIRQ    = 177772          203#
PIRQVE= 000240            297#
```

```
PR0    = 000000          220#
PR1    = 000040          221#
PR2    = 000100          222#
PR3    = 000140          223#
PR4    = 000200          224#
PR5    = 000240          225#
PR6    = 000300          226#
PR7    = 000340          227#
PS     = 177776          200#   201
PSW    = 177776          201#  1271*  1311*  1318*  1338*  1348*  1371*  1377*  1405*  1508*  1535*  1547*  1554*
                        1561*  1567*  1575*  1587*
PWRMSG  013236          2200   2207#
PWRVEC= 000024           292#   593*   594*  2169*  2170*  2179*  2185*  2197*  2198*
RBEG    001500           577    579#
RDCHR = 104405          1943   2009   2387#
RDLIN = 104406          1896   2388#
RDOCT = 104407          1762   2389#
RESVEC= 000010           287#
R0     =%000000          208#   576*   578*   626  1095*  1099  1103*  1104  1108*  1109  1113*  1114  1118*
                        1119   1123*  1124  1128*  1129  1133*  1134  1138*  1139  1143*  1144  1148*  1149
                        1153*  1167*  1171  1175*  1176  1180*  1181  1185*  1186  1190*  1191  1195*  1196
                        1200*  1201  1205*  1206  1210*  1211  1215*  1216  1220*  1221  1225*  1255*  1277*
                        1396*  1495*  1643*  1646  1893  1897*  1900  1916*  2046  2047*  2048*  2055*  2056*
                        2057*  2058*  2059*  2060  2065  2070*  2072*  2076  2078  2171  2196*  2238  2239*
                        2244   2249  2252*  2306  2314*  2318  2319  2321*  2322*  2323  2345*  2365  2366*
                        2367   2368*  2369*  2370*  2371*
R1     =%000001          209#  1101*  1102  1106*  1107  1111*  1112  1116*  1117  1121*  1122  1126*  1127
                        1131*  1132  1136*  1137  1141*  1142  1146*  1147  1151*  1152  1155  1173*  1174
                        1178*  1179  1183*  1184  1188*  1189  1193*  1194  1198*  1199  1203*  1204  1208*
                        1209   1213*  1214  1218*  1219  1223*  1224  1227  1894  1898*  1902*  1904*  1905*
                        1909*  1912  1915*  2172  2195*  2307  2344*
R2     =%000002          210#  1895  1899*  1903*  1905*  1907*  1913  1914*  2173  2194*
R3     =%000003          211#  2005  2006*  2007  2010*  2011  2015  2017  2019*  2021*  2119  2129*  2134*
                        2135*  2138*  2143*  2144*  2145  2154*  2174  2193*
R4     =%000004          212#  2120  2122*  2123*  2124*  2125  2126*  2140  2142*  2150*  2153*  2175  2192*
R5     =%000005          213#  2121  2127*  2129*  2131*  2132*  2133*  2134  2152*  2176  2191*
R6     =%000006          214#   216   581*   582*   583
R7     =%000007          215#   217
SP     =%000006          216#   585*   603*   613*   615   673*  1310  1316  1337  1346  1370  1375  1404
                        1503   1542  1566  1570  1763  1788*  1791  1793  1794  1823  1824  1828*  1950
                        1870*  1873*  1891*  1892*  1893*  1894*  1895*  1897  1900*  1908*  1909  1911  1912*
                        1914   1915  1916  1931*  1932*  1933  1938*  1941*  1942*  1945  1948*  1951  1953
                        1955   1956*  1958  1960  1962  1963*  1964*  1965*  1969*  1970  1972*  1973*  1974*
                        1975*  1976*  1986*  1987*  1990*  1991*  1992  1994  1996*  2005*  2010  2021  2022*
                        2023*  2024*  2046*  2051*  2072  2076*  2111*  2112  2113  2114*  2119*  2120*  2121*
                        2127   2152  2153  2154  2155*  2156*  2171*  2172*  2173*  2174*  2175*  2176*  2177*
                        2178   2186*  2190  2191  2192  2193  2194  2195  2196  2201*  2238*  2239  2249*
                        2251   2252  2253*  2255  2257  2259  2265  2267*  2269*  2277*  2281  2285  2296
                        2290   2306*  2307*  2314  2315*  2326  2327*  2328*  2338  2339*  2344  2345  2355*
                        2366
STACK = 001100          191#   585   673
STKLMT= 177774          202#
SWR     001136          391#   605*  607  611*  621*  644  651  944  1386  1416  1452  1485  1752
                        1763*  1783  1797  1799  1805  1812  1853  1865  1868  1927  1955*  1962*  2177
                        2190*
SWREG   000176          323#   611  1752  1927  1938
```

```
SW0    = 000001            255*
SW00   = 000001            245*    255
SW01   = 000002            244*    254
SW02   = 000004            243*    253
SW03   = 000010            242*    252
SW04   = 000020            241*    251
SW05   = 000040            240*    250
SW06   = 000100            239*    249
SW07   = 000200            238*    248
SW08   = 000400            237*    247
SW09   = 001000            236*    246
SW1    = 000002            254*
SW10   = 002000            235*
SW11   = 004000            234*
SW12   = 010000            233*
SW13   = 020000            232*
SW14   = 040000            231*
SW15   = 100000            230*
SW2    = 000004            253*
SW3    = 000010            252*
SW4    = 000020            251*
SW5    = 000040            250*
SW6    = 000100            249*
SW7    = 000200            248*
SW8    = 000400            247*
SW9    = 001000            246*
TBITVE= 000014             288*
TKVEC  = 000060            295*
TPVEC  = 000064            296*
TRAPVE= 000034             294*    591*    592*
TRTVEC= 000014             289*
TST1    002654             681*
TST10   003216             764*
TST11   003302             782*
TST12   003414             799     805*
TST13   003524             821     827*
TST14   003602             842*
TST15   003660             856*
TST16   003730             863     870*
TST17   003766             875     881*
TST2    002672             689*
TST20   004024             886     892*
TST21   004066             898     905*
TST22   004124             910     916*
TST23   004166             922     928*
TST24   004230             935     943*
TST25   004312             954     960*
TST26   004362             967     974*
TST27   004432             981     988*
TST3    002730             694     700*
TST30   004502             995    1001*
TST31   004652            1037*
TST32   004752            1062*
TST33   005100            1091*
TST34   005460            1157    1163*
TST35   006040            1229    1235*
```

```
TST36   006110            1242    1250*
TST37   006156            1257    1267*
TST4    002766             706     712*
TST40   006270            1283    1294*
TST41   006376            1314    1323*
TST42   006532            1342    1344    1355*
TST43   006636            1374    1383*
TST44   007010            1388    1403    1415*
TST45   007164            1417    1451*
TST46   007302            1453    1482*
TST47   007454            1487    1502    1515*
TST5    003024             717     723*
TST50   007550            1532*
TST51   007644            1551*
TST52   007770            1569    1580*
TST53   010036            1594*
TST6    003062             728     734*
TST7    003140             748*
TYPE  = 104400             631     635     639     645     652    1642    1754    1760    1856    1935    1937    1940    1947
                          1966    2013    2016    2020    2045    2062    2064    2067    2069    2073    2080    2146    2199
                          2260    2382*
TYPOC = 104401            1939    2053    2077    2383*
TYPON = 104403            2385*
TYPOS = 104402            2384*
$APTHD  001000             352     358*
$ARG1   001732             617*
$ASTAT= ****** U          2332    2347
$ATYC   013612            2303    2305*
$ATY1   013566            2301*
$ATY3   013574            2245    2302*
$ATY4   013604            1861    2304*
$BASE   001250             459*    623
$BDADR  001122             385*
$BDDAT  001126             387*    583     692*    693     705*    715*    716     726*    727     739*    740     754*    755
                           770*    771*    772     797*    798     819*    820     932*    933     847*    848     861*    962
                           873*    874     884*    885     896*    897     908*    909     920*    921     934*    952*    953
                           965*    966     979*    980     993*    994    1012*   1013*   1014    1022*   1023*   1025    1049*
                          1050    1079*   1080    1155*   1156    1227*   1228    1240*   1241    1256*   1340*   1341    1421*
                          1442*   1457*   1738    1741
$CDW1   001254             461*
$CDW2   001256             462*
$CHARC  013562            2262*   2272*   2279    2288*   2293*
$CKSWR  012010            1927*   2386
$CMTAG  001100             373*    580     581     589     595     596     597
$CM1  = 000002             403*    404*    405*
$CM2  = 000004             403*    404*    405*
$CM3  = 000002             401*    403
$CNTLG  012453            1935    2030*
$CNTLU  012446            1947    2029*
$CPUOP  001222             430*
$CRLF   001171             408*   1856    1880    1966    2029    2045    2064    2069    2073    2261    2296
$DDW0   001260             463*
$DDW1   001262             464*
$DDW10  001304             473*
$DDW11  001306             474*
$DDW12  001310             475*
```

```
$DDW13  001312            476#
$DDW14  001314            477#
$DDW15  001316            478#
$DDW2   001264            465#
$DDW3   001266            466#
$DDW4   001270            467#
$DDW5   001272            468#
$DDW6   001274            469#
$DDW7   001276            470#
$DDW8   001300            471#
$DDW9   001302            472#
$DEVCT  001204            421#
$DEVM   001252            460#
$DOAGN  010250           1638    1644    1650#
$ENDAD  010240            337    1646#   1875
$ENDCT  010220            595    1640#
$ENDMG  010257           1642    1654#
$ENULL  010254           1653#
$ENV    001214            426#   1858    2240    2310    2334
$ENVM   001215            427#    619    2242    2247    2312
$EOP    010164           1588    1630#
$EOPCT  010212            595*   1637#   1641
$ERFLG  001103            376#   1772    1801    1803    1809*   1831    1846*   1880
$ERMAX  001115            382#    598*   1803    1826*   1831
$ERROR  011526            589    1845#
$ERRPC  001116            383#   1738    1740    1741    1850*   1851*   1852    1980    2051
$ERRTB  001320            498#   2059
$ERRTY  012502           1855    2044#
$ERTTL  001112            380#   1849*   1880
$ESCAP  001166            406#    597*   1825*   1871    1873    1880
$ETABL  001214            425#
$ETEND  001320            364     481#
$FATAL  001176            418#   2338*
$FFLG   014032           2301*   2304*   2332    2341*   2349#
$FILLC  001154            399#   2265    2296
$FILLS  001153            398#   2296
$GOADR  001120            384#
$GODAT  001124            386#    690*    693     702*    713*    716     724*    727     737*    738     740     743*    750*
                          753     755     759*    766*    769     772     776*    783*    784     798     806*    907     820
                          830*    831     833     836*    845*    846     848     951*    859*    862     871*    874     982*
                          885     894*    897     906*    909     918*    921     931*    950*    953     963*    966     977*
                          980     991*    994    1007*   1008    1011    1014    1021    1024*   1039*   1041    1047    1050
                         1056*   1071*   1072*   1075*   1076    1080    1093*   1094*   1095    1156    1165*   1166*   1167
                         1228    1236*   1241    1251*   1339*   1422*   1440*   1459*   1738    1741
$GET42  010230           1643#
$HD   = 000000            187
$HIBTS  001000            359#
$HIOCT  012006           1913*   1918#
$ICNT   001104            377#   1816*   1817    1819*   1830
$ILLUP  013230           2169    2185    2204#
$ITEMB  001114            381#   1852*   1860    1880    2048
$LF     001172            409#   1880    2020    2029    2296
$LFLG   014031           2342*   2348#
$LPADR  001106            378#    599*   1807*   1823*   1828    1830
$LPERR  001110            379#    600*    736*    749*    765*    829*    844*    1002*   1039*   1063*   1419*   1455*   1807
                         1824*   1830    1870
```

```
$MADR1  001226            443#
$MADR2  001232            447#
$MADR3  001236            450#
$MADR4  001242            453#
$MAIL   001174            360     364     416#    617     1822    1858    2240
$MAMS1  001224            437#
$MAMS2  001230            445#
$MAMS3  001234            448#
$MAMS4  001240            451#
$MBADR  001002            360#
$MFLG   014030           2302#    2308    2343#   2347#
$MNEW   012471           1940     2033#
$MSGAD  001210            423#    2318#   2321
$MSGLG  001212            424#    2323#
$MSGTY  001174            417#    2316    2324#   2336    2340#
$MSWR   012460           1937     2031#
$MTYP1  001225            438#
$MTYP2  001231            446#
$MTYP3  001235            449#
$MTYP4  001241            452#
$MXCNT  011524           1820     1830#
$NULL   001152            397#    2267    2296
$NWTST= 000001            678#     686#    697#    709#    720#    731#    745#    761#    779#    802#    824#    839#    853#
                          867#     878#    889#    902#    913#    925#    940#    957#    971#    985#    998#   1034#   1059#
                         1088#    1160#   1232#   1247#   1264#   1291#   1320#   1352#   1380#   1412#   1448#   1479#   1512#
                         1529#    1548#   1577#   1591#
$OCNT   013060           2118#    2147#   2160#
$OMODE  013062           2113#    2117#   2122    2125#   2135#   2162#
$OVER   011510           1784     1800    1808    1818    1827#
$PASS   001202            420#     618#   1634#   1635#   1653    1814    1831
$PASTM  001006            362#
$PRIOR  001246            456#     625
$PWRAD  013224           2202#
$PWRDN  013064            593     2169#   2197
$PWRMG  013220           2200#
$PWRUP  013136           2179     2185#
$QUES   001170            407#    1880    2013    2029    2296
$RDCHR  012246           1986#    2387
$RDDEC= ****** U         2390
$RDLIN  012330           2005#    2388
$RDOCT  011706           1891#    2389
$RDSZ = 000010           1998#
$REGAD  001156            401#
$REGO   001160            403#
$REGI   001162            404#
$RTNAD  010252           1652#
$R2A  = ****** U         2390
$SAVRE= ****** U         2390
$SAVR6  013234           2178#    2186    2187#   2188#   2206#
$SCOPE  011246            587     1781#
$SETUP= 000037            536#     586     587     589     591     593     595     596     597     599     616    1632    1782
                         1846     1868    1875
$STUP = 177777            536#
$SVLAD  011454           1792     1821#
$SVPC = 000214            335#     340
$SWR  = 165400            177#     187     306     307     308     309     310     311     312     405     406     407     596
```

```
                    597     599     600     682     690     701     713     724     735     749     765     783     805
                    828     843     857     871     882     893     906     917     929     944     961     975     989
                   1002    1038    1063    1092    1164    1236    1251    1268    1295    1324    1356    1384    1416
                   1452    1483    1516    1533    1552    1581    1595    1625    1633    1645    1651    1653    1773
                   1774    1775    1776    1777    1783    1795    1797    1798    1801    1802    1803    1810    1811
                   1812    1824    1827    1830    1838    1839    1840    1841    1849    1853    1865    1868    1880
                   2203
$SWREG  001216      428*    621
$SWRMK= 000000      312     313    1777    1778    1799
$TESTN  001200      419*   1822*
$TIMES  001164      405*    596*    701*    735*    828*    843*    929*   1268*   1516*   1581*   1633*   1910*   1917
                   1820*   1830
$TKB    001144      394*   1923    1931    1990
$TKS    001142      393*   1923    1929    1988
$TN   = 000054      177*    187     678     682*    686     690*    694     697     701*    706     709     713*    717
                    720     724*    728     731     735*    745     749*    761     765*    779     783*    799     802
                    806*    821     824     828*    839     843*    853     857*    863     867     871*    875     879
                    882*    886     889     893*    898     902     906*    910     913     917*    922     925     929*
                    935     940     944*    954     957     961*    967     971     975*    991     985     989*    995
                    998    1002*   1034    1038*   1059    1063*   1088    1092*   1157    1160    1164*   1229    1232
                   1236*   1242    1247    1251*   1257    1264    1268*   1283    1291    1295*   1314    1320    1324*
                   1342    1344    1352    1356*   1374    1380    1384*   1388    1403    1412    1416*   1417    1448
                   1452*   1453    1479    1483*   1487    1502    1512    1516*   1529    1533*   1548    1552*   1569
                   1577    1581*   1591    1595*
$TPB    001150      396*   2285*   2296
$TPFLG  001155      400*   2234    2296
$TPS    001146      395*   2283    2296
$TRAP   014034      591    2365*
$TRP  = 000010     2373*   2383*   2384*   2385*   2386*   2387*   2388*   2389*   2390*
$TRPAD  014056     2370    2381*
$TSTM   001004      361*
$TSTNM  001102      375*   1632*   1772    1799    1821*   1822    1827    1831    1848    1880
$TTYIN  012436     2006    2007    2024    2028*
$TYPBN= ****** U   2386
$TYPDS= ****** U   2386
$TYPE   013304     2234*   2329    2373    2382
$TYPEC  013516     1968    2264    2271    2278    2283*   2284
$TYPEX  013564     2289    2291    2294*
$TYPOC  012662     2116*   2383
$TYPON  012676     2115    2118*   2385
$TYPOS  012636     2111*   2384
$UNIT   001206      422*
$UNITM  001010      363*
$USWR   001220      429*
$VECT1  001244      454*    624
$VECT2  001245      455*
$XTSTR  011260     1786*
$$GET4= 000000     1645*
$OFILL  013061     2112*   2116*   2126    2161*
$40CAT= ****** U   1783    1855
.     = 014076      317*    321*    335     336*    338*    340*    341*    348     349*    351*    353*    372*    410
                    584     599     600     634*    638*    655*   1653    1656    1737*   1749*   1830    1831    1880
                   1923    2028*   2029    2035    2083*   2181    2205    2214*   2296    2350*
.$ASTA= ****** U   2302    2305
.$X   = 001000      348*    353
```

```
CLERIN    940*   949   962   976   990  1045  1070  1097  1169  1427  1463
CLEROT    939*   947   961   975   989  1044  1048  1069  1077  1096  1100  1105  1110  1115  1120
         1125  1130  1135  1140  1145  1150  1168  1172  1177  1182  1187  1192  1197  1202  1207
         1212  1217  1222  1426  1462  1602  1607
CLRVCT    573*   674  1407  1504  1543  1571  1583
COMMEN    298*
ENDCOM    298*
ERROR     192*   695   707   718   729   742   757   774   800   822   835   850   864   876   887
          899   911   923   936   955   968   982   996  1016  1027  1052  1082  1158  1230  1243
         1258  1285  1288  1304  1317  1334  1343  1347  1364  1376  1402  1434  1444  1470  1501
         1520  1523  1541  1568
ESCAPE    299*
GETPRI    299*
MULT      298*
NEWTST    298*   678   686   697   709   720   731   745   761   779   802   824   839   853   867
          878   889   902   913   925   940   957   971   985   998  1034  1059  1088  1160  1232
         1247  1263  1291  1320  1352  1380  1412  1448  1479  1512  1529  1548  1577  1591
POP       298*  1914  2190  2191  2344  2345
PUSH      298*  1893  2171  2177  2305  2307  2328
REPORT    298*
SCOPE     193*   681   689   700   712   723   734   748   764   782   805   827   842   856   870
          881   892   905   916   928   943   960   974   988  1001  1037  1062  1091  1163  1235
         1250  1267  1294  1323  1355  1383  1415  1451  1482  1515  1532  1551  1580  1594  1631
SETPRI    298*
SETTRA   2373*  2383  2384  2385  2386  2387  2388  2389
SETUP     298*   590
SKIP      298*   694   706   717   728   741   799   821   863   875   886   898   910   922   935
          954   967   981   995  1015  1026  1051  1081  1157  1229  1242  1257  1283  1314  1342
         1344  1374  1388  1403  1417  1425  1433  1443  1453  1461  1469  1487  1502  1569
SLASH     298*
SPACE     298*
STARS     298*   333   345   347   354   368   410   414   573   575   678   680   686   688   697
          699   709   711   720   722   731   733   745   747   761   763   779   781   802   804
          824   826   839   841   853   855   867   869   878   880   889   891   902   904   913
          915   925   927   940   942   957   959   971   973   985   987   998  1000  1034  1036
         1059  1061  1089  1090  1160  1162  1232  1234  1247  1249  1264  1266  1291  1293  1320
         1322  1352  1354  1380  1382  1412  1414  1448  1450  1479  1481  1512  1514  1529  1531
         1548  1550  1577  1579  1591  1593  1621  1769  1834  1883  1922  1978  1998  2039  2088
         2167  2183  2219  2300  2359
SWRSU     298*   601*
TRMTRP   2373*
TYPBIN    298*
TYPDEC    298*
TYPNAM    298*
TYPNUM    298*
TYPOCS    298*
TYPOCT    298*  1938  2051  2075
TYPTXT    298*   631   635   639   645   652  1754
$$CMRE    365*   403   404
$$CMTM    365*
$$ESCA    298*
$$NEWT    298*   678   686   697   709   720   731   745   761   779   802   824   839   853   867
          878   889   902   913   925   940   957   971   985   998  1034  1059  1088  1160  1232
         1247  1264  1291  1320  1352  1380  1412  1448  1479  1512  1529  1548  1577  1591
$$SET    2373*  2383  2384  2385  2386  2387  2388  2389
$$SETM    617*   618
```

B06


```
$$SKIP     298#    694    706    717    728    793    821    863    875    886    898    910    922    935    954
           967     981    995   1157   1229   1242   1257   1283   1314   1342   1344   1374   1388   1403   1417
          1453    1487   1502   1569
.EQUAT     177#    187
.HEADE     177#
.SETUP     177#    536
.SWRHI     177#    301
.SWRLO     313#
.$ACT1     177#    330
.$APTB     411#
.$APTH     177#    343
.$APTY     177#   2297
.$CATC     177#    314
.$CMTA     177#    365
.$EOP      177#   1618
.$ERRO     177#   1931
.$ERRT     177#   2036
.$PARM     177#
.$POWE     177#   2164
.$RDOC     177#   1980
.$READ     177#   1919
.$SAVE     177#
.$SCOP     177#   1766
.$SPAC     177#
.$SWDO     177#
.$TRAP     177#   2356
.$TYPE     177#   2216
.$TYPO     177#   2085
```

ADD	659	661	663	666	668	670	1528	1909	1948	1956	1963	1965	2059	2114	2124
	2253	2315	2327	2339											
ASL	1059	1076	1030	1056	1084	1446	1475	1902	1904	1906	1972	1973	1974	2056	2057
	2058	2369													
ASR	2322														
BEQ	608	627	694	706	717	728	741	756	773	799	821	834	849	910	922
	935	945	954	967	981	995	1009	1015	1026	1051	1081	1157	1229	1242	1342
	1387	1461	1486	1644	1753	1798	1800	1802	1806	1815	1847	1869	1872	1901	1954
	1961	1971	2061	2066	2079	2141	2243	2256	2291	2309	2313	2333	2335		
BG	1918														
BGE	1638	1995	2148												
BGT	1804														
BHI	1009	1013	1021	1023	1072	1075	1094	1166	1400	1438	1499	1518	1635	1908	1932
	1969	1991	1996	2138											
BIC	1103	1102	1107	1112	1117	1122	1127	1132	1137	1142	1147	1152	1174	1179	1184
	1189	1194	1199	1204	1209	1214	1219	1224	1302	1332	1362	1439	1536	1535	1976
	2143	2144													
BICB	2048														
BIS	882	874	885	897	944	1008	1025	1041	1066	1341	1386	1415	1424	1452	1460
	1495	1783	1797	1905	1912	1853	1868								
BISB	2119	2242	2247	2279	2312										
BIT	2008														
BITB	1993	2149	2270												
BLK	1257	1433													
BNE	588	629	630	744	760	777	837	852	863	875	886	898	1031	1042	1057
	1067	1085	1417	1425	1447	1453	1476	1519	1522	1612	1658	1784	1813	1954	1959
	1976	1928	1934	1946	1952	1959	2012	2018	2049	2071	2139	2189	2241	2248	2250
	2258	2266	2280	2287	2311	2317	2320	2337							
BPL	1443	1469	1866	1930	1989	2137	2235	2294							
BR	577	632	636	640	646	653	1283	1314	1344	1374	1388	1403	1487	1502	1565
	1569	1599	1755	1786	1792	1795	1808	1811	1964	1910	1949	1957	1977	2014	2054
	2081	2115	2130	2151	2181	2205	2237	2263	2273	2282	2289	2303	2325		
CLR	576	582	596	597	619	675	677	682	693	684	702	737	752	930	845
	857	893	917	930	931	948	950	961	975	989	1024	1044	1048	1069	1077
	1096	1100	1105	1110	1115	1120	1125	1130	1135	1140	1145	1150	1168	1172	1177
	1182	1187	1192	1197	1202	1207	1212	1217	1222	1237	1252	1270	1271	1282	1297
	1311	1318	1326	1338	1339	1348	1357	1371	1377	1389	1393	1405	1406	1408	1410
	1421	1426	1428	1430	1440	1441	1458	1462	1464	1466	1488	1492	1505	1507	1508
	1509	1533	1544	1546	1547	1552	1561	1567	1572	1574	1575	1584	1586	1587	1603
	1607	1632	1633	1656	1810	1825	1898	1899	1941	1942	2047	2128	2187		
CLRB	1809	2019	2262	2288	2341	2342	2343								
CMP	583	607	693	716	727	740	755	772	798	820	909	921	953	966	980
	994	1014	1050	1080	1156	1228	1239	1241	1254	1310	1316	1337	1346	1370	1375
	1404	1503	1521	1542	1566	1570	1752	1793	1817	1875	1927	1933	1945	1951	1959
	1982	1994	2007												
CMPB	833	848	1799	1803	1858	2011	2017	2240	2255	2257	2265	2286	2290	2310	
COM	771	795	786	787	788	789	790	791	792	793	794	808	809	910	911
	812	813	814	815	816	817	1006	1103	1108	1113	1118	1123	1128	1133	1139
	1143	1148	1153	1175	1180	1185	1190	1195	1200	1205	1210	1215	1220	1235	
DEC	1611	1636	1657	2055											
DECB	2136	2147	2269	2272											
EMT	192														
HALT	321	1758	1867	1877	2180	2204	2236								
INC	743	1634	1816	1849	1975	2142	2150	2188	2340						
INCB	836	851	1821	1846	2292										
IOT	193														

```
CMP     326   327   328   628   946  1588  1651  1659
CMPB    643   550   656  1596  1598  1646  1955  1861  1968  2245  2264  2271  2278  2329
MOV     578   581   585   587   588   589   590   591   592   593   594   595   599   600   603
        604   605   606   611   612   613   615   621   623   624   625   644   651   657   658
        660   662   664   665   667   669   671   673   674   676   690   691   692   731   732
        755   713   714   715   724   725   726   735   736   738   739   749   750   754   755
        766   768   770   783   784   795   806   807   819   828   829   832   843   944   958
        959   860   861   871   872   873   882   883   884   894   895   896   906   907   909
        918   919   920   929   932   934   949   951   952   962   963   964   965   976   977
        978   979   990   991   992   993  1002  1004  1005  1007  1011  1012  1022  1038  1039
       1045  1047  1049  1063  1064  1070  1071  1076  1079  1093  1095  1097  1099  1101  1104
       1106  1109  1111  1114  1116  1119  1121  1124  1126  1129  1131  1134  1136  1139  1141
       1144  1146  1149  1151  1155  1165  1167  1169  1171  1173  1176  1178  1181  1183  1186
       1188  1191  1193  1196  1198  1201  1203  1206  1208  1211  1213  1216  1218  1221  1223
       1227  1236  1238  1240  1251  1253  1255  1256  1268  1272  1273  1274  1275  1276  1277
       1298  1299  1300  1301  1309  1327  1328  1329  1330  1331  1336  1340  1358  1359  1360
       1361  1369  1390  1391  1392  1394  1395  1396  1407  1409  1419  1420  1422  1427  1429
       1455  1456  1457  1463  1465  1489  1490  1491  1493  1494  1495  1504  1506  1516  1524
       1526  1527  1534  1535  1543  1545  1553  1554  1560  1571  1573  1591  1583  1585  1595
       1597  1601  1604  1605  1606  1608  1609  1610  1639  1643  1763  1798  1789  1791  1794
       1807  1819  1820  1823  1824  1827  1828  1848  1850  1970  1873  1991  1892  1893  1894
       1895  1997  1912  1913  1914  1915  1916  1938  1955  1962  1964  1986  1987  2005  2006
       2021  2022  2023  2024  2046  2051  2060  2065  2070  2072  2076  2111  2119  2120  2121
       2127  2134  2152  2153  2154  2155  2156  2169  2170  2171  2172  2173  2174  2175  2176
       2177  2178  2179  2185  2186  2190  2191  2192  2193  2194  2195  2196  2197  2198  2201
       2238  2239  2244  2252  2267  2306  2307  2314  2318  2323  2324  2326  2328  2338  2344
       2345  2365  2366  2370
MOVB    598   931   946   847  1442  1822  1825  1852  1960  1900  1931  1990  2010  2015  2112
       2113  2116  2117  2118  2122  2125  2126  2145  2249  2277  2285  2301  2302  2304  2368
NEG    2123
NOP     795   796   818  1373  1374  1278  1279  1280  1281  1295  1296  1303  1312  1313  1324
       1325  1333  1356  1363  1372  1373  1384  1385  1397  1398  1399  1401  1483  1484  1496
       1497  1498  1500  1537  1538  1539  1540  1556  1557  1558  1562  1563  1564  1647  1648
       1649
RESET   579   672   704   933  1269  1517  1582  1645
ROL    1903  1905  1907  2129  2131  2132  2133  2135
RTI     614  1286  1289  1829  1879  1917  1967  1997  2025  2157  2203  2254
RTS    1613  1759  1764  2074  2294  2346  2371
SUB    1525  1851  2321
TRAP   2373  2383  2384  2385  2386  2387  2388  2389
TST     610   626   629  1790  1814  1865  1871  1911  1953  1960  1970  2078  2140  2251  2259
       2281  2316  2334  2336  2367
TSTB   1432  1468  1801  1929  1988  2234  2283  2308  2319  2332
.ASCII  407   408
.ASCIZ  409   634   638   642   648   655  1654  1662  1667  1672  1677  1682  1687  1691  1696
       1705  1714  1719  1724  1729  1731  1747  1757  2029  2030  2031  2033  2082  2207
.BLKB  2028
.BYTE   375   376   381   382   397   398   399   400   426   427   437   439   445   446   449
        449   451   452   454   455   456   457  1653  1743  1745  1962  1863  2026  2027  2159
       2159  2160  2161  2347  2348  2349
.ENABL  177
.END   2390
.ENDC   182   192   284   298   309   311   312   313   327   334   339   340   346   349   355
        369   373   375   401   405   406   407   411   415   437   445   448   451   454   455
        456   457   460   461   462   463   464   465   466   467   468   469   470   471   472
        473   474   475   476   477   478   479   483   536   574   576   585   586   589   591
```

```
                593   595   596   597   599   601   623   634   638   642   648   655   679   680   681
                692   695   698   699   700   701   702   707   710   711   712
                713   718   721   722   723   724   729   732   733   734   735   736   742   746   747
                748   749   762   763   764   765   780   781   782   783   800   803   804   805   806
                822   825   826   827   828   829   840   841   842   843   844   854   855   856   857
                864   868   869   870   871   876   879   880   881   882   887   890   891   892   893
                899   903   904   905   906   911   914   915   916   917   923   926   927   928   929
                930   936   941   942   943   944   955   958   959   960   961   968   972   973   974
                975   982   986   987   988   989   996   999  1000  1001  1002  1016  1027  1035  1036
               1037  1038  1052  1060  1061  1062  1063  1082  1089  1090  1091  1092  1158  1161  1162
               1163  1164  1230  1233  1234  1235  1236  1243  1248  1249  1250  1251  1258  1265  1266
               1267  1268  1269  1284  1292  1293  1294  1295  1315  1321  1322  1323  1324  1343  1345
               1353  1354  1355  1356  1375  1381  1382  1383  1384  1389  1404  1413  1414  1415  1416
               1418  1426  1434  1444  1449  1450  1451  1452  1454  1462  1470  1480  1481  1482  1483
               1488  1503  1513  1514  1515  1516  1517  1530  1531  1532  1533  1549  1550  1551  1552
               1570  1578  1579  1580  1581  1582  1592  1593  1594  1595  1622  1624  1625  1627  1629
               1632  1638  1641  1642  1643  1645  1651  1653  1656  1757  1770  1773  1778  1783  1785
               1796  1799  1800  1801  1803  1805  1812  1816  1821  1823  1827  1830  1831  1835  1838
               1846  1850  1855  1856  1857  1865  1875  1879  1880  1884  1886  1919  1923  1979  1998
               1999  2006  2008  2011  2013  2029  2040  2055  2084  2089  2168  2177  2178  2194  2190
               2191  2201  2203  2207  2220  2249  2301  2302  2305  2332  2347  2360  2366  2369  2382
               2383  2384  2385  2386  2387  2388  2389  2390
.EQUIV          192   193   201   216   217   246   247   248   249   250   251   252   253   254   255
                274   275   276   277   278   279   280   281   282   283
.EVEN           415   458   634   638   642   648   655  1737  1749  1757  2083  2214  2350
.IF             178   190   256   284   309   310   311   312   313   324   333   336   338   345   347
                354   368   372   374   401   405   406   407   410   411   414   437   445   448   451
                454   455   456   457   460   461   462   463   464   465   466   467   468   469   470
                471   472   473   474   475   476   477   478   479   483   536   573   575   580   585
                587   589   591   593   595   596   597   599   617   633   637   641   647   654   679
                680   682   686   688   690   694   697   699   701   702   706   709   711   713   717
                720   722   724   728   731   733   735   736   741   745   747   749   761   763   765
                779   781   783   799   802   804   806   821   824   926   828   829   839   941   843
                844   853   855   857   863   867   869   871   875   878   880   882   886   889   891
                893   898   902   904   906   910   913   915   917   922   925   927   929   930   935
                940   942   944   954   957   959   961   967   971   973   975   981   985   987   989
                995   998  1000  1002  1015  1026  1034  1036  1038  1051  1059  1061  1063  1081  1098
               1090  1092  1157  1160  1162  1164  1229  1232  1234  1236  1242  1247  1249  1251  1257
               1264  1266  1268  1269  1283  1291  1293  1295  1314  1320  1322  1324  1342  1344  1352
               1354  1356  1374  1380  1382  1384  1388  1403  1412  1414  1416  1417  1425  1433  1443
               1448  1450  1452  1453  1461  1469  1479  1481  1483  1487  1502  1512  1514  1516  1517
               1529  1531  1533  1548  1550  1552  1569  1577  1579  1581  1582  1591  1593  1595  1621
               1622  1623  1624  1625  1626  1627  1631  1637  1640  1642  1643  1645  1651  1653  1654
               1656  1756  1769  1772  1777  1782  1783  1795  1797  1798  1799  1801  1802  1803  1812
               1814  1822  1824  1829  1830  1831  1834  1837  1846  1849  1853  1855  1856  1858  1865
               1868  1875  1879  1880  1883  1886  1898  1922  1923  1978  1998  2006  2007  2011  2012
               2028  2029  2039  2054  2070  2088  2167  2177  2178  2183  2190  2191  2199  2201  2203
               2207  2219  2240  2300  2302  2305  2332  2347  2359  2365  2369  2373  2383  2384  2385
               2385  2387  2388  2389  2390
.IFF            190   309   310   312   313   334   338   340   346   348   355   369   372   375   401
                411   415   574   576   585   679   680   681   692   687   699   699   690   695   699
                699   700   701   707   710   711   712   713   718   721   722   723   724   729   730
                733   734   735   741   746   747   748   749   762   763   764   765   790   791   793
                793   800   803   804   805   806   822   825   826   827   828   829   840   841   843
                854   855   856   857   864   868   869   870   871   876   879   880   890   891   893
                898   901   892   893   899   903   904   905   906   911   914   915   916   917   923
```

```
                926   927   928   929   936   941   942   943   944   955   958   959   960   961   969
                972   973   974   975   982   986   987   988   989   996   999  1000  1001  1002  1015
               1026  1035  1036  1037  1038  1051  1060  1061  1062  1063  1081  1089  1090  1091  1092
               1158  1161  1162  1163  1164  1230  1233  1234  1235  1236  1243  1248  1249  1250  1251
               1258  1265  1266  1267  1268  1294  1292  1293  1294  1295  1315  1321  1322  1323  1324
               1343  1345  1353  1354  1355  1356  1375  1381  1382  1383  1384  1389  1404  1413  1414
               1415  1416  1418  1425  1433  1443  1449  1450  1451  1452  1454  1461  1469  1480  1481
               1482  1483  1488  1503  1513  1514  1515  1516  1517  1530  1531  1532  1533  1549  1550
               1551  1552  1570  1578  1579  1580  1581  1582  1592  1593  1594  1595  1622  1626  1632
               1637  1640  1653  1770  1796  1799  1800  1803  1830  1831  1835  1837  1850  1875  1880
               1884  1923  1945  1979  1981  1986  1998  1999  2008  2012  2029  2040  2055  2084  2089
               2168  2194  2199  2220  2301  2350  2366
.IFF            634   638   642   648   655  1757  1811  1856  1902  1918  1919  1923  1945  1981  1986
.IFTF           634   638   642   648   655  1757  1809  1855  1898  1902  1918  1937  1951  1979  1982
.IIF            177   182   187   306   307   308   309   312   313   321   410   415   586   589   595
                596   597   599   600   616  1624  1625  1632  1633  1653  1656  1773  1774  1775  1776
               1777  1778  1782  1810  1811  1827  1830  1831  1838  1839  1840  1841  1946  1868  1875
               1880  1923  1939  2021  2029  2035  2052  2077  2296  2382  2383  2384  2385  2386  2387
               2388  2389
.IRP            536   678   696   697   709   720   731   745   761   779   802   824   839   953   957
                978   889   902   913   925   940   957   971   985   998  1034  1059  1088  1160  1232
               1247  1264  1291  1320  1352  1380  1412  1448  1479  1512  1529  1548  1577  1591  1782
               1893  1914  2171  2177  2190  2191  2306  2307  2328  2344  2345
.LIST           177   296   312   321   401   403   404   405   411   415   536   601   634   638   642
                648   655   678   682   686   690   697   701   709   713   720   724   731   735   745
                749   761   765   779   783   802   806   824   828   839   843   853   857   867   871
                878   882   889   893   902   906   913   917   925   929   940   944   957   961   971
                975   985   989   998  1002  1034  1038  1059  1063  1088  1092  1160  1164  1232  1236
               1247  1251  1264  1268  1291  1295  1320  1324  1352  1356  1380  1384  1412  1416  1448
               1452  1479  1483  1512  1516  1529  1533  1548  1552  1577  1581  1591  1595  1632  1645
               1757  1777  1875  1998  2373  2382  2383  2384  2385  2386  2387  2388  2389  2390
.MACRO          313   365   573   617   939   940  2373
.MCALL          177   298   411   601
.NLIST          177   298   312   321   401   403   404   405   411   415   536   601   634   638   642
                648   655   678   682   686   690   697   701   709   713   720   724   731   735   745
                749   761   765   779   783   802   806   824   828   839   843   853   857   867   871
                878   882   889   893   902   906   913   917   925   929   940   944   957   961   971
                975   985   989   998  1002  1034  1038  1059  1063  1088  1092  1160  1164  1232  1236
               1247  1251  1264  1268  1291  1295  1320  1324  1352  1356  1380  1384  1412  1416  1448
               1452  1479  1483  1512  1516  1529  1533  1548  1552  1577  1581  1591  1595  1632  1645
               1757  1777  1875  1998  2373  2382  2383  2384  2385  2386  2387  2388  2389  2390
.PAGE           301   365   410
.REM              1
.REPT           321   403   785   808  1099  1171
.SBTTL          188   302   315   325   331   343   366   412   484   678   686   697   709   720   731
                745   761   779   802   824   939   853   867   878   889   902   913   925   940   957
                971   985   998  1034  1059  1088  1160  1232  1247  1264  1291  1320  1352  1380  1412
               1448  1479  1512  1529  1548  1577  1591  1619  1750  1767  1832  1881  1920  2037  2086
               2165  2217  2298  2357  2374
.TITLE          177
.WORD           321   322   323   339   359   360   361   362   363   364   374   377   378   379   390
                383   384   385   386   387   388   389   390   391   392   401   403   404   417   419
                419   420   421   422   423   424   428   429   430   443   447   450   453   459   463
                461   462   463   464   465   466   467   468   469   470   471   472   473   474   475
                476   477   478  1637  1640  1652  1918  2063  2068  2162  2200  2202  2246  2293  2333
```

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*,DZLPIB.SEQ/SOL/CRF DS:ERF2=DZLPIB.P11
RUN-TIME: 57 29 6 SECONDS
RUN-TIME RATIO: 224/93=2.3
CORE USED: 24K (47 PAGES)

Spooler running 10 Seconds, 307 KCS, 289 disk reads, 4 disk writes, 21 pages