DR11-K

DIGITAL I/O TEST

MD-11-DZDRG-C

EP-DZDRG-C-DL-A

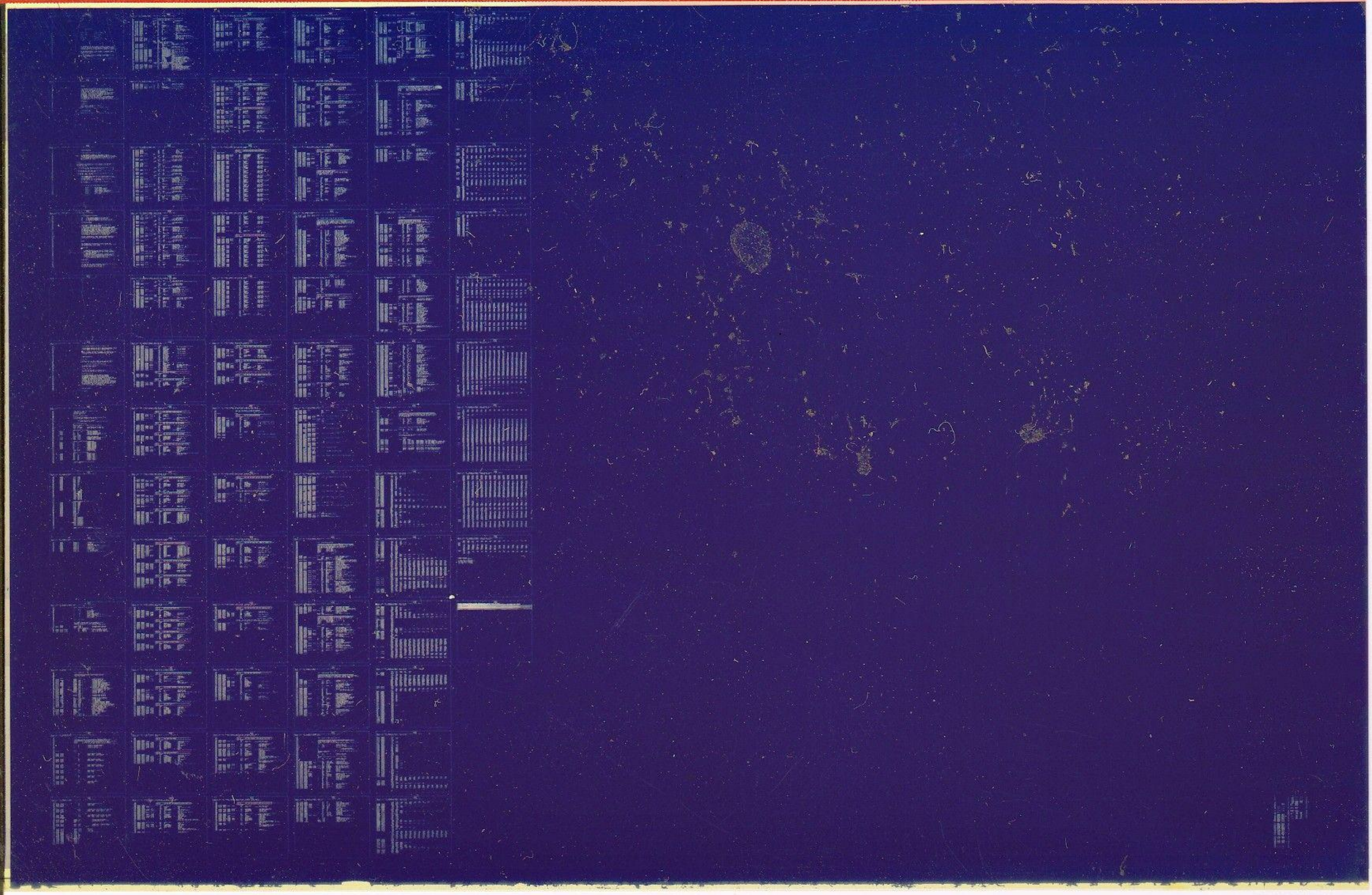
COPYRIGHT 1976

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA



.REM%

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZDRG-C-D
PRODUCT NAME:	DR11-K DIGITAL I/O TEST
DATE:	MAY 21, 1976
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	RAYMOND SHOOP

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1976 BY DIGITAL EQUIPMENT CORPORATION.

1. ABSTRACT

THIS PROGRAM IS A LOGIC TEST OF THE DR11-K 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 CHARACTERSTICS.
THE PROGRAM WILL HANDLE ALL CONFIGURATIONS OF INPUT INTERRUPT
SWITCHES AND INPUT DATA LATCHING JUMPERS. FOR SYSTEMS WITH
CONSECUTIVE MULTIPLE DR11-K'S, THESE CONFIGERATIONS MUST
BE THE SAME. THE FOLLOWING JUMPERS MUST
BE INSERTED TO EXECUTE THE LOGIC TEST: W21A, W22A AND W23A.
THIS PROGRAM WILL TEST SEQUENTIAL DR11-K'S STARTING AT THE
BUS ADDRESS AND VECTOR IN LOCATIONS "BASEBA" AND "BASEIV".
FOR NORMAL FACTORY CONFIG., ALL SWITCH REGISTER BITS SHOULD BE RESET.

2. REQUIREMENTS

2.1 EQUIPTMENT

PDP-11 FAMILY COMPUTER WITH TELETYPE
DR11-K OPTION INSTALLED
BC08R-1 ONE FOOT OUTPUT TO INPUT WRAPAROUND CABLE

3. LOADING PROCEDURE

THE PROCEDURE FOR LOADING BINARY TAPES SHOULD BE FOLLOWED.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS (LOGIC TEST)

IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER. IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH REGISTER (LOC. 176) IS USED.

CONTROL:

THIS PROGRAM ALSO SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER (LOC. 176) FROM THE TTY. THIS CAN BE ACCOMPLISHED BY DOING THE FOLLOWING:

1) TYPE CONTROL G <↑G>; THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.

2) THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)

3) AFTER THE ''NEW='' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:

   A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED AND ONLY 6 NUMBERS WILL BE ALLOWED)
   IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.

   B) IF A CONTROL U <↑U> IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 2.

WILL BE USED AS THE SOFTWARE DISPLAY REGISTER.

SW 15 = 1	HALT ON ERROR
SW 14 = 1	LOOP ON CURRENT SUB-TEST
SW 13 = 1	INHIBIT ERROR TYPOUTS
SW 12 = 1	LOOP ON CURRENTLY SELECTED DR11-K
SW 11 = 1	INHIBIT SUB-TEST INTERATIONS
SW 10 = 1	OUTPUT TO INPUT WRAPAROUND CABLE NOT CONNECTED
SW 09 = 1	LOOP ON ERROR
SW 08 = 1	LOOP ON TEST IN SWR <7:0>

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 CONTROL LINE LOOP.
214 IS THE STARTING ADDRESS OF THE COULTER INTERFACE LOOP

5. OPERATING PROCEDURE

THE FOLLOWING JUMPERS MUST BE INSTALLED TO EXECUTE THE
LOGIC TEST: W21A, W22A AND W23A
IF THE CUSTOMER HAS SELECTED THE "B" SECTION OF THESE JUMPERS
IT MUST BE RETURNED TO THE "FACTORY" POSITION BEFORE RUNNING
THE LOGIC TEST. ** WORST CASE WILL ONLY BE CHANGING THREE JUMPERS. **
THE OPERATOR MUST INSERT THE CORRECT INFORMATION IN THE SWITCH
REGISTER WHEN REQUESTED BY THE PROGRAM OR AN ERROR WILL OCCUR.
WITH THE INPUT INTERRUPT SWITCHES AND DATA LATCHING JUMPERS
IN THE FACTORY POSITION, ALL SWITCH REGISTER BITS SHOULD BE RESET.
ONCE STARTED THE TEST WILL RUN IN IT'S NORMAL MANNER
WITHOUT OPERATOR INTERVENTION OR SWITCH SELECTION.

*** NOTE: OPERATOR MUST INSERT INFORMATION WHEN REQUESTED BY
PROGRAM. THE MACHINE WILL TYPE: SWR=XXXXXXNEW= AFTER NEW=
THE INFORMATION IS INSERTED. REFER TO SECTION 4.1 2) ***

6. ERRORS

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

7. RESTRICTIONS

1. IF SEQUENTIAL DR11-K'S, ALL DR11-K'S MUST BE IN THE
   SAME INTERRUPT SWITCHES AND DATA PATH JUMPER CONFIGERATION.
2. THE FOLLOWING JUMPERS MUST BE IN THE "FACTORY" POSITION:
   W21A, W22A AND W23A
3. THE OPERATOR MUST SUPPLY THE CORRECT INTERRUPT SWITCHES
   AND DATA PATH JUMPER JUMPER AND SWITCH CONFIGURATION
   INFORMATION TO THE INITALIZATION QUESTIONS OR AN ERROR WILL OCCUR.
4. FOR MULTIPLE GROUPS OF CONSECUTIVE DR11-K'S:
   THIS DIAGNOSTIC MUST BE RUN FOR EACH GROUP.

8. MISCELANEOUS

8.1 EXECUTION TIME

THE LOGIC TEST WILL TAKE APPROXIMATELY 60 SECONDS FOR COMPLETION
ON A PDP11/05 TYPE AND WILL TYPE 'END PASS'.
THE CONTROL LINE LOOP WILL NEVER EXIT.
THE COULTER INTERFACE LOOP WILL NEVER EXIT.

```
   182
```

8.2 DEVICE ADDRESS PROGRAM LOCATIONS (AT APPROX. 1300)

LOCATION BASEAD CONTAINS THE DR11-K BASE DEVICE ADDRESS (767770)
LOCATION BASEIV CONTAINS THE DR11-K BASE INTERRUPT VECTOR (300)
LOCATION BASEBR CONTAINS THE DR11-K 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.

9. PROGRAM DESCRIPTION

9.1 LOGIC TEST

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

W21A, W22A AND W23A CAN BE DIAGNOSED.

THE PROGRAM CHECKS THAT THE DR11-K CAN INTERRUPT AND THAT "RESET" WILL WORK CORRECTLY.

9.2 CONTROL LINE LOOP

THIS TEST LOOP PROVIDES THE OPERATOR WITH A SCOPE LOOP FOR CHECKING W21, W22 AND W23 IN THE "B" POSITION.

9.3 COULTER FUNCTION LOOP

THE COULTER INTERFACE LOOP PROVIDES THE OPERATOR WITH SOFTWARE INTERFACE WITH THE COULTER HARDWARE.
THE OPERATOR IS INSTRUCTED TO RUN A SAMPLE ON THE COULTER.
THE PROGRAM WILL WAIT FOR AN INTERRUPT FROM THE DR11K. UPON AN INTERRUPT, THE PROGRAM DELAYS 70 MSEC. TO ALLOW THE DATA LINES TO SETTLE. THE INPUT DATA IS SAVED IN A CORE BUFFER.
THE HIGHER FOUR BITS ARE THE TEST NUMBERS WITH ZERO BEING THE RUN TERMINATOR. UPON COMPLETION OF A RUN (SAMPLE), THE PROGRAM WILL TYPE THE RESULTS IN THE FOLLOWING FORMAT AND THEN RESTART THE RUN SEQUENCE. THE TYPED RESULTS CAN BE COMPARED TO THE COULTER TYPOUT TO VERIFY A SUCCESSFUL RUN.

"N-XXX" WHERE N = TEST NUMBER AND XXX = TEST RESULTS

10. TABLE OF CONTENTS

ATTACHED.%

```
                                .TITLE  DR11-K  LOGIC TEST  MAINDEC-11-DZDRG-C
                                ;*COPYRIGHT (C) 1975
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY RAYMOND SHOOP
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C1),MAR 24, 1976.
                                ;*
                                        ;REVISED 21 MAY 76 BY F. ROEMER
                                        ;TO SUPPORT DYNAMIC LOADING OF SOFTWARE SWITCH REGISTER

                                .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
```

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

                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                        000004  ERRVEC= 4                       ;;TIME OUT AND OTHER ERRORS
```

```
                                                
   344          000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
   345          000014                  TBITVEC=14              ;;"T" BIT
   346          000014                  TRTVEC= 14              ;;TRACE TRAP
   347          000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
   348          000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
   349          000024                  PWRVEC= 24              ;;POWER FAIL
   350          000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
   351          000034                  TRAPVEC=34              ;;"TRAP" TRAP
   352          000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
   353          000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
   354          000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
```

```
                                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                                        ;*
                                        ;*      SWITCH                  USE
                                        ;*      ------          --------------------
                                        ;*        15            HALT ON ERROR
                                        ;*        14            LOOP ON TEST
                                        ;*        13            INHIBIT ERROR TYPEOUTS
                                        ;*        12            LOOP ON CURRENTLY SELECTED DR11-K
                                        ;*        11            INHIBIT ITERATIONS
                                        ;*        10            OUTPUT TO INPUT WRAPAROUND 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  001366                         JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  001372                         JMP     @#BEGIN1        ;JUMP TO THE RESTART ADDRESS
000210  000137  011726                         JMP     @#EXTTST        ;JUMP TO THE CONTROL LINES LOOP
000214  000137  012032                         JMP     @#COULTR        ;JUMP TO THE COULTER FUNCTION LOOP
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102  000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002             $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000             $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHIC  ($REGO) WAS OBTAINED
001162  000000          $REGO:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000          $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
                        $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
                        $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001172  077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
001173  015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001174  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************
```

```
                                .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

001176                          $ERRTB:

                                ;ITEM   1
001176  012322                          EM1             ;STATUS REGISTER IN ERROR
001200  012742                          DH1             ;ERRPC  DRADD   STATUS EXPECTED
001202  013176                          DT1             ;$ERRPC DRADD   $BDDAT $GDDAT
001204  000000                          0

                                ;ITEM   2
001206  012353                          EM2             ;INPUT REGISTER IN ERROR
001210  013003                          DH2             ;ERRPC  DRADD   INPUT EXPECTED
001212  013176                          DT1             ;$ERRPC DRADD   $BDDAT $GDDAT
001214  000000                          0

                                ;ITEM 3
001216  012403                          EM3             ;OUTPUT REGISTER IN ERROR
001220  013044                          DH3             ;ERRPC  DRADD   OUTPUT EXPECTED
001222  013176                          DT1             ;$ERRPC DRADD   $BDDAT $GDDAT
001224  000000                          0

                                ;ITEM   4
001226  012434                          EM4             ;INPUT FAILED TO INTERRUPT
001230  013105                          DH4             ;ERRPC  DRADD
001232  013210                          DT4             ;$ERRPC DRADD
001234  000000                          0

                                ;ITEM   5
001236  012466                          EM5             ;OUTPUT FAILED TO INTERRUPT
001240  013105                          DH4             ;ERRPC  DRADD
001242  013210                          DT4             ;$ERRPC DRADD
001244  000000                          0

                                ;ITEM   6
001246  012521                          EM6             ;UNEXPECTED INTERRUPT
001250  013105                          DH4             ;ERRPC  DRADD
001252  013210                          DT4             ;$ERRPC DRADD
001254  000000                          0
```

```
                                ;ITEM   7
001256  012546                          EM7             ;OPERATOR INTERVENTION ERROR
001260  013105                          DH4             ;ERRPC  DRADD
001262  013210                          DT4             ;$ERRPC DRADD
001264  000000                          0

                                ;ITEM   10
001266  012602                          EM10            ;INTERRUPT INPUT BIT FAILED TO SET INPUT READY
001270  013123                          DH10            ;ERRPC  DRADD   STATUS  EXPECTED  INPUT BIT
001272  013216                          DT10            ;$ERRPC DRADD   $BDDAT  $GDDAT    BRLEV3
001274  000000                          0

                                ;ITEM   11
001276  012665                          EM11            ;NON-INTERRUPTING INPUT BIT SET INPUT READY
001300  013123                          DH10            ;ERRPC  DRADD   STATUS  EXPECTED  INPUT BIT
001302  013216                          DT10            ;$ERRPC DRADD   $BDDAT  $GDDAT    BRLEV3
001304  000000                          0

001306  167770          BASEBA: 167770                  ;STARTING BASE BUS ADDRESS
001310  000300          BASEIV: 300                     ;STARTING BASE INTERRUPT ADDRESS
001312  000000          NMBEXT: 0                       ;ADDITIONAL DR-11-K
001314  000000          NBEXT:  0

001316  167770          DRADD:  167770                  ;CURRENT DR11-K STARTING ADDRESS
001320  000300          DRIV:   300                     ;CURRENT DR11-K STARTING INTERRUPT VECTOR
001322  000200          DRBRL:  200                     ;DRA BR LEVEL


001324  167770          GRSTAT: 167770          ;DR STATUS
001326  167772          GRDAI:  167772          ;INPUT REG.
001330  167774          GRDIO:  167774          ;OUTPUT REG.
001332  167775          GRBHIO: 167775          ;OUTPUT REG HIGH BYTE
001334  000000          NOTLCH: 0
001336  000000          INTBIT: 0
001340  000300          GRIVA:  300
001342  000302          GRIVSA: 302
001344  000304          GRIVB:  304
001346  000306          GRIVSB: 306
001350  000200          DIOBRL: 200
001352  000000          BRLEV1: 0
001354  000000          BRLEV2: 0
001356  000000          BRLEV3: 0
001360  000000          ODDJMP: 0
001362  000000          MINSIN: 0
001364  000000          TRANST: 0
```

```
                                        ;;*******************************************************
                                        ;       DIGITAL I-O LOGIC TEST
                                        ;*******************************************************
001366  005000                  BEGIN:  CLR     R0                      ;CLEAR R0
001370  000402                          BR      RBEG
001372  012700  177777          BEGIN1: MOV     #-1,R0                  ;LOAD R0
001376  000005                  RBEG:   RESET
001400  012706  001100                  MOV     #STACK,SP               ;LOAD STACK
001404  012737  001432  000004          MOV     #1$,@#4                 ;LOAD BUS ERROR
001412  013702  001306                  MOV     BASEBA,R2               ;LOAD STARTING ADDRESS
001416  005003                          CLR     R3                      ;CLEAR COUNT
001420  005712                  2$:     TST     (R2)                    ;TEST IF EXISTENT
001422  162702  000010                  SUB     #10,R2                  ;EXIST, UPDATE TEST ADDRESS
001426  005203                          INC     R3                      ;UPDATE # OF DR11'S
001430  000773                          BR      2$
001432  022626                  1$:     CMP     (SP)+,(SP)+             ;POP STACK
001434  005703                          TST     R3                      ;TEST IF FIRST DOES EXIST
001436  001001                          BNE     3$                      ;BR
001440  000000                          HALT                            ;FIRST DR11-K DOES NOT EXIST
                                                                        ;CHECK THE PROGRAM DEVICE ADDRESS
001442  005303                  3$:     DEC     R3                      ;ADJUST R3
001444  010337  001312                  MOV     R3,NMBEXT               ;SAVE THE NUMBER OF ADDITIONAL DR11-K
001450  012737  000006  000004          MOV     #6,@#4                  ;RESET BUS ERROR
001456  005037  000006                  CLR     @#6
001462  000005                  RBEG1:  RESET
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001464  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001470  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001472  022706  001140                  CMP     #SWR,R6 ;;DONE?
001476  001374                          BNE     .-6             ;;LOOP BACK IF NO
001500  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
001504  012737  013232  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001512  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001520  012737  013504  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001526  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001534  012737  015566  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001542  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001550  012737  014224  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001556  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001564  013737  011412  011404          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
001572  005037  001166                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
001576  005037  001170                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001602  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
001610  012737  001610  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
001616  012737  001616  001110          MOV     #.,$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.
001624  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001630  012737  001664  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001636  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001644  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001652  022777  177777  177260          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001660  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
```

```
583  001662  000403                        BR      65$                ;;BRANCH IF NO TIMEOUT
584  001664  012716  001672          64$:  MOV     #65$,(SP)          ;;SET UP FOR TRAP RETURN
585  001670  000002                        RTI
586  001672  012737  000176  001140  65$:  MOV     #SWREG,SWR         ;;POINT TO SOFTWARE SWR
587  001700  012737  000174  001142        MOV     #DISPREG,DISPLAY
588  001706  012637  000004          66$:  MOV     (SP)+,@#ERRVEC     ;;RESTORE ERROR VECTOR
589
```

```
                                                TST     @#42                    ;TEST IF UNDER A MONITOR
                                                BNE     4$                      ;BR IF
                                                TST     R0                      ;TEST R0
                                                BEQ     2$                      ;BR IF CLEARED
                                        4$:     JMP     IOTEST                  ;JUMP IF SET
                                        2$:
                                                TYPE    ,68$                ;;TYPE ASCIZ STRING
                                                BR      67$                 ;;GET OVER THE ASCIZ
                                        ;;68$:  .ASCIZ  <15><12><15><12>/DR11-K DIGITAL INPUT OUTPUT LOGIC TEST/
                                        67$:
                                                TYPE    ,70$                ;;TYPE ASCIZ STRING
                                                BR      69$                 ;;GET OVER THE ASCIZ
                                        ;;70$:  .ASCIZ  <15><12>/MAINDEC-11-DZDRG-C/<15><12>
                                        69$:
                                                MOV     NMBEXT,-(SP)
                                                TYPOS
                                                .WORD   2
                                                TYPE    ,72$                ;;TYPE ASCIZ STRING
                                                BR      71$                 ;;GET OVER THE ASCIZ
                                        ;;72$:  .ASCIZ  /(8) ADDITIONAL DR11-K'S CONNECTED/
                                        71$:
                                                TYPE    ,74$                ;;TYPE ASCIZ STRING
                                                BR      73$                 ;;GET OVER THE ASCIZ
                                        ;;74$:  .ASCIZ  <15><12>/SET SWITCH REGISTER BITS EQUAL TO THE NON-LATCHING BITS/
                                        73$:
                                                TYPE    ,76$                ;;TYPE ASCIZ STRING
                                                BR      75$                 ;;GET OVER THE ASCIZ
                                        ;;76$:  .ASCIZ  <15><12>/DEPRESS CONT./
                                        75$:
                                                HALT                            ;WAIT FOR OPERATOR
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                                                TST     @#42                ;;ARE WE RUNNING UNDER XXDP/ACT?
                                                BNE     77$                 ;;BRANCH IF YES
                                                CMP     SWR,#SWREG          ;;SOFTWARE SWITCH REG SELECTED?
                                                BNE     78$                 ;;BRANCH IF NO
                                                GTSWR                       ;;GET SOFT-SWR SETTINGS
                                                BR      78$
                                        77$:    MOVB    #1,$AUTOB           ;;SET AUTO-MODE INDICATOR
                                        78$:
                                                MOV     @SWR,NOTLCH             ;SAVE SWITCHES
                                                TYPE    ,80$                ;;TYPE ASCIZ STRING
                                                BR      79$                 ;;GET OVER THE ASCIZ
                                        ;;80$:  .ASCIZ  <15><12>/SET SWITCH REGISTER BITS EQUAL TO THE INTERRUPTING BITS/
                                        79$:
                                                TYPE    ,82$                ;;TYPE ASCIZ STRING
                                                BR      81$                 ;;GET OVER THE ASCIZ
                                        ;;82$:  .ASCIZ  <15><12>/DEPRESS CONT./
                                        81$:
                                                HALT                            ;WAIT FOR OPERATOR
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                                                TST     @#42                ;;ARE WE RUNNING UNDER XXDP/ACT?
                                                BNE     83$                 ;;BRANCH IF YES
                                                CMP     SWR,#SWREG          ;;SOFTWARE SWITCH REG SELECTED?
                                                BNE     84$                 ;;BRANCH IF NO
                                                GTSWR                       ;;GET SOFT-SWR SETTINGS
                                                BR      84$
```

```
                                                83$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
                                                84$:
                                                        MOV     @SWR,INTBIT             ;SAVE SWITCHES
                                                        TYPE    ,86$            ;;TYPE ASCIZ STRING
                                                        BR      85$             ;;GET OVER THE ASCIZ
                                                ;;86$:  .ASCIZ  <15><12>/SET SWITCH REGISTER BITS 15-12 EQUAL TO POSITIVE INPUT BITS/
                                                85$:
                                                        TYPE    ,88$            ;;TYPE ASCIZ STRING
                                                        BR      87$             ;;GET OVER THE ASCIZ
                                                ;;88$:  .ASCIZ  <15><12>/DEPRESS CONT./
                                                87$:
                                                        HALT
                                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                                                        TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
                                                        BNE     89$             ;;BRANCH IF YES
                                                        CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
                                                        BNE     90$             ;;BRANCH IF NO
                                                        GTSWR                   ;;GET SOFT-SWR SETTINGS
                                                        BR      90$
                                                89$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
                                                90$:
                                                        MOV     @SWR,MINSIN             ;SAVE MINUS INPUT
                                                        TYPE    ,92$            ;;TYPE ASCIZ STRING
                                                        BR      91$             ;;GET OVER THE ASCIZ
                                                ;;92$:  .ASCIZ  <15><12>/SET SWITCH REGISTER BITS 15-12 EQUAL TO TRANSITION INPUT BITS/
                                                91$:
                                                        TYPE    ,94$            ;;TYPE ASCIZ STRING
                                                        BR      93$             ;;GET OVER THE ASCIZ
                                                ;;94$:  .ASCIZ  <15><12>/DEPRESS CONT./
                                                93$:
                                                        HALT
                                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                                                        TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
                                                        BNE     95$             ;;BRANCH IF YES
                                                        CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
                                                        BNE     96$             ;;BRANCH IF NO
                                                        GTSWR                   ;;GET SOFT-SWR SETTINGS
                                                        BR      96$
                                                95$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
                                                96$:
                                                        MOV     @SWR,TRANST             ;SAVE SWITCHES

                                                .SBTTL  DETERMINE THAT THE OPERATOR DID NOT MAKE A LOGICAL SWITCH ERROR
                                                        MOV     #BIT12,$GDDAT           ;LOAD TEST BIT
                                                10$:    BIT     $GDDAT,MINSIN           ;TEST IF NEG INPUT BIT
                                                        BEQ     11$                     ;BR IF NOT
                                                        BIT     $GDDAT,TRANST           ;TEST IF TRANS. INPUT
                                                        BEQ     11$                     ;BR IF NOT
                                                        ERROR   7                       ;LOGICAL OPERATOR ERROR
                                                                                        ;INPUT BIT CANNOT BE NEG. AND TRANS. AT THE SAME
                                                        JMP     BEGIN                   ;
                                                11$:    ROL     $GDDAT                  ;MOVE LEFT
                                                        BCC     10$                     ;BR UNTIL DONE

                                                        TYPE    ,98$            ;;TYPE ASCIZ STRING
```

```
   702  003140  000434                          BR      97$                     ;;GET OVER THE ASCIZ
   703                                  ;;98$:  .ASCIZ  <15><12>/SET SWITCH REGISTER WITH THE DESIRED PROGRAM OPTIONS/
   704  003232                          97$:
   705  003232  104401  003240                   TYPE    ,100$                   ;;TYPE ASCIZ STRING
   706  003236  000412                          BR      99$                     ;;GET OVER THE ASCIZ
   707                                  ;;100$: .ASCIZ  <15><12>/DEPRESS CONT./<15><12><12>
   708  003264                          99$:
   709  003264  000000                          HALT                            ;WAIT FOR OPERATOR
   710                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
   711  003266  005737  000042                   TST     @#42                    ;;ARE WE RUNNING UNDER XXDP/ACT?
   712  003272  001006                          BNE     101$                    ;;BRANCH IF YES
   713  003274  023727  001140  000176          CMP     SWR,#SWREG              ;;SOFTWARE SWITCH REG SELECTED?
   714  003302  001005                          BNE     102$                    ;;BRANCH IF NO
   715  003304  104406                          GTSWR                           ;;GET SOFT-SWR SETTINGS
   716  003306  000403                          BR      102$
   717  003310  112737  000001  001134  101$:   MOVB    #1,$AUTOB               ;;SET AUTO-MODE INDICATOR
   718  003316                          102$:
   719  003316  000005                  IOTEST: RESET
   720  003320  013737  001306  001316          MOV     BASEBA,DRADD            ;LOAD INITIAL STARTING ADDRESS
   721  003326  013737  001310  001320          MOV     BASEIV,DRIV             ;LOAD INITIAL STARTING VECTOR
   722  003334  013737  001312  001314          MOV     NMBEXT,NBEXT            ;RELOAD # AVAILABLE
   723  003342  000005                  RBEG2:  RESET
   724  003344  012702  000232                  MOV     #232,R2                 ;LOAD R2
   725  003350  012701  000230                  MOV     #230,R1                 ;LOAD R1
   726  003354  010221                  5$:     MOV     R2,(R1)+                ;LOAD .+2
   727  003356  005021                          CLR     (R1)+                   ;LOAD HALT
   728  003360  010102                          MOV     R1,R2                   ;LOAD R2
   729  003362  005722                          TST     (R2)+                   ;BUMP R2
   730  003364  020227  001076                  CMP     R2,#$CMTAG-2            ;TEST FOR LAST
   731  003370  001371                          BNE     5$                      ;BR UNTIL DONE
```

```
                                IOTST1: RESET
003372  000005
003374  004737  003402                  JSR     PC,SETADD               ;SET UP BUS ADDRESS AND VECTOR
003400  000453                          BR      IOTTS1
003402  013737  001316  001324  SETADD: MOV     DRADD,GRSTAT            ;LOAD 1ST ADDRESS
003410  013737  001316  001326          MOV     DRADD,GRDAI             ;LOAD 2ND ADDRESS
003416  062737  000002  001326          ADD     #2,GRDAI
003424  013737  001316  001330          MOV     DRADD,GRDIO             ;LOAD 3RD ADDRESS
003432  062737  000004  001330          ADD     #4,GRDIO
003440  013737  001316  001332          MOV     DRADD,GRBHIO            ;LOAD 4TH ADDRESS
003446  062737  000005  001332          ADD     #5,GRBHIO
003454  013737  001320  001340          MOV     DRIV,GRIVA              ;LOAD FIRST VECTOR
003462  013737  001320  001342          MOV     DRIV,GRIVSA
003470  062737  000002  001342          ADD     #2,GRIVSA
003476  013737  001320  001344          MOV     DRIV,GRIVB              ;LOAD 2ND VECTOR
003504  062737  000004  001344          ADD     #4,GRIVB
003512  013737  001320  001346          MOV     DRIV,GRIVSB
003520  062737  000006  001346          ADD     #6,GRIVSB
003526  000207                          RTS     PC
003530  013737  001322  001350  IOTTS1: MOV     DRBRL,DIOBRL            ;LOAD BR LEVEL
003536  005037  001360                  CLR     ODDJMP
003542  053737  001362  001360          BIS     MINSIN,ODDJMP           ;SET MINUS INPUT BITS
003550  053737  001364  001360          BIS     TRANST,ODDJMP           ;SET TRANSITION INPUT BITS
003556  012706  001100                  MOV     #STACK,SP               ;LOAD STACK
003562  013777  001342  175550          MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
003570  005077  175546                  CLR     @GRIVSA
003574  013777  001346  175542          MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
003602  005077  175540                  CLR     @GRIVSB
                        ;;**************************************************************
                        ;*TEST 1         TEST FOR NO BUS ERRORS
                        ;;**************************************************************
003606  000004            TST1:   SCOPE
003610  005077  175510            CLR     @GRSTAT
003614  005077  175506            CLR     @GRDAI
003620  005077  175504            CLR     @GRDIO

                        ;;**************************************************************
                        ;*TEST 2         TEST THAT OUTPUT REG. CAN HOLD #-1
                        ;;**************************************************************
003624  000004            TST2:   SCOPE
003626  012737  177777  001124    MOV     #-1,$GDDAT              ;LOAD EXPECTED
003634  012777  177777  175466    MOV     #-1,@GRDIO              ;ALL ONES TO REGISTER
003642  017737  175462  001126    MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
003650  023737  001124  001126    CMP     $GDDAT,$BDDAT           ;COMPARE
003656  001401                    BEQ     TST3            ;;BR IF EQUAL
003660  104003                    ERROR   3                       ;REG WILL NOT HOLD ONES
```

```
                                        ;;*****************************************************************
                                        ;*TEST 3        TEST THAT RESET CLEARS OUTPUT REG.
                                        ;;*****************************************************************
003662  000004                  TST3:   SCOPE
003664  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
003672  005037  001124                  CLR     $GDDAT
003676  012777  177777  175424          MOV     #-1,@GRDIO
003704  000005                          RESET                           ;SET DATA TO ALL ONES
003706  017737  175416  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
003714  001401                          BEQ     TST4            ;;BR IF EQUAL
003716  104003                          ERROR   3                       ;REG FAILED TO CLEAR

                                        ;;*****************************************************************
                                        ;*TEST 4        TEST THAT OUTPUT REG. CAN HOLD #52525
                                        ;;*****************************************************************
003720  000004                  TST4:   SCOPE
003722  012737  052525  001124          MOV     #52525,$GDDAT           ;LOAD EXPECTED VALUE
003730  012777  052525  175372          MOV     #52525,@GRDIO
003736  017737  175366  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
003744  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
003752  001401                          BEQ     TST5            ;;BR IF EQUAL
003754  104003                          ERROR   3                       ;DATA NOT=52525

                                        ;;*****************************************************************
                                        ;*TEST 5        TEST THAT OUTPUT REG. CAN HOLD #125252
                                        ;;*****************************************************************
003756  000004                  TST5:   SCOPE
003760  012737  125252  001124          MOV     #125252,$GDDAT          ;LOAD EXPECTED VALUE
003766  012777  125252  175334          MOV     #125252,@GRDIO
003774  017737  175330  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
004002  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004010  001401                          BEQ     TST6            ;;BR IF EQUAL
004012  104003                          ERROR   3                       ;DATA NOT=125252

                                        ;;*****************************************************************
                                        ;*TEST 6        TEST THAT OUTPUT REG. CAN HOLD A COUNT PATTERN
                                        ;;*****************************************************************
004014  000004                  TST6:   SCOPE
004016  012737  000004  001166          MOV     #4,$TIMES       ;;DO 4 ITERATIONS
004024  012737  004036  001110          MOV     #2$,$LPERR              ;LOAD SCOPE ERROR RETURN
004032  005037  001124                  CLR     $GDDAT                  ;CLEAR PATTERN
004036  013777  001124  175264  2$:     MOV     $GDDAT,@GRDIO           ;LOAD THE OUTPUT REG.
004044  017737  175260  001126          MOV     @GRDIO,$BDDAT           ;READ THE REG.
004052  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
004060  001401                          BEQ     1$              ;;BR IF EQUAL
004062  104003                          ERROR   3                       ;OUTPUT REG.FAILED TO HOLD COUNT PATTERN
004064  005237  001124          1$:     INC     $GDDAT                  ;UPDATE THE PATTERN
004070  001362                          BNE     2$                      ;TRY AGAIN
```

```
                                        ;;*****************************************************************
                                        ;*TEST 7         FLOAT A 1 ACROSS THE OUTPUT REGISTER
                                        ;;*****************************************************************
004072  000004                  TST7:   SCOPE
004074  012737  004110  001110          MOV     #1$,$LPERR              ;LOAD SCOPE ERROR RETURN
004102  012737  000001  001124          MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE

004110  005077  175214          1$:     CLR     @GRDIO                  ;CLEAR OUTPUT
004114  053777  001124  175206          BIS     $GDDAT,@GRDIO           ;SET THAT BIT
004122  017737  175202  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
004130  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST RESULTS
004136  001401                          BEQ     2$                      ;BR IF EQUAL ?
004140  104003                          ERROR   3

004142  006337  001124          2$:     ASL     $GDDAT                  ;SHIFT EXPECTED DATA
004146  001360                          BNE     1$                      ;BR UNTIL DONE
                                        ;;*****************************************************************
                                        ;*TEST 10        FLOAT A 0 ACROSS THE OUTPUT REGISTER
                                        ;;*****************************************************************
004150  000004                  TST10:  SCOPE
004152  012737  004166  001110          MOV     #1$,$LPERR              ;LOAD SCOPE ERROR RETURN
004160  012737  000001  001124          MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE

004166  012777  177777  175134  1$:     MOV     #-1,@GRDIO              ;LOAD OUTPUT TO A ONE
004174  043777  001124  175126          BIC     $GDDAT,@GRDIO           ;CLEAR A BIT
004202  017737  175122  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
004210  005137  001126                  COM     $BDDAT                  ;COMPLEMENT IT
004214  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;EQUAL ?
004222  001401                          BEQ     2$                      ;BR IF EQUAL
004224  104003                          ERROR   3

004226  006337  001124          2$:     ASL     $GDDAT                  ;SHIFT LEFT
004232  001355                          BNE     1$                      ;BRANCH UNTIL DONE

                                        ;;*****************************************************************
                                        ;*TEST 11        TEST FOR SLOW OUTPUT GATES WITH #125252
                                        ;;*****************************************************************
004234  000004                  TST11:  SCOPE
004236  012737  125252  001124          MOV     #125252,$GDDAT          ;LOAD EXPECTED VALUE
004244  013777  001124  175056          MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT
004252  005177  175052                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004256  005177  175046                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004262  005177  175042                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004266  005177  175036                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004272  005177  175032                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004276  005177  175026                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004302  005177  175022                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004306  005177  175016                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004312  017737  175012  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
004320  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST REGISTER
004326  001401                          BEQ     TST12           ;;BR IF CORRECT
004330  104003                          ERROR   3
```

```
                                        ;;*****************************************************************
                                        ;*TEST 12       TEST FOR SLOW OUTPUT GATES WITH #52525
                                        ;;*****************************************************************
004332  000004                  TST12:  SCOPE
004334  012737  052525  001124          MOV     #52525,$GDDAT           ;LOAD EXPECTED VALUE
004342  013777  001124  174760          MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT REG.
004350  005177  174754                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004354  005177  174750                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004360  005177  174744                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004364  005177  174740                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004370  005177  174734                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004374  005177  174730                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004400  005177  174724                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004404  005177  174720                  COM     @GRDIO                  ;COMPLEMENT OUTPUT REG.
004410  017737  174714  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
004416  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST PATTERN
004424  001401                          BEQ     TST13           ;;BR IF EQUAL
004426  104003                          ERROR   3               ;OUTPUT REGISTER IN ERROR

                                        ;;*****************************************************************
                                        ;*TEST 13       TEST THAT OUTPUT CAN HOLD LOW BYTE COUNT PATTERN
                                        ;;*****************************************************************
004430  000004                  TST13:  SCOPE
004432  012737  000400  001166          MOV     #400,$TIMES     ;;DO 400 ITERATIONS
004440  012737  004462  001110          MOV     #2$,$LPERR              ;LOAD SCOPE ERROR RETURN
004446  012737  052400  001124          MOV     #52400,$GDDAT           ;LOAD EXPECTED DATA
004454  013777  001124  174646          MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT REG.
004462  113777  001124  174640  2$:     MOVB    $GDDAT,@GRDIO           ;LOAD THE OUTPUT
004470  017737  174634  001126          MOV     @GRDIO,$BDDAT           ;READ OUTPUT REG.
004476  023737  001124  001126          CMP     $GDDAT,$BDDAT
004504  001401                          BEQ     1$                      ;BRANCH IF EQUAL
004506  104003                          ERROR   3                       ;ERROR, OUTPUT REG. IN ERROR
004510  105237  001124          1$:     INCB    $GDDAT                  ;UPDATE PATTERN
004514  001362                          BNE     2$

                                        ;;*****************************************************************
                                        ;*TEST 14       TEST THAT OUTPUT CAN HOLD HIGH BYTE COUNT PATTERN
                                        ;;*****************************************************************
004516  000004                  TST14:  SCOPE
004520  012737  000400  001166          MOV     #400,$TIMES     ;;DO 400 ITERATIONS
004526  012737  004550  001110          MOV     #2$,$LPERR              ;LOAD SCOPE ERROR RETURN
004534  012737  000252  001124          MOV     #252,$GDDAT             ;LOAD EXPECTED
004542  013777  001124  174560          MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT REG.
004550  113777  001125  174554  2$:     MOVB    $GDDAT+1,@GRBHIO        ;LOAD THE HIGH BYTE
004556  017737  174546  001126          MOV     @GRDIO,$BDDAT           ;READ  OUTPUT REG.
004564  023737  001124  001126          CMP     $GDDAT,$BDDAT
004572  001401                          BEQ     1$                      ;BRANCH IF EQUAL
004574  104003                          ERROR   3                       ;ERROR, OUTPUT REG. IN ERROR
004576  105237  001125          1$:     INCB    $GDDAT+1
004602  001362                          BNE     2$
```

```
                                ;;*****************************************************************
                                ;*TEST 15        TEST OUTPUT DATA ACCEPT FLAG
                                ;;*****************************************************************
004604  000004                TST15:  SCOPE
004606  005077  174516                  CLR     @GRDIO
004612  012777  177777  174506        MOV     #-1,@GRDAI
004620  012737  100000  001124        MOV     #BIT15,$GDDAT           ;LOAD EXPECTED
004626  012777  100000  174470        MOV     #BIT15,@GRSTAT          ;SET BIT 15
004634  017737  174464  001126        MOV     @GRSTAT,$BDDAT          ;READ STATUS
004642  033737  001124  001126        BIT     $GDDAT,$BDDAT
004650  001001                      BNE     TST16           ;;BR IF SET
004652  104001                      ERROR   1                       ;ERROR, BIT 15 FAILED TO SET

                                ;;*****************************************************************
                                ;*TEST 16        TEST OUTPUT INTERRUPT ENABLE
                                ;;*****************************************************************
004654  000004                TST16:  SCOPE
004656  005077  174442                  CLR     @GRSTAT                 ;CLEAR STATUS
004662  012737  040000  001124        MOV     #BIT14,$GDDAT           ;LOAD EXPECTED
004670  012777  040000  174426        MOV     #BIT14,@GRSTAT          ;LOAD BIT 14
004676  017737  174422  001126        MOV     @GRSTAT,$BDDAT          ;READ STATUS
004704  033737  001124  001126        BIT     $GDDAT,$BDDAT
004712  001001                      BNE     TST17           ;;BR IF SET
004714  104001                      ERROR   1                       ;ERROR BIT 14 FAILED TO SET

                                ;;*****************************************************************
                                ;*TEST 17        TEST INPUT DATA READY FLAG
                                ;;*****************************************************************
004716  000004                TST17:  SCOPE
004720  012737  000200  001124        MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
004726  012777  000200  174370        MOV     #BIT7,@GRSTAT           ;SET BIT 7
004734  017737  174364  001126        MOV     @GRSTAT,$BDDAT          ;READ RESULT
004742  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE
004750  001401                      BEQ     TST20           ;;BR IF EQUAL
004752  104001                      ERROR   1                       ;ERROR, BIT 7 FAILED TO SET

                                ;;*****************************************************************
                                ;*TEST 20        TEST INPUT INTERRUPT ENABLE
                                ;;*****************************************************************
004754  000004                TST20:  SCOPE
004756  005077  174342                  CLR     @GRSTAT                 ;CLEAR STATUS
004762  012737  000100  001124        MOV     #BIT6,$GDDAT            ;LOAD EXPECTED
004770  012777  000100  174326        MOV     #BIT6,@GRSTAT           ;SET BIT 6
004776  017737  174322  001126        MOV     @GRSTAT,$BDDAT          ;READ RESULT
005004  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE
005012  001401                      BEQ     TST21           ;;BR IF EQUAL
005014  104001                      ERROR   1                       ;ERROR, BIT 6 FAILED TO SET
```

```
                                        ;;*****************************************************************
                                        ;*TEST 21       TEST THAT RESET CLEARS THE DIGITAL STATUS REGISTER
                                        ;;*****************************************************************
005016  000004                  TST21:  SCOPE
005020  012737  000040  001166          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
005026  005077  174272                  CLR     @GRSTAT                 ;CLEAR STATUS
005032  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
005036  012777  140300  174260          MOV     #BIT15!BIT14!BIT7!BIT6,@GRSTAT
005044  000005                          RESET
005046  017737  174252  001126          MOV     @GRSTAT,$BDDAT          ;READ RESULT
005054  001401                          BEQ     TST22           ;;BR IF EQUAL
005056  104001                          ERROR   1                       ;ERROR, RESET FAILED TO CLEAR DIGITAL
                                                                        ;STATUS REGISTER


                                        ;;*****************************************************************
                                        ;*TEST 22       TEST EXTERNAL TRANSFERS - CABLE MUST BE CONNECTED
                                        ;;*****************************************************************
005060  000004                  TST22:  SCOPE
005062  032777  002000  174050          BIT     #BIT10,@SWR             ;TEST SWITCH BIT
005070  001402                          BEQ     1$                      ;BRANCH IF DOWN
005072  000137  007406                  JMP     DRT21                   ;BYPASS SOME TEST USING THE EXTERNAL CABLE
005076                          1$:
005076  005077  174226                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
005102  012777  177777  174216          MOV     #-1,@GRDAI              ;CLEAR INPUT
005110  005037  001124                  CLR     $GDDAT                  ;CLEAR EXPECTED
005114  012777  000000  174206          MOV     #0,@GRDIO               ;LOAD THE OUTPUT
005122  017737  174200  001126          MOV     @GRDAI,$BDDAT           ;READ THE INPUT
005130  043737  001360  001126          BIC     ODDJMP,$BDDAT           ;MASK
005136  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005144  001401                          BEQ     TST23           ;;BR IF EQUAL
005146  104002                          ERROR   2                       ;ERROR, INPUT DID NOT EQUAL THE OUTPUT REG.


                                        ;;*****************************************************************
                                        ;*TEST 23       TEST INPUT WITH #-1
                                        ;;*****************************************************************
005150  000004                  TST23:  SCOPE
005152  005077  174152                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
005156  012777  177777  174142          MOV     #-1,@GRDAI              ;CLEAR INPUT
005164  012737  177777  001124          MOV     #-1,$GDDAT              ;LOAD EXPECTED
005172  043737  001360  001124          BIC     ODDJMP,$GDDAT           ;MASK
005200  013777  001124  174122          MOV     $GDDAT,@GRDIO           ;LOAD THE OUTPUT
005206  017737  174114  001126          MOV     @GRDAI,$BDDAT           ;READ INPUT
005214  043737  001360  001126          BIC     ODDJMP,$BDDAT           ;MASK
005222  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005230  001401                          BEQ     TST24           ;;BR IF EQUAL
005232  104002                          ERROR   2                       ;ERROR, INPUT DID NOT EQUAL THE OUTPUT
```

```
1023                                    ;;*****************************************************************
1024                                    ;*TEST 24       TEST INPUT WITH #52525
1025                                    ;;*****************************************************************
1026  005234  000004                   TST24:  SCOPE
1027  005236  005077  174066                   CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1028  005242  012777  177777  174056           MOV     #-1,@GRDAI              ;CLEAR INPUT
1029  005250  012737  052525  001124           MOV     #52525,$GDDAT           ;LOAD EXPECTED
1030  005256  043737  001360  001124           BIC     ODDJMP,$GDDAT           ;MASK
1031  005264  013777  001124  174036           MOV     $GDDAT,@GRDIO           ;LOAD THE OUTPUT
1032  005272  017737  174030  001126           MOV     @GRDAI,$BDDAT           ;READ INPUT
1033  005300  043737  001360  001126           BIC     ODDJMP,$BDDAT           ;MASK
1034  005306  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE
1035  005314  001401                           BEQ     TST25           ;;BR IF EQUAL
1036  005316  104002                           ERROR   2                       ;ERROR, INPUT DID NOT EQUAL OUTPUT
1037
1038                                    ;;*****************************************************************
1039                                    ;*TEST 25       TEST INPUT WITH #125252
1040                                    ;;*****************************************************************
1041  005320  000004                   TST25:  SCOPE
1042  005322  005077  174002                   CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1043  005326  012777  177777  173772           MOV     #-1,@GRDAI              ;CLEAR INPUT
1044  005334  012737  125252  001124           MOV     #125252,$GDDAT          ;LOAD EXPECTED
1045  005342  043737  001360  001124           BIC     ODDJMP,$GDDAT           ;MASK
1046  005350  013777  001124  173752           MOV     $GDDAT,@GRDIO           ;LOAD THE OUTPUT
1047  005356  017737  173744  001126           MOV     @GRDAI,$BDDAT           ;READ INPUT
1048  005364  043737  001360  001126           BIC     ODDJMP,$BDDAT           ;MASK
1049  005372  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE
1050  005400  001401                           BEQ     TST26           ;;BR IF EQUAL
1051  005402  104002                           ERROR   2                       ;ERROR, INPUT DID NOT EQUAL OUTPUT
```

```
                                        ;;*****************************************************************
                                        ;*TEST 26       TEST THE NEG. AND TRANSITION LATCHING INPUT DATA BITS
                                        ;;*****************************************************************
005404  000004                  TST26:  SCOPE
005406  012737  000001  001166          MOV     #1,$TIMES       ;;DO 1 ITERATION
005414  005737  001360                  TST     ODDJMP                  ;TEST IF ANY ODD JUMPERS
005420  001461                          BEQ     TST27           ;;BR IF NONE
005422  012737  010000  001124          MOV     #BIT12,$GDDAT           ;LOAD TEST BIT
005430  033737  001360  001124  1$:     BIT     ODDJMP,$GDDAT           ;TEST IF ODD JUMPER BIT
005436  001447                          BEQ     2$                      ;BR IF NOT
005440  033737  001334  001124          BIT     NOTLCH,$GDDAT           ;TEST IF LATCHING INPUT BIT
005446  001043                          BNE     2$                      ;BR IF NOT
005450  013777  001124  173652          MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT
005456  012777  177777  173642          MOV     #-1,@GRDAI              ;CLEAR INPUT
005464  043777  001124  173636          BIC     $GDDAT,@GRDIO           ;CLEAR OUTPUT BIT
005472  017737  173630  001126          MOV     @GRDAI,$BDDAT           ;READ INPUT REG
005500  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005506  001401                          BEQ     3$                      ;BR IF EQUAL
005510  104002                          ERROR   2                       ;ERROR, NEG. INPUT OR NEG. TRANSITION
                                                                        ; INPUT BIT FAILED TO SET INPUT REGISTER
005512  033737  001124  001362  3$:     BIT     $GDDAT,MINSIN           ;TEST IF NEG. INPUT BIT
005520  001016                          BNE     2$                      ;BR IF
005522  012777  177777  173576          MOV     #-1,@GRDAI              ;CLEAR INPUT
005530  053777  001124  173572          BIS     $GDDAT,@GRDIO           ;LOAD OUTPUT
005536  017737  173564  001126          MOV     @GRDAI,$BDDAT           ;READ INPUT
005544  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
005552  001401                          BEQ     2$                      ;BR IF EQUAL
005554  104002                          ERROR   2                       ;ERROR, POSITIVE INPUT TRANSITION
                                                                        ;LOGIC FAILED TO SET INPUT REGISTER BIT
005556  006137  001124          2$:     ROL     $GDDAT                  ;CHANGE DATA PATTERN
005562  103322                          BCC     1$                      ;BR IF MORE DATA
```

```
1095                                    ;;*****************************************************************
1096                                    ;*TEST 27       FLOAT A 1 ACROSS NON-LATCHING INPUT BITS
1097                                    ;;*****************************************************************
1098  005564  000004                    TST27:  SCOPE
1099  005566  012737  005614  001110            MOV     #2$,$LPERR              ;LOAD ERROR SCOPE RETURN
1100
1101  005574  012737  000001  001354            MOV     #BIT0,BRLEV2            ;LOAD EXPECTED
1102  005602  013737  001334  001356            MOV     NOTLCH,BRLEV3           ;GET NON-LATCH
1103  005610  005137  001356                    COM     BRLEV3                  ;COMPLEMENT
1104  005614  013737  001354  001124  2$:       MOV     BRLEV2,$GDDAT           ;LOAD GOOD
1105  005622  033737  001124  001360            BIT     $GDDAT,ODDJMP           ;TEST IF ODD JUMPER
1106  005630  001042                            BNE     1$                      ;BYPASS IF ODD JUMPER
1107  005632  033737  001124  001334            BIT     $GDDAT,NOTLCH           ;TEST FOR NON-LATCH
1108  005640  001436                            BEQ     1$                      ;BR IF LATCHING
1109
1110  005642  013777  001124  173460            MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT
1111  005650  017737  173452  001126            MOV     @GRDAI,$BDDAT           ;READ INPUT
1112  005656  043737  001356  001126            BIC     BRLEV3,$BDDAT           ;MASK TO LATCH BITS
1113  005664  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1114  005672  001401                            BEQ     3$              ;;BR IF EQUAL
1115  005674  104002                            ERROR   2                       ;INPUT REGISTER IN ERROR
                                                                                ;WAS CORRECT LATCH/NON-LATCH SUPPLIED ?
1116
1117
1118                                    ;SUB-TEST CLEAR THE OUTPUT BIT AND TEST THE INPUT DOES NOT LATCH
1119
1120  005676  043777  001124  173424  3$:       BIC     $GDDAT,@GRDIO           ;CLEAR OUTPUT BIT
1121  005704  017737  173416  001126            MOV     @GRDAI,$BDDAT           ;READ INPUT
1122  005712  043737  001356  001126            BIC     BRLEV3,$BDDAT           ;MASK TO LATCH BITS
1123  005720  005037  001124                    CLR     $GDDAT                  ;CLEAR EXPECTED
1124  005724  033737  001354  001126            BIT     BRLEV2,$BDDAT           ;TEST FOR BIT
1125  005732  001401                            BEQ     1$              ;;BR IF CLEARED
1126  005734  104002                            ERROR   2                       ;INPUT BIT LATCHED IN ERROR
                                                                                ;WAS CORRECT LATCH/NON-LATCH SUPPLIED ?
1127
1128
1129  005736  006337  001354          1$:       ASL     BRLEV2                  ;CHANGE PATTERN
1130  005742  001324                            BNE     2$                      ;BR UNTIL DONE
1131
```

```
1122                                    ;;*****************************************************************
1123                                    ;*TEST 30       FLOAT A 1 ACROSS LATCHING INPUT BITS
1124                                    ;;*****************************************************************
1125  005744  000004                    TST30:  SCOPE
1126  005746  012737  005774  001110            MOV     #2$,$LPERR              ;LOAD ERROR SCOPE RETURN
1127  005754  012737  000001  001124            MOV     #BIT0,$GDDAT            ;LOAD EXPECTED VALUE
1128  005762  005077  173342                    CLR     @GRDIO                  ;CLEAR OUTPUT REG
1129  005766  012777  177777  173332            MOV     #-1,@GRDAI              ;CLEAR INPUT
1130
1131  005774  033737  001124  001334  2$:      BIT     $GDDAT,NOTLCH           ;TEST FOR NON-LATCHING
1132  006002  001021                           BNE     1$                      ;BR IF NON-LATCH
1133  006004  033737  001124  001360           BIT     $GDDAT,ODDJMP           ;TEST IF ODD JUMPER
1134  006012  001015                           BNE     1$                      ;BYPASS IF ODD JUMPER
1135
1136  006014  013777  001124  173306           MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT
1137  006022  005077  173302                   CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1138  006026  017737  173274  001126           MOV     @GRDAI,$BDDAT           ;READ INPUT REG.
1139  006034  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE
1140  006042  001401                           BEQ     1$              ;;BR IF EQUAL
1141  006044  104002                           ERROR   2                       ;INPUT REGISTER FAILED TO LATCH DATA
1142
1143  006046  053777  001124  173252  1$:      BIS     $GDDAT,@GRDAI           ;CLEAR INPUT BIT
1144  006054  006337  001124                   ASL     $GDDAT                  ;CHANGE PATTERN
1145  006060  001345                           BNE     2$                      ;BR UNTIL DONE
1146
1147                                    ;;*****************************************************************
1148                                    ;*TEST 31       FLOAT A 0 ACROSS LATCHING INPUT BITS
1149                                    ;;*****************************************************************
1150  006062  000004                    TST31:  SCOPE
1151  006064  012737  006112  001110            MOV     #2$,$LPERR              ;LOAD ERROR SCOPE RETURN
1152  006072  012737  000001  001356            MOV     #BIT0,BRLEV3            ;LOAD EXPECTED
1153  006100  005077  173224                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1154  006104  012777  177777  173214            MOV     #-1,@GRDAI              ;CLEAR INPUT
1155
1156  006112  033737  001356  001334  2$:      BIT     BRLEV3,NOTLCH           ;TEST FOR LATCHING
1157  006120  001032                           BNE     1$                      ;BR IF NOT
1158  006122  033737  001124  001360           BIT     $GDDAT,ODDJMP           ;TEST IF ODD JUMPER
1159  006130  001026                           BNE     1$                      ;BYPASS IF ODD JUMPER BIT
1160
1161  006132  012737  177777  001124           MOV     #-1,$GDDAT              ;LOAD
1162  006140  043737  001334  001124           BIC     NOTLCH,$GDDAT
1163  006146  043737  001356  001124           BIC     BRLEV3,$GDDAT           ;MAKE BRLEV3
1164  006154  013777  001124  173146           MOV     $GDDAT,@GRDIO           ;LOAD OUTPUT
1165  006162  005077  173142                   CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1166
1167  006166  017737  173134  001126           MOV     @GRDAI,$BDDAT           ;READ INPUT
1168  006174  023737  001124  001126           CMP     $GDDAT,$BDDAT           ;COMPARE
1169  006202  001401                           BEQ     1$              ;;BR IF EQUAL
1170  006204  104002                           ERROR   2                       ;INPUT REGISTER FAILED TO LATCH DATA
1171
1172  006206  053777  001356  173112  1$:      BIS     BRLEV3,@GRDAI           ;CLEAR INPUT BIT
1173  006214  006337  001356                   ASL     BRLEV3                  ;CHANGE PATTERN
1174  006220  001334                           BNE     2$                      ;BR UNTIL DONE
```

```
1175                                    ;;*****************************************************************
1176                                    ;*TEST 32       TEST FOR SLOW INPUT GATES WITH #125252
1177                                    ;;*****************************************************************
1178  006222  000004                    TST32:  SCOPE
1179
1180  006224  012737  125252  001124            MOV     #125252,$GDDAT          ;LOAD EXPECTED
1181  006232  043737  001334  001124            BIC     NOTLCH,$GDDAT           ;CONVERT
1182  006240  043737  001360  001124            BIC     ODDJMP,$GDDAT           ;MASK
1183  006246  013700  001124                    MOV     $GDDAT,R0               ;LOAD PATTERN
1184  006252  005077  173052                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1185  006256  012777  177777  173042          MOV     #-1,@GRDAI              ;CLEAR INPUT
1186
1187  006264  010077  173040                    MOV     R0,@GRDIO               ;LOAD OUTPUT
1188  006270  005077  173034                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1189  006274  017701  173026                    MOV     @GRDAI,R1               ;READ INPUT
1190  006300  050177  173022                    BIS     R1,@GRDAI               ;CLEAR INPUT
1191  006304  005100                            COM     R0
1192  006306  010077  173016                    MOV     R0,@GRDIO               ;LOAD OUTPUT
1193  006312  005077  173012                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1194  006316  017701  173004                    MOV     @GRDAI,R1               ;READ INPUT
1195  006322  050177  173000                    BIS     R1,@GRDAI               ;CLEAR INPUT
1196  006326  005100                            COM     R0
1197  006330  010077  172774                    MOV     R0,@GRDIO               ;LOAD OUTPUT
1198  006334  005077  172770                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1199  006340  017701  172762                    MOV     @GRDAI,R1               ;READ INPUT
1200  006344  050177  172756                    BIS     R1,@GRDAI               ;CLEAR INPUT
1201  006350  005100                            COM     R0
1202  006352  010077  172752                    MOV     R0,@GRDIO               ;LOAD OUTPUT
1203  006356  005077  172746                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1204  006362  017701  172740                    MOV     @GRDAI,R1               ;READ INPUT
1205  006366  050177  172734                    BIS     R1,@GRDAI               ;CLEAR INPUT
1206  006372  005100                            COM     R0
1207  006374  010077  172730                    MOV     R0,@GRDIO               ;LOAD OUTPUT
1208  006400  005077  172724                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1209  006404  017701  172716                    MOV     @GRDAI,R1               ;READ INPUT
1210  006410  050177  172712                    BIS     R1,@GRDAI               ;CLEAR INPUT
1211  006414  005100                            COM     R0
1212  006416  010077  172706                    MOV     R0,@GRDIO               ;LOAD OUTPUT
1213  006422  005077  172702                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1214  006426  017701  172674                    MOV     @GRDAI,R1               ;READ INPUT
1215  006432  050177  172670                    BIS     R1,@GRDAI               ;CLEAR INPUT
1216  006436  005100                            COM     R0
1217  006440  010077  172664                    MOV     R0,@GRDIO               ;LOAD OUTPUT
1218  006444  005077  172660                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1219  006450  017701  172652                    MOV     @GRDAI,R1               ;READ INPUT
1220  006454  050177  172646                    BIS     R1,@GRDAI               ;CLEAR INPUT
1221  006460  005100                            COM     R0
1222  006462  010077  172642                    MOV     R0,@GRDIO               ;LOAD OUTPUT
1223  006466  005077  172636                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1224  006472  017701  172630                    MOV     @GRDAI,R1               ;READ INPUT
1225  006476  050177  172624                    BIS     R1,@GRDAI               ;CLEAR INPUT
1226  006502  005100                            COM     R0
1227  006504  010077  172620                    MOV     R0,@GRDIO               ;LOAD OUTPUT
1228  006510  005077  172614                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1229  006514  017701  172606                    MOV     @GRDAI,R1               ;READ INPUT
1230  006520  050177  172602                    BIS     R1,@GRDAI               ;CLEAR INPUT
```

```
1231  006524  005100                          COM     R0
1232  006526  010077  172576                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1233  006532  005077  172572                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1234  006536  017701  172564                  MOV     @GRDAI,R1               ;READ INPUT
1235  006542  050177  172560                  BIS     R1,@GRDAI               ;CLEAR INPUT
1236  006546  005100                          COM     R0
1237  006550  010077  172554                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1238  006554  005077  172550                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1239  006560  017701  172542                  MOV     @GRDAI,R1               ;READ INPUT
1240  006564  050177  172536                  BIS     R1,@GRDAI               ;CLEAR INPUT
1241  006570  005100                          COM     R0
1242
1243  006572  010137  001126                  MOV     R1,$BDDAT               ;LOAD READ
1244  006576  000240                          NOP
1245  006600  000240                          NOP
1246  006602  000240                          NOP
1247  006604  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE
1248  006612  001401                          BEQ     TST33           ;;BR IF EQUAL
1249  006614  104002                          ERROR   2                       ;INPUT GATE SLOW
1250
1251                                  ;;*****************************************************************
1252                                  ;*TEST 33       TEST FOR SLOW INPUT GATES WITH #52525
1253                                  ;;*****************************************************************
1254  006616  000004                  TST33:  SCOPE
1255
1256  006620  012737  052525  001124          MOV     #52525,$GDDAT           ;SETUP EXPECTED
1257  006626  043737  001360  001124          BIC     ODDJMP,$GDDAT           ;MASK ODD JUMPER BITS
1258  006634  043737  001334  001124          BIC     NOTLCH,$GDDAT           ;CONVERT
1259  006642  013700  001124                  MOV     $GDDAT,R0
1260  006646  005077  172456                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1261  006652  012777  177777  172446          MOV     #-1,@GRDAI              ;CLEAR INPUT
1262
1263  006660  010077  172444                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1264  006664  005077  172440                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1265  006670  017701  172432                  MOV     @GRDAI,R1               ;READ INPUT
1266  006674  050177  172426                  BIS     R1,@GRDAI               ;CLEAR INPUT
1267  006700  005100                          COM     R0
1268  006702  010077  172422                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1269  006706  005077  172416                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1270  006712  017701  172410                  MOV     @GRDAI,R1               ;READ INPUT
1271  006716  050177  172404                  BIS     R1,@GRDAI               ;CLEAR INPUT
1272  006722  005100                          COM     R0
1273  006724  010077  172400                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1274  006730  005077  172374                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1275  006734  017701  172366                  MOV     @GRDAI,R1               ;READ INPUT
1276  006740  050177  172362                  BIS     R1,@GRDAI               ;CLEAR INPUT
1277  006744  005100                          COM     R0
1278  006746  010077  172356                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1279  006752  005077  172352                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1280  006756  017701  172344                  MOV     @GRDAI,R1               ;READ INPUT
1281  006762  050177  172340                  BIS     R1,@GRDAI               ;CLEAR INPUT
1282  006766  005100                          COM     R0
1283  006770  010077  172334                  MOV     R0,@GRDIO               ;LOAD OUTPUT
1284  006774  005077  172330                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1285  007000  017701  172322                  MOV     @GRDAI,R1               ;READ INPUT
1286  007004  050177  172316                  BIS     R1,@GRDAI               ;CLEAR INPUT
```

```
1287  007010  005100                        COM     R0
1288  007012  010077  172312                MOV     R0,@GRDIO               ;LOAD OUTPUT
1289  007016  005077  172306                CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1290  007022  017701  172300                MOV     @GRDAI,R1               ;READ INPUT
1291  007026  050177  172274                BIS     R1,@GRDAI               ;CLEAR INPUT
1292  007032  005100                        COM     R0
1293  007034  010077  172270                MOV     R0,@GRDIO               ;LOAD OUTPUT
1294  007040  005077  172264                CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1295  007044  017701  172256                MOV     @GRDAI,R1               ;READ INPUT
1296  007050  050177  172252                BIS     R1,@GRDAI               ;CLEAR INPUT
1297  007054  005100                        COM     R0
1298  007056  010077  172246                MOV     R0,@GRDIO               ;LOAD OUTPUT
1299  007062  005077  172242                CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1300  007066  017701  172234                MOV     @GRDAI,R1               ;READ INPUT
1301  007072  050177  172230                BIS     R1,@GRDAI               ;CLEAR INPUT
1302  007076  005100                        COM     R0
1303  007100  010077  172224                MOV     R0,@GRDIO               ;LOAD OUTPUT
1304  007104  005077  172220                CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1305  007110  017701  172212                MOV     @GRDAI,R1               ;READ INPUT
1306  007114  050177  172206                BIS     R1,@GRDAI               ;CLEAR INPUT
1307  007120  005100                        COM     R0
1308  007122  010077  172202                MOV     R0,@GRDIO               ;LOAD OUTPUT
1309  007126  005077  172176                CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1310  007132  017701  172170                MOV     @GRDAI,R1               ;READ INPUT
1311  007136  050177  172164                BIS     R1,@GRDAI               ;CLEAR INPUT
1312  007142  005100                        COM     R0
1313  007144  010077  172160                MOV     R0,@GRDIO               ;LOAD OUTPUT
1314  007150  005077  172154                CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1315  007154  017701  172146                MOV     @GRDAI,R1               ;READ INPUT
1316  007160  050177  172142                BIS     R1,@GRDAI               ;CLEAR INPUT
1317  007164  005100                        COM     R0
1318
1319  007166  010137  001126                MOV     R1,$BDDAT               ;LOAD VALUE READ
1320  007172  000240                        NOP
1321  007174  000240                        NOP
1322  007176  000240                        NOP
1323  007200  023737  001124  001126        CMP     $GDDAT,$BDDAT           ;COMPARE
1324  007206  001401                        BEQ     TST34           ;;BR IF EQUAL
1325  007210  104002                        ERROR   2
```

```
1326                                    ;;*****************************************************************
1327                                    ;*TEST 34       TEST THAT RESET CLEARS INPUT REGISTER BITS
1328                                    ;;*****************************************************************
1329  007212  000004                    TST34:  SCOPE
1330  007214  012737  000040  001166            MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1331  007222  005077  172102                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1332  007226  012777  177777  172072            MOV     #-1,@GRDAI              ;CLEAR INPUT
1333  007234  012777  177777  172066            MOV     #-1,@GRDIO              ;LOAD OUTPUT
1334  007242  005037  001124                    CLR     $GDDAT                  ;LOAD EXPECTED
1335  007246  000005                            RESET
1336  007250  017737  172052  001126            MOV     @GRDAI,$BDDAT           ;READ INPUT REG.
1337  007256  043737  001360  001126            BIC     ODDJMP,$BDDAT           ;MASK ODD JUMPERS
1338  007264  001401                            BEQ     TST35           ;;BR IF ALL BITS CLEARED
1339  007266  104002                            ERROR   2                       ;INPUT REG. FAILED TO CLEAR UPON RESET INST.
1340
1341                                    ;;*****************************************************************
1342                                    ;*TEST 35       TEST THAT WHEN OUTPUTTING THE INPUT DATA READY FLAG SETS
1343                                    ;;*****************************************************************
1344  007270  000004                    TST35:  SCOPE
1345  007272  012737  000200  001124            MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
1346  007300  005077  172020                    CLR     @GRSTAT
1347  007304  012777  000000  172016            MOV     #0,@GRDIO               ;OUTPUT 0
1348  007312  022727  000000  000000            CMP     #0,#0                   ;DELAY
1349  007320  017737  172000  001126            MOV     @GRSTAT,$BDDAT          ;READ RESULTS
1350  007326  023737  001124  001126            CMP     $GDDAT,$BDDAT           ;COMPARE
1351  007334  001401                            BEQ     TST36           ;;BR IF EQUAL
1352  007336  104001                            ERROR   1                       ;INPUT DATA READY FLAG FAILED TO SET
1353
1354                                    ;TEST THAT WHEN THE INPUT BUFFER IS READ THE OUTPUT FLAG IS SET
1355
1356                                    ;;*****************************************************************
1357                                    ;*TEST 36       TEST THAT WHEN THE INPUT BUFFER IS READ THE OUTPUT FLAG IS SET
1358                                    ;;*****************************************************************
1359  007340  000004                    TST36:  SCOPE
1360  007342  012737  100000  001124            MOV     #BIT15,$GDDAT           ;LOAD EXPECTED
1361  007350  005077  171750                    CLR     @GRSTAT
1362  007354  012777  000000  171746            MOV     #0,@GRDIO
1363  007362  022727  000000  000000            CMP     #0,#0
1364  007370  017700  171732                    MOV     @GRDAI,R0               ;READ INPUT
1365  007374  017737  171724  001126            MOV     @GRSTAT,$BDDAT          ;READ RESULTS
1366  007402  100401                            BMI     TST37           ;;BR IF SET
1367  007404  104001                            ERROR   1                       ;INPUT DATA READY FLAG FAILED TO SET
1368
```

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

007406                                  DRT21:
                                        ;;*****************************************************************
                                        ;*TEST 37        TEST FOR UNEXPECTED INTERRUPT
                                        ;;*****************************************************************
007406  000004                          TST37:  SCOPE
007410  012737  000040  001166                  MOV     #40,$TIMES      ;;DO 40 ITERATIONS
007416  000005                                  RESET
007420  005077  171700                          CLR     @GRSTAT
007424  005037  177776                          CLR     PSW
007430  012777  007510  171702                  MOV     #1$,@GRIVA              ;SET UP INTERRUPT INPUT VECTOR
007436  012777  000340  171676                  MOV     #340,@GRIVSA
007444  012777  007514  171672                  MOV     #2$,@GRIVB              ;SET UP INTERRUPT OUTPUT VECTOR
007452  012777  000340  171666                  MOV     #340,@GRIVSB
007460  012777  000000  171642                  MOV     #0,@GRDIO               ;OUTPUT
007466  017700  171634                          MOV     @GRDAI,R0               ;INPUT
007472  000240                                  NOP
007474  000240                                  NOP
007476  000240                                  NOP
007500  000240                                  NOP
007502  005077  171616                          CLR     @GRSTAT                 ;CLEAR STATUS
007506  000404                                  BR      TST40           ;;

007510  104006                          1$:     ERROR   6                       ;ERROR, DIGITAL INPUT INTERRUPTED
007512  000002                                  RTI

007514  104006                          2$:     ERROR   6                       ;ERROR, DIGITAL OUTPUT INTERRUPTED
007516  000002                                  RTI
```

```
1399                                    ;;*****************************************************************
1400                                    ;*TEST 40       TEST THAT THE INPUT CAN INTR. USING THE MAINT. BIT
1401                                    ;;*****************************************************************
1402  007520  000004                    TST40:  SCOPE
1403  007522  000240                            NOP
1404  007524  000240                            NOP
1405  007526  005077  171572                    CLR     @GRSTAT                 ;CLEAR STATUS
1406  007532  012777  007574  171600            MOV     #1$,@GRIVA              ;LOAD RETURN VECTOR
1407  007540  012777  000340  171574            MOV     #340,@GRIVSA            ;
1408  007546  013777  001346  171570            MOV     GRIVSB,@GRIVB           ;LOAD OUTPUT VECTOR
1409  007554  012777  000000  171564            MOV     #0,@GRIVSB
1410  007562  052777  000040  171534            BIS     #BIT5,@GRSTAT           ;MAINT. INT C
1411  007570  000240                            NOP
1412  007572  104004                            ERROR   4                       ;ERROR, INPUT FAILED TO INTERRUPT
1413
1414                                    ;SUB-TEST, TEST THAT IF PSW IS LOWERED AGAIN NO INTERRUPT WILL OCCUR
1415                                    ; IF INTERRUPT OCCURS 'INT DONE C' FAILED TO CLEAR INT C FLOP
1416
1417  007574  012777  007616  171536  1$:     MOV     #2$,@GRIVA              ;LOAD INPUT VECTOR
1418  007602  022626                            CMP     (SP)+,(SP)+             ;POP THE STACK *4
1419  007604  005037  177776                    CLR     PSW                     ;LOWER PRIOR.
1420  007610  000240                            NOP
1421  007612  000240                            NOP
1422  007614  000404                            BR      TST41           ;;<NEXT TEST>
1423
1424  007616  022626                    2$:     CMP     (SP)+,(SP)+             ;POP THE STACK
1425  007620  104006                            ERROR   6                       ;UNEXPECTED INTERRUPT
1426  007622  005037  177776                    CLR     PSW
```

```
1427                                    ;;*****************************************************************
1428                                    ;*TEST 41       TEST THAT THE INPUT INTR. CLEARS INT. ENABLE VIA MAINT. BIT
1429                                    ;;*****************************************************************
1430  007626  000004                    TST41:  SCOPE
1431  007630  000240                            NOP
1432  007632  000240                            NOP
1433  007634  005077  171464                    CLR     @GRSTAT                 ;CLEAR STATUS
1434  007640  012777  007710  171472            MOV     #1$,@GRIVA              ;LOAD RETURN VECTOR
1435  007646  012777  000340  171466            MOV     #340,@GRIVSA            ;
1436  007654  013777  001346  171462            MOV     GRIVSB,@GRIVB           ;LOAD OUTPUT VECTOR
1437  007662  012777  000000  171456            MOV     #0,@GRIVSB
1438  007670  012777  000100  171426            MOV     #BIT6,@GRSTAT           ;LOAD INPUT INTR. ENABLE
1439  007676  052777  000040  171420            BIS     #BIT5,@GRSTAT           ;MAINT. INT C
1440  007704  000240                            NOP
1441  007706  104004                            ERROR   4                       ;ERROR, INPUT FAILED TO INTERRUPT
1442
1443  007710  012777  007752  171422  1$:       MOV     #2$,@GRIVA              ;LOAD INPUT VECTOR
1444  007716  022626                            CMP     (SP)+,(SP)+             ;POP THE STACK *4
1445  007720  005037  177776                    CLR     PSW                     ;LOWER PRIOR.
1446  007724  005037  001124                    CLR     $GDDAT                  ;CLEAR EXPECTED
1447  007730  017737  171370  001126            MOV     @GRSTAT,$BDDAT          ;READ STATUS
1448  007736  032737  000100  001126            BIT     #BIT6,$BDDAT            ;TEST BIT 6
1449  007744  001406                            BEQ     TST42           ;;BR IF CLEARED
1450  007746  104001                            ERROR   1                       ;INPUT INTR. FAILED TO CLEAR
1451  007750  000404                            BR      TST42           ;;<NEXT TEST>
1452
1453  007752  022626                    2$:     CMP     (SP)+,(SP)+             ;POP THE STACK
1454  007754  104006                            ERROR   6                       ;UNEXPECTED INTERRUPT
1455  007756  005037  177776                    CLR     PSW
```

```
1456                                    ;;*****************************************************************
1457                                    ;*TEST 42       TEST THAT THE OUTPUT CAN INTR. USING THE MAINT. BIT
1458                                    ;;*****************************************************************
1459  007762  000004                    TST42:  SCOPE
1460  007764  000240                            NOP
1461  007766  005077  171332                    CLR     @GRSTAT                 ;CLEAR STATUS
1462  007772  013777  001342  171340            MOV     GRIVSA,@GRIVA           ;LOAD INPUT VECTOR
1463  010000  012777  000000  171334            MOV     #0,@GRIVSA
1464  010006  012777  010034  171330            MOV     #1$,@GRIVB              ;LOAD OUTPUT VECTOR
1465  010014  012777  000340  171324            MOV     #340,@GRIVSB
1466  010022  052777  020000  171274            BIS     #BIT13,@GRSTAT          ;MAINT. INTERRUPT
1467  010030  000240                            NOP
1468  010032  104005                            ERROR   5                       ;OUTPUT FAILED TO INTERRUPT
1469
1470                                    ;SUB-TEST, TEST THAT IF PSW IS LOWERED AGAIN, NO INTERRUPT WILL OCCUR
1471                                    ; IF IT DOES, 'INT DONE D HIGH' FAILED TO CLEAR 'INT D' FLOP
1472
1473  010034  012777  010056  171302  1$:     MOV     #2$,@GRIVB              ;LOAD OUTPUT VECTOR
1474  010042  022626                            CMP     (SP)+,(SP)+             ;POP THE STACK *4
1475  010044  005037  177776                    CLR     PSW
1476  010050  000240                            NOP
1477  010052  000240                            NOP
1478  010054  000404                            BR      TST43           ;;<NEXT TEST>
1479  010056  022626                    2$:     CMP     (SP)+,(SP)+             ;POP THE STACK
1480  010060  104006                            ERROR   6                       ;UNEXPECTED INTERRUPT
1481  010062  005037  177776                    CLR     PSW
1482
```

```
1483                                    ;;*****************************************************************
1484                                    ;*TEST 43       TEST FOR INTR. FROM DRA INPUT
1485                                    ;*****************************************************************
1486  010066  000004                   TST43:  SCOPE
1487  010070  000240                           NOP
1488  010072  000240                           NOP
1489  010074  032777  002000  171036           BIT     #BIT10,@SWR             ;TEST SWITCH
1490  010102  001401                           BEQ     1$
1491  010104  000455                           BR      TST44           ;;
1492  010106  005077  171212           1$:     CLR     @GRSTAT
1493  010112  012777  010202  171220           MOV     #2$,@GRIVA              ;IN CASE OF INTERRUPT
1494  010120  012777  000340  171214           MOV     #340,@GRIVSA
1495  010126  013777  001346  171210           MOV     GRIVSB,@GRIVB
1496  010134  005077  171206                   CLR     @GRIVSB
1497  010140  012777  000100  171156           MOV     #BIT6,@GRSTAT           ;ENABLE INPUT INTR.
1498  010146  012777  000000  171154           MOV     #0,@GRDIO               ;OUTPUT
1499  010154  017700  171146                   MOV     @GRDAI,R0               ;INPUT
1500  010160  000240                           NOP                             ;LET INTERRUPT OCCUR
1501  010162  000240                           NOP
1502  010164  000240                           NOP
1503  010166  042777  000100  171130           BIC     #BIT6,@GRSTAT
1504  010174  000240                           NOP
1505  010176  104004                           ERROR   4                       ;ERROR, INPUT FAILED TO INTERRUPT
1506  010200  000417                           BR      TST44           ;;
1507  010202  022626                   2$:     CMP     (SP)+,(SP)+
1508  010204  005037  177776                   CLR     PSW
1509  010210  005077  171110                   CLR     @GRSTAT                 ;CLEAR STATUS
1510  010214  013777  001342  171116           MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
1511  010222  005077  171114                   CLR     @GRIVSA
1512  010226  013777  001346  171110           MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
1513  010234  005077  171106                   CLR     @GRIVSB
1514
```

```
1515                                    ;;*****************************************************************
1516                                    ;*TEST 44       TEST THAT INTERRUPT INPUT BITS SET INPUT READY FLAG
1517                                    ;;*****************************************************************
1518  010240  000004                    TST44:  SCOPE
1519  010242  032777  002000  170670            BIT     #BIT10,@SWR             ;TEST CABLE SWITCH
1520  010250  001067                            BNE     TST45           ;;BYPASS IF NO I/O CABLE
1521
1522  010252  012737  010266  001110            MOV     #1$,$LPERR              ;LOAD ERROR SCOPE RETURN
1523  010260  012737  000001  001356            MOV     #BIT0,BRLEV3            ;LOAD INTERRUPT BIT
1524  010266  005037  001126          1$:       CLR     $BDDAT                  ;CLEAR BAD DATA
1525  010272  012737  000200  001124            MOV     #BIT7,$GDDAT            ;LOAD GOOD DATA
1526
1527  010300  033737  001356  001336            BIT     BRLEV3,INTBIT           ;TEST IF THIS BIT WILL INTERRUPT
1528  010306  001445                            BEQ     3$              ;;NO TRY NEXT BIT
1529  010310  005077  171014                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1530  010314  012777  177777  171004            MOV     #-1,@GRDAI              ;CLEAR INPUT
1531  010322  005077  170776                    CLR     @GRSTAT                 ;CLEAR STATUS
1532  010326  053777  001356  170774            BIS     BRLEV3,@GRDIO           ;LOAD OUTPUT
1533  010334  043777  001356  170766            BIC     BRLEV3,@GRDIO           ;CLEAR OUTPUT
1534  010342  053777  001356  170760            BIS     BRLEV3,@GRDIO           ;LOAD OUTPUT
1535  010350  005077  170750                    CLR     @GRSTAT                 ;CLEAR FLAG FROM DATA READY
1536                                                                            ;SHOULD REMAIN SET VIA DIRECT SET SIDE
1537                                                                            ;IF INTERRUPT INPUT SWITCH IS ON
1538  010354  105777  170744                    TSTB    @GRSTAT                 ;TEST READY BIT
1539  010360  100401                            BMI     2$              ;;BR IF SET
1540  010362  104010                            ERROR   10                      ;INPUT INTERRUPT BIT FAILED TO SET INPUT READY
1541                                                                            ;?? DID OPERATOR GIVE CORRECT
1542                                                                            ;INPUT INTERRUPT BITS ??
1543
1544  010364  043777  001356  170736  2$:       BIC     BRLEV3,@GRDIO           ;CLEAR OUTPUT
1545  010372  053777  001356  170726            BIS     BRLEV3,@GRDAI           ;CLEAR INPUT
1546  010400  005037  001124                    CLR     $GDDAT                  ;CLEAR EXPECTED
1547  010404  005077  170714                    CLR     @GRSTAT                 ;CLEAR STATUS
1548  010410  117737  170710  001126            MOVB    @GRSTAT,$BDDAT          ;READ STATUS
1549  010416  100001                            BPL     3$              ;;BR IF CLEARED
1550  010420  104001                            ERROR   1                       ;INPUT READY FAILED TO CLEAR
1551
1552  010422  006337  001356          3$:       ASL     BRLEV3                  ;CHANGE BIT
1553  010426  001317                            BNE     1$                      ;BR IF NOT DONE
```

```
1554                                    ;;*****************************************************************
1555                                    ;*TEST 45       TEST THAT NON-INTERRUPT INPUT BITS DO NOT SET INPUT READY FLAG
1556                                    ;;*****************************************************************
1557  010430  000004                    TST45:  SCOPE
1558  010432  032777  002000  170500            BIT     #BIT10,@SWR             ;TEST CABLE SWITCH
1559  010440  001050                            BNE     TST46           ;;BYPASS IF NO I/O CABLE
1560
1561  010442  012737  010456  001110            MOV     #1$,$LPERR              ;LOAD ERROR SCOPE RETURN
1562  010450  012737  000001  001356            MOV     #BIT0,BRLEV3            ;LOAD NON-INTERRUPT BIT
1563  010456  012737  000200  001126  1$:       MOV     #200,$BDDAT             ;LOAD BAD DATA
1564  010464  005037  001124                    CLR     $GDDAT                  ;CLEAR GOOD DATA
1565
1566  010470  033737  001356  001336            BIT     BRLEV3,INTBIT           ;TEST IF THIS BIT WILL INTERRUPT
1567  010476  001026                            BNE     3$              ;;YES SKIP AND TRY NEXT BIT
1568  010500  005077  170624                    CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1569  010504  012777  177777  170614            MOV     #-1,@GRDAI              ;CLEAR INPUT
1570  010512  005077  170606                    CLR     @GRSTAT                 ;CLEAR STATUS
1571  010516  053777  001356  170604            BIS     BRLEV3,@GRDIO           ;LOAD OUTPUT
1572  010524  043777  001356  170576            BIC     BRLEV3,@GRDIO           ;CLEAR OUTPUT
1573  010532  053777  001356  170570            BIS     BRLEV3,@GRDIO           ;SET OUTPUT
1574  010540  005077  170560                    CLR     @GRSTAT                 ;CLEAR FLAG FROM DATA READY
1575                                                                            ;SHOULD REMAIN SET VIA DIRECT SET SIDE
1576  010544  105777  170554                    TSTB    @GRSTAT                 ;TEST READY BIT
1577  010550  100001                            BPL     3$              ;;BR IF CLEAR
1578  010552  104011                            ERROR   11                      ;INPUT NON-INTERRUPT BIT SET INPUT READY
1579                                                                            ;?? DID OPERATOR GIVE CORRECT
1580                                                                            ;INPUT INTERRUPT BITS ??
1581
1582
1583  010554  006337  001356            3$:     ASL     BRLEV3                  ;CHANGE BIT
1584  010560  001336                            BNE     1$                      ;BR IF NOT DONE
```

```
                                        ;;*********************************************************************
                                        ;*TEST 46        TEST FOR INTR. FROM DRA OUTPUT
                                        ;;*********************************************************************
010562  000004                  TST46:  SCOPE
010564  000240                          NOP
010566  000240                          NOP
010570  032777  002000  170342          BIT     #BIT10,@SWR             ;TEST SWITCH
010576  001401                          BEQ     1$
010600  000455                          BR      TST47           ;;
010602  005077  170516          1$:     CLR     @GRSTAT
010606  012777  010676  170530          MOV     #2$,@GRIVB      ;IN CASE OF INTERRUPT
010614  012777  000340  170524          MOV     #340,@GRIVSB
010622  013777  001342  170510          MOV     GRIVSA,@GRIVA
010630  005077  170506                  CLR     @GRIVSA
010634  012777  040000  170462          MOV     #BIT14,@GRSTAT          ;ENABLE OUTPUT INTR.
010642  012777  000000  170460          MOV     #0,@GRDIO               ;OUTPUT
010650  017700  170452                  MOV     @GRDAI,R0               ;INPUT
010654  000240                          NOP                             ;LET INTERRUPT OCCUR
010656  000240                          NOP
010660  000240                          NOP
010662  042777  040000  170434          BIC     #BIT14,@GRSTAT
010670  000240                          NOP
010672  104005                          ERROR   5                       ;ERROR, OUTPUT FAILED TO INTERRUPT
010674  000417                          BR      TST47           ;;
010676  022626                  2$:     CMP     (SP)+,(SP)+
010700  013777  001342  170432          MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
010706  005077  170430                  CLR     @GRIVSA
010712  013777  001346  170424          MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
010720  005077  170422                  CLR     @GRIVSB
010724  005037  177776                  CLR     PSW
010730  005077  170370                  CLR     @GRSTAT                 ;CLEAR STATUS

                                        ;;*********************************************************************
                                        ;*TEST 47        PRE-INTERRUPT SETUP
                                        ;;*********************************************************************
010734  000004                  TST47:  SCOPE
010736  012737  000001  001166          MOV     #1,$TIMES       ;;DO 1 ITERATION
010744  000005                          RESET
010746  042737  177437  001350          BIC     #177437,DIOBRL
010754  001001                          BNE     1$
010756  104007                          ERROR   7                       ;BR LEVEL INDICATED FOR DIGITAL I/O WAS 0
010760  022737  000340  001350  1$:     CMP     #340,DIOBRL
010766  001001                          BNE     2$
010770  104007                          ERROR   7                       ;BR LEVEL INDICATED FOR DIGITAL I/O WAS 7
010772  013737  001350  001352  2$:     MOV     DIOBRL,BRLEV1
011000  162737  000040  001352          SUB     #40,BRLEV1
011006  013737  001350  001354          MOV     DIOBRL,BRLEV2
011014  013737  001350  001356          MOV     DIOBRL,BRLEV3
011022  062737  000040  001356          ADD     #40,BRLEV3
```

```
                                        ;;*****************************************************************
                                        ;*TEST 50        TEST FOR INTR. FROM DRA INPUT ON LEVEL INDICATED -1 VIA MAINT. INT
                                        ;;*****************************************************************
011030  000004                        TST50:  SCOPE
011032  005077  170266                        CLR     @GRSTAT                 ;CLEAR ENABLES
011036  012777  011072  170274               MOV     #1$,@GRIVA              ;SET UP VECTORS
011044  013737  001352  177776               MOV     BRLEV1,PSW              ;CHANGE PSW
011052  052777  000040  170244               BIS     #BIT5,@GRSTAT           ;GENERATE INPUT MAINT. INTERRUPT
011060  000240                               NOP
011062  000240                               NOP
011064  000240                               NOP
011066  000240                               NOP
011070  104004                               ERROR   4                       ;ERROR, NO INTERRUPT FROM DIGITAL I/O INPUT
011072  022626                        1$:     CMP     (SP)+,(SP)+
011074  013777  001342  170236               MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
011102  005077  170234                        CLR     @GRIVSA
011106  013777  001346  170230               MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
011114  005077  170226                        CLR     @GRIVSB
011120  005037  177776                        CLR     PSW                     ;LOWER PSW
                                        ;;*****************************************************************
                                        ;*TEST 51        TEST FOR NO INTR. FROM DRA INPUT ON LEVEL INDICATED  VIA MAINT. INT
                                        ;;*****************************************************************
011124  000004                        TST51:  SCOPE
011126  005077  170172                        CLR     @GRSTAT                 ;CLEAR ENABLES
011132  012777  011204  170200               MOV     #1$,@GRIVA              ;SET UP VECTORS
011140  013737  001354  177776               MOV     BRLEV2,PSW              ;CHANGE PSW
011146  052777  000040  170150               BIS     #BIT5,@GRSTAT           ;GENERATE INPUT MAINT. INTERRUPT
011154  000240                               NOP
011156  000240                               NOP
011160  000240                               NOP
                                        ;SUB-TEST, NOW LOWER THE PSW AND ALLOW THE INTERRUPT
011162  012777  011216  170150               MOV     #2$,@GRIVA              ;RESET VECTOR
011170  005037  177776                        CLR     PSW                     ;LOWER PSW
011174  000240                               NOP
011176  000240                               NOP
011200  000240                               NOP
011202  000403                               BR      3$                      ;ERROR
011204  022626                        1$:     CMP     (SP)+,(SP)+
011206  005037  177776                        CLR     PSW
011212  104006                        3$:     ERROR   6                       ;UNEXPECTED INTERRUPT
011214  000415                               BR      TST52           ;;
011216  022626                        2$:     CMP     (SP)+,(SP)+             ;POP THE STACK
011220  013777  001342  170112               MOV     GRIVSA,@GRIVA           ;RESET INPUT VECTOR
011226  005077  170110                        CLR     @GRIVSA
011232  013777  001346  170104               MOV     GRIVSB,@GRIVB           ;RESET OUTPUT VECTOR
011240  005077  170102                        CLR     @GRIVSB
011244  005037  177776                        CLR     PSW
```

```
                                        ;;*********************************************************************
                                        ;*TEST 52       DETERMINE IF MORE DR11-K'S ARE TO BE TESTED
                                        ;;*********************************************************************
                                TST52:  SCOPE
011250  000004
011252  012737  000001  001166          MOV     #1,$TIMES       ;;DO 1 ITERATION
011260  005737  001314          BYPASS: TST     NBEXT                   ;TEST IF ANY
011264  001415                          BEQ     1$                      ;BR IF NONE
011266  032777  010000  167644          BIT     #SW12,@SWR              ;TEST BIT12 OF SWR
011274  001024                          BNE     BYPAS1                  ;INHIBIT TESTING NEXT DR11-K
011276  162737  000010  001316          SUB     #10,DRADD               ;UPDATE DEVICE ADDRESS
011304  062737  000010  001320          ADD     #10,DRIV                ;UPDATE DEVICE VECTOR
011312  005337  001314                  DEC     NBEXT                   ;ANOTHER ONE ?
011316  000413                          BR      BYPAS1                  ;BR IF ANOTHER
011320  013737  001306  001316  1$:     MOV     BASEBA,DRADD            ;RELOAD ADDRESS
011326  013737  001310  001320          MOV     BASEIV,DRIV             ;RELOAD VECTOR
011334  013737  001312  001314          MOV     NMBEXT,NBEXT            ;RELOAD NUMBER
011342  000137  011356                  JMP     $EOP                    ;DONE
011346  012700  177777          BYPAS1: MOV     #-1,R0
011352  000137  003342                  JMP     RBEG2                   ;TEST ANOTHER UNIT

                                        .SBTTL  END OF PASS ROUTINE

                                        ;;*********************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                        ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO BYPAS1

011356                          $EOP:
011356  000004                          SCOPE
011360  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
011364  005037  001166                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
011370  005237  001100                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
011374  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
011402  005327                          DEC     (PC)+           ;;LOOP?
011404  000001                  $EOPCT: .WORD   1
011406  003022                          BGT     $DOAGN          ;;YES
011410  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
011412  000001                  $ENDCT: .WORD   1
011414  011404                          $EOPCT
011416  104401  011463                  TYPE    ,$ENDMG         ;;TYPE "END PASS #"
011422  013746  001100                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
011426  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
011430  104401  011460                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
011434  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
011440  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
011442  000005                          RESET                   ;;CLEAR THE WORLD
011444  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
011446  000240                          NOP                     ;;SAVE ROOM
011450  000240                          NOP                     ;;FOR
011452  000240                          NOP                     ;;ACT11
011454                          $DOAGN:
011454  000137                          JMP     @(PC)+          ;;RETURN
011456  011346                  $RTNAD: .WORD   BYPAS1
011460    377     377     000   $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
```

```
  1737  011463     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
  1738  011470  050040  051501  020123
  1739  011476  000043
  1740                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
  1741
  1742                                  ;;**********************************************************************
  1743                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
  1744                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
  1745                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
  1746                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
  1747                                  ;*REPLACED WITH SPACES.
  1748                                  ;*CALL:
  1749                                  ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
  1750                                  ;*      TYPDS                   ;;GO TO THE ROUTINE
  1751
  1752  011500                          $TYPDS:
  1753  011500  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  1754  011502  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  1755  011504  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  1756  011506  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  1757  011510  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
  1758  011512  012746  020200                 MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
  1759  011516  016605  000020                 MOV     20(SP),R5       ;;GET THE INPUT NUMBER
  1760  011522  100004                         BPL     1$              ;;BR IF INPUT IS POS.
  1761  011524  005405                         NEG     R5              ;;MAKE THE BINARY NUMBER POS.
  1762  011526  112766  000055  000001        MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
  1763  011534  005000                  1$:    CLR     R0              ;;ZERO THE CONSTANTS INDEX
  1764  011536  012703  011714                 MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
  1765  011542  112723  000040                 MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
  1766  011546  005002                  2$:    CLR     R2              ;;CLEAR THE BCD NUMBER
  1767  011550  016001  011704                 MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
  1768  011554  160105                  3$:    SUB     R1,R5           ;;FORM THIS BCD DIGIT
  1769  011556  002402                         BLT     4$              ;;BR IF DONE
  1770  011560  005202                         INC     R2              ;;INCREASE THE BCD DIGIT BY 1
  1771  011562  000774                         BR      3$
  1772  011564  060105                  4$:    ADD     R1,R5           ;;ADD BACK THE CONSTANT
  1773  011566  005702                         TST     R2              ;;CHECK IF BCD DIGIT=0
  1774  011570  001002                         BNE     5$              ;;FALL THROUGH IF 0
  1775  011572  105716                         TSTB    (SP)            ;;STILL DOING LEADING 0'S?
  1776  011574  100407                         BMI     7$              ;;BR IF YES
  1777  011576  106316                  5$:    ASLB    (SP)            ;;MSD?
  1778  011600  103003                         BCC     6$              ;;BR IF NO
  1779  011602  116663  000001  177777        MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
  1780  011610  052702  000060          6$:    BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
  1781  011614  052702  000040          7$:    BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
  1782  011620  110223                         MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
  1783  011622  005720                         TST     (R0)+           ;;JUST INCREMENTING
  1784  011624  020027  000010                 CMP     R0,#10          ;;CHECK THE TABLE INDEX
  1785  011630  002746                         BLT     2$              ;;GO DO THE NEXT DIGIT
  1786  011632  003002                         BGT     8$              ;;GO TO EXIT
  1787  011634  010502                         MOV     R5,R2           ;;GET THE LSD
  1788  011636  000764                         BR      6$              ;;GO CHANGE TO ASCII
  1789  011640  105726                  8$:    TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
  1790  011642  100003                         BPL     9$              ;;BR IF NO
  1791  011644  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
  1792  011652  105013                  9$:    CLRB    (R3)            ;;SET THE TERMINATOR
```

```
1793  011654  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
1794  011656  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
1795  011660  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
1796  011662  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
1797  011664  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
1798  011666  104401  011714                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
1799  011672  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
1800  011700  012616                          MOV     (SP)+,(SP)
1801  011702  000002                          RTI                     ;;RETURN TO USER
1802  011704  023420                  $DTBL:  10000.
1803  011706  001750                          1000.
1804  011710  000144                          100.
1805  011712  000012                          10.
1806  011714  000004                  $DBLK:  .BLKW   4
1807
1808                                  ;;**********************************************************************
1809                                  ;*TEST 53        MISC. EXTERNAL LOGIC TEST
1810                                  ;;**********************************************************************
1811  011724  000004                  TST53:  SCOPE
1812  011726  004737  003402          EXTTST: JSR     PC,SETADD               ;SET UP ADDRESS AND VECTOR
1813  011732  012737  040000  012026  2$:     MOV     #BIT14,EXTCNT           ;LOAD COUNT
1814  011740                          1$:
1815  011740  005077  167364                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1816  011744  012777  125252  167356          MOV     #125252,@GRDIO          ;LOAD OUTPUT
1817  011752  017737  167350  012024          MOV     @GRDAI,EXTTMP           ;READ INPUT
1818  011760  013777  012024  167340          MOV     EXTTMP,@GRDAI           ;CLEAR INPUT
1819  011766  005077  167336                  CLR     @GRDIO                  ;CLEAR OUTPUT REGISTER
1820  011772  012777  052525  167330          MOV     #52525,@GRDIO           ;LOAD OUTPUT
1821  012000  017737  167322  012024          MOV     @GRDAI,EXTTMP           ;READ INPUT
1822  012006  013777  012024  167312          MOV     EXTTMP,@GRDAI           ;CLEAR INPUT
1823  012014  005337  012026                  DEC     EXTCNT                  ;FINISHED COUNT
1824  012020  001347                          BNE     1$
1825  012022  000743                          BR      2$                      ;LOOP
1826
1827  012024  000000                  EXTTMP: 0
1828  012026  000000                  EXTCNT: 0
```

```
1829                                    ;;*****************************************************************
1830                                    ;*TEST 54        COULTER INTERFACE TEST
1831                                    ;*****************************************************************
1832  012030  000004                    T$T54:  SCOPE
1833  012032  004737  003402            COULTR: JSR     PC,SETADD               ;SETUP BUS ADDRESS AND VECTOR
1834  012036  005077  167262                    CLR     @GRSTAT                 ;CLEAR INPUT STATUR REG
1835  012042  012777  012104  167270            MOV     #20$,@GRIVA             ;LOAD INPUT VECTOR
1836  012050  012777  000340  167264            MOV     #340,@GRIVSA
1837  012056  104401                    1$:     TYPE
1838  012060  012272                            RUNMSG                          ;TELL OPERATOR TO "RUN SAMPLE"
1839  012062  012701  015646                    MOV     #BUFFER,R1              ;LOAD POINTER FOR RESULTS
1840  012066  012777  000100  167230    4$:     MOV     #BIT6,@GRSTAT           ;ENABLE INTERRUPT
1841  012074  005037  177776                    CLR     PS                      ;LOWER PSW
1842  012100  000001                            WAIT                            ;WAIT FOR OPERATOR
1843  012102  000771                            BR      4$
1844
1845  012104  022626                    20$:    CMP     (SP)+,(SP)+             ;POP STACK
1846  012106  012737  000106  012270            MOV     #70.,100$               ;LOAD # OF 1 MSEC DELAYS
1847  012114  012700  000240            6$:     MOV     #240,R0                 ;LOAD 1 MSEC. DELAY WEIGHT
1848                                                                            ;240 FOR 11/05 TYPE
1849                                                                            ;620 FOR 11/40 TYPE
1850  012120  005300                    3$:     DEC     R0                      ;DELAY
1851  012122  001376                            BNE     3$
1852  012124  005337  012270                    DEC     100$                    ;FINISHED ALL MSEC. DELAY ?
1853  012130  001371                            BNE     6$                      ;BR IF NOT
1854  012132  017700  167170                    MOV     @GRDAI,R0               ;SAVE RESULTS
1855  012136  010021                            MOV     R0,(R1)+                ;SAVE IN BUFFER
1856  012140  012777  177777  167160            MOV     #-1,@GRDAI              ;CLEAR INPUT
1857  012146  042700  007777                    BIC     #7777,R0                ;MASK
1858  012152  001345                            BNE     4$                      ;BR IF NOT LAST SAMPLE
1859  012154  012741  123456                    MOV     #123456,-(R1)           ;LOAD TERM.
1860
1861  012160  012703  015646                    MOV     #BUFFER,R3              ;LOAD RESULT POINTER
1862  012164  012300                    5$:     MOV     (R3)+,R0                ;GET RESULT
1863  012166  022700  123456                    CMP     #123456,R0              ;TEST FOR TERM.
1864  012172  001731                            BEQ     1$                      ;BR IF DONE
1865  012174  004737  012202                    JSR     PC,40$                  ;CONVERT AND TYPE
1866  012200  000771                            BR      5$                      ;CONT. TYPING UNTIL DONE
1867
1868
1869                                    ;SUBROUTINE FOR THE COULTER TEST
1870  012202  012737  000055  012314    40$:    MOV     #55,MSGRUS+2            ;FIX ASCII MESSAGE
1871  012210  012702  012321                    MOV     #MSPNT1,R2              ;LOAD DEST. POINTER
1872  012214  012737  000004  012270            MOV     #4,100$                 ;LOAD COUNT
1873  012222  000404                            BR      12$
1874
1875  012224  006000                    10$:    ROR     R0
1876  012226  006000                            ROR     R0
1877  012230  006000                            ROR     R0
1878  012232  006000                            ROR     R0
1879  012234  010001                    12$:    MOV     R0,R1                   ;LOAD R1
1880  012236  042701  177760                    BIC     #177760,R1              ;MASK
1881  012242  052701  000060                    BIS     #60,R1                  ;MAKE DIGIT
1882  012246  110142                            MOVB    R1,-(R2)                ;SAVE DIGIT
1883  012250  005337  012270                    DEC     100$                    ;FINISHED ?
1884  012254  001363                            BNE     10$                     ;BR IF NOT
```

```
1885  012256  000337  012314                  SWAB    MSGRUS+2                ;ADJUST MESSAGE
1886  012262  104401                          TYPE
1887  012264  012312                          MSGRUS
1888  012266  000207                          RTS     PC                      ;EXIT
1889
1890  012270  000000                  100$:   0
1891  012272  005015  051012  047125  RUNMSG: .ASCIZ  <15><12><12>/RUN SAMPLE/<15><12>
1892  012300  051440  046501  046120
1893  012306  006505  000012
1894  012312  005015  047055  047116  MSGRUS: .ASCII  <15><12>/-NNNN/
1895  012320     116
1896  012321     000                  MSPNT1: .BYTE   0
1897                                          .EVEN
1898
1899  012322  052123  052101  051525  EM1:    .ASCIZ  /STATUS REGISTER IN ERROR/
1900  012330  051040  043505  051511
1901  012336  042524  020122  047111
1902  012344  042440  051122  051117
1903  012352     000
1904  012353     111  050116  052125  EM2:    .ASCIZ  /INPUT REGISTER IN ERROR/
1905  012360  051040  043505  051511
1906  012366  042524  020122  047111
1907  012374  042440  051122  051117
1908  012402     000
1909  012403     117  052125  052520  EM3:    .ASCIZ  /OUTPUT REGISTER IN ERROR/
1910  012410  020124  042522  044507
1911  012416  052123  051105  044440
1912  012424  020116  051105  047522
1913  012432  000122
1914  012434  047111  052520  020124  EM4:    .ASCIZ  /INPUT FAILED TO INTERRUPT/
1915  012442  040506  046111  042105
1916  012450  052040  020117  047111
1917  012456  042524  051122  050125
1918  012464  000124
1919  012466  052517  050124  052125  EM5:    .ASCIZ  /OUTPUT FAILED TO INTERRUPT/
1920  012474  043040  044501  042514
1921  012502  020104  047524  044440
1922  012510  052116  051105  052522
1923  012516  052120     000
1924  012521     125  042516  050130  EM6:    .ASCIZ  /UNEXPECTED INTERRUPT/
1925  012526  041505  042524  020104
1926  012534  047111  042524  051122
1927  012542  050125  000124
1928  012546  050117  051105  052101  EM7:    .ASCIZ  /OPERATOR INTERVENTION ERROR/
1929  012554  051117  044440  052116
1930  012562  051105  042526  052116
1931  012570  047511  020116  051105
1932  012576  047522  000122
1933  012602  047111  042524  051122  EM10:   .ASCIZ  /INTERRUPT INPUT BIT FAILED TO SET INPUT READY FLAG/
1934  012610  050125  020124  047111
1935  012616  052520  020124  044502
1936  012624  020124  040506  046111
1937  012632  042105  052040  020117
1938  012640  042523  020124  047111
1939  012646  052520  020124  042522
1940  012654  042101  020131  046106
```

```
1941  012662  043501     000
1942  012665     116  047117  044455  EM11:   .ASCIZ  /NON-INTERRUPT INPUT BIT SET INPUT READY FLAG/
1943  012672  052116  051105  052522
1944  012700  052120  044440  050116
1945  012706  052125  041040  052111
1946  012714  051440  052105  044440
1947  012722  050116  052125  051040
1948  012730  040505  054504  043040
1949  012736  040514  000107
1950
1951  012742  051105  050122  020103  DH1:    .ASCIZ  /ERRPC   DRADD   STATUS  EXPECTED/
1952  012750  020040  051104  042101
1953  012756  020104  020040  052123
1954  012764  052101  051525  020040
1955  012772  054105  042520  052103
1956  013000  042105     000
1957  013003     105  051122  041520  DH2:    .ASCIZ  /ERRPC   DRADD   INPUT   EXPECTED/
1958  013010  020040  042040  040522
1959  013016  042104  020040  044440
1960  013024  050116  052125  020040
1961  013032  042440  050130  041505
1962  013040  042524  000104
1963  013044  051105  050122  020103  DH3:    .ASCIZ  /ERRPC   DRADD   OUTPUT  EXPECTED/
1964  013052  020040  051104  042101
1965  013060  020104  020040  052517
1966  013066  050124  052125  020040
1967  013074  054105  042520  052103
1968  013102  042105     000
1969  013105     105  051122  041520  DH4:    .ASCIZ  /ERRPC   DRADD/
1970  013112  020040  042040  040522
1971  013120  042104     000
1972  013123     105  051122  041520  DH10:   .ASCIZ  /ERRPC   DRADD   STATUS  EXPECT  INPUT BIT/
1973  013130  020040  042040  040522
1974  013136  042104  020040  051440
1975  013144  040524  052524  020123
1976  013152  042440  050130  041505
1977  013160  020124  044440  050116
1978  013166  052125  041040  052111
1979  013174     000
1980          013176                  .EVEN
1981  013176  001116  001316  001126  DT1:    $ERRPC,DRADD,$BDDAT,$GDDAT,0
1982  013204  001124  000000
1983  013210  001116  001316  000000  DT4:    $ERRPC,DRADD,0
1984  013216  001116  001316  001126  DT10:   $ERRPC,DRADD,$BDDAT,$GDDAT,BRLEV3,0
1985  013224  001124  001356  000000
```

```
                                        .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
013232                                  $SCOPE:
013232  104407                                 CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
013234  032777  040000  165676  1$:      BIT     #BIT14,@SWR             ;;LOOP ON PRESENT TEST?
013242  001111                         BNE     $OVER                   ;;YES IF SW14=1
                                        ;#####START OF CODE FOR THE XOR TESTER#####
013244  000416                         $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
013246  013746  000004                         MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
013252  012737  013272  000004                 MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
013260  005737  177060                         TST     @#177060        ;;TIME OUT ON XOR?
013264  012637  000004                         MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
013270  000463                               BR      $SVLAD          ;;GO TO THE NEXT TEST
013272  022626                         5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
013274  012637  000004                         MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
013300  000423                               BR      7$              ;;LOOP ON THE PRESENT TEST
013302                                  6$:;#####END OF CODE FOR THE XOR TESTER#####
013302  032777  000400  165630               BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
013310  001404                               BEQ     2$              ;;BR IF NO
013312  127737  165622  001102               CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
013320  001462                               BEQ     $OVER           ;;BR IF YES
013322  105737  001103           2$:         TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
013326  001421                               BEQ     3$              ;;BR IF NO
013330  123737  001115  001103               CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
013336  101015                               BHI     3$              ;;BR IF NO
013340  032777  001000  165572               BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
013346  001404                               BEQ     4$              ;;BR IF NO
013350  013737  001110  001106  7$:          MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
013356  000443                               BR      $OVER
013360  105037  001103           4$:         CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
013364  005037  001166                       CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
013370  000415                               BR      1$              ;;ESCAPE TO THE NEXT TEST
013372  032777  004000  165540  3$:          BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
013400  001011                               BNE     1$              ;;BR IF YES
013402  005737  001100                       TST     $PASS           ;;IF FIRST PASS OF PROGRAM
013406  001406                               BEQ     1$              ;;      INHIBIT ITERATIONS
013410  005237  001104                       INC     $ICNT           ;;INCREMENT ITERATION COUNT
013414  023737  001166  001104               CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
013422  002021                               BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
013424  012737  000001  001104  1$:          MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
013432  013737  013502  001166               MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
013440  105237  001102           $SVLAD:     INCB    $TSTNM          ;;COUNT TEST NUMBERS
```

```
2042  013444  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
2043  013450  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
2044  013454  005037  001170                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
2045  013460  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
2046  013466  013777  001102  165446  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
2047  013474  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
2048  013500  000002                          RTI                     ;;FIXES PS
2049  013502  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
2050                                   .SBTTL  ERROR HANDLER ROUTINE
2051
2052                                   ;;**************************************************************
2053                                   ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2054                                   ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
2055                                   ;*AND GO TO $ERRTYP ON ERROR
2056                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
2057                                   ;*SW15=1        HALT ON ERROR
2058                                   ;*SW13=1        INHIBIT ERROR TYPEOUTS
2059                                   ;*SW09=1        LOOP ON ERROR
2060                                   ;*CALL
2061                                   ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
2062
2063  013504                           $ERROR:
2064  013504  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2065  013506  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
2066  013512  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
2067  013514  013777  001102  165420          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
2068  013522  005237  001112                  INC     $ERTTL          ;;INC THE ERROR COUNT
2069  013526  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
2070  013532  162737  000002  001116          SUB     #2,$ERRPC
2071  013540  117737  165352  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
2072  013546  032777  020000  165364          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
2073  013554  001004                          BNE     20$             ;;SKIP TYPEOUTS
2074  013556  004737  013642                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
2075  013562  104401  001173                  TYPE    ,$CRLF
2076  013566                           20$:
2077  013566  005777  165346          2$:     TST     @SWR            ;;HALT ON ERROR
2078  013572  100002                          BPL     3$              ;;SKIP IF CONTINUE
2079  013574  000000                          HALT                    ;;HALT ON ERROR!
2080  013576  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
2081  013600  032777  001000  165332  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
2082  013606  001402                          BEQ     4$              ;;BR IF NO
2083  013610  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
2084  013614  005737  001170          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
2085  013620  001402                          BEQ     5$              ;;BR IF NONE
2086  013622  013716  001170                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
2087  013626                           5$:
2088  013626  022737  011444  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
2089  013634  001001                          BNE     6$              ;;BRANCH IF NO
2090  013636  000000                          HALT                    ;;YES
2091  013640                           6$:
2092  013640  000002                          RTI                     ;;RETURN
```

```
                                        .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

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

013642                                  $ERRTYP:
013642  104401  001173                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
013646  010046                                  MOV     R0,-(SP)        ;;SAVE R0
013650  005000                                  CLR     R0              ;;PICKUP THE ITEM INDEX
013652  153700  001114                          BISB    @#$ITEMB,R0
013656  001004                                  BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                        ;;TYPE THE PC OF THE ERROR
013660  013746  001116                          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;ERROR ADDRESS
013664  104402                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
013666  000426                                  BR      6$              ;;GET OUT
013670  005300                          1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
013672  006300                                  ASL     R0              ;;      WORK FOR THE ERROR TABLE
013674  006300                                  ASL     R0
013676  006300                                  ASL     R0
013700  062700  001176                          ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
013704  012037  013714                          MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
013710  001404                                  BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
013712  104401                                  TYPE                    ;;TYPE THE "ERROR MESSAGE"
013714  000000                          2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
013716  104401  001173                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
013722  012037  013732                  3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
013726  001404                                  BEQ     5$              ;;SKIP TYPEOUT IF 0
013730  104401                                  TYPE                    ;;TYPE THE "DATA HEADER"
013732  000000                          4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
013734  104401  001173                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
013740  011000                          5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
013742  001004                                  BNE     7$              ;;GO TYPE THE DATA
013744  012600                          6$:     MOV     (SP)+,R0        ;;RESTORE R0
013746  104401  001173                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
013752  000207                                  RTS     PC              ;;RETURN
013754                                  7$:
013754  013046                                  MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
013756  104402                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
013760  005710                                  TST     (R0)            ;;IS THERE ANOTHER NUMBER?
013762  001770                                  BEQ     6$              ;;BR IF NO
013764  104401  013772                          TYPE    ,8$             ;;TYPE TWO(2) SPACES
013770  000771                                  BR      7$              ;;LOOP
013772  020040    000           8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
013776                                  .EVEN
```

```
2141
2142                                    .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
2143
2144                                    ;;*****************************************************************
2145                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2146                                    ;*OCTAL (ASCII) NUMBER AND TYPE IT.
2147                                    ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2148                                    ;*CALL:
2149                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2150                                    ;*      TYPOS                   ;;CALL FOR TYPEOUT
2151                                    ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2152                                    ;*      .BYTE   M               ;;M=1 OR 0
2153                                    ;*                                      ;;1=TYPE LEADING ZEROS
2154                                    ;*                                      ;;0=SUPPRESS LEADING ZEROS
2155                                    ;*
2156                                    ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2157                                    ;*$TYPOS OR $TYPOC
2158                                    ;*CALL:
2159                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2160                                    ;*      TYPON                   ;;CALL FOR TYPEOUT
2161                                    ;*
2162                                    ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
2163                                    ;*CALL:
2164                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2165                                    ;*      TYPOC                   ;;CALL FOR TYPEOUT
2166
2167  013776  017646  000000           $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
2168  014002  116637  000001  014221           MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
2169  014010  112637  014223                   MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
2170  014014  062716  000002                   ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
2171  014020  000406                           BR      $TYPON
2172  014022  112737  000001  014221  $TYPOC:  MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
2173  014030  112737  000006  014223           MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
2174  014036  112737  000005  014220  $TYPON:  MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
2175  014044  010346                           MOV     R3,-(SP)        ;;SAVE R3
2176  014046  010446                           MOV     R4,-(SP)        ;;SAVE R4
2177  014050  010546                           MOV     R5,-(SP)        ;;SAVE R5
2178  014052  113704  014223                   MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
2179  014056  005404                           NEG     R4
2180  014060  062704  000006                   ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
2181  014064  110437  014222                   MOVB    R4,$OMODE       ;;SAVE IT FOR USE
2182  014070  113704  014221                   MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2183  014074  016605  000012                   MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
2184  014100  005003                           CLR     R3              ;;CLEAR THE OUTPUT WORD
2185  014102  006105                   1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2186  014104  000404                           BR      3$              ;;GO DO MSB
2187  014106  006105                   2$:     ROL     R5              ;;FORM THIS DIGIT
2188  014110  006105                           ROL     R5
2189  014112  006105                           ROL     R5
2190  014114  010503                           MOV     R5,R3
2191  014116  006103                   3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2192  014120  105337  014222                   DECB    $OMODE          ;;TYPE THIS DIGIT?
2193  014124  100016                           BPL     7$              ;;BR IF NO
2194  014126  042703  177770                   BIC     #177770,R3      ;;GET RID OF JUNK
2195  014132  001002                           BNE     4$              ;;TEST FOR 0
2196  014134  005704                           TST     R4              ;;SUPPRESS THIS 0?
```

```
  2197   014136   001403                          BEQ     5$              ;;BR IF YES
  2198   014140   005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
  2199   014142   052703   000060                 BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
  2200   014146   052703   000040         5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
  2201   014152   110337   014216                 MOVB    R3,8$           ;;SAVE FOR TYPING
  2202   014156   104401   014216                 TYPE    ,8$             ;;GO TYPE THIS DIGIT
  2203   014162   105337   014220         7$:     DECB    $OCNT           ;;COUNT BY 1
  2204   014166   003347                         BGT     2$              ;;BR IF MORE TO DO
  2205   014170   002402                         BLT     6$              ;;BR IF DONE
  2206   014172   005204                         INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
  2207   014174   000744                         BR      2$              ;;GO DO THE LAST DIGIT
  2208   014176   012605                 6$:     MOV     (SP)+,R5        ;;RESTORE R5
  2209   014200   012604                         MOV     (SP)+,R4        ;;RESTORE R4
  2210   014202   012603                         MOV     (SP)+,R3        ;;RESTORE R3
  2211   014204   016666   000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
  2212   014212   012616                         MOV     (SP)+,(SP)
  2213   014214   000002                         RTI                     ;;RETURN
  2214   014216      000                 8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
  2215   014217      000                         .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
  2216   014220      000                 $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
  2217   014221      000                 $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
  2218   014222   000000                 $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;**********************************************************************
                                        ;POWER DOWN ROUTINE
014224  012737  014370  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
014232  012737  000340  000026         MOV     #340,@#PWRVEC+2  ;;PRIO:7
014240  010046                         MOV     R0,-(SP)         ;;PUSH R0 ON STACK
014242  010146                         MOV     R1,-(SP)         ;;PUSH R1 ON STACK
014244  010246                         MOV     R2,-(SP)         ;;PUSH R2 ON STACK
014246  010346                         MOV     R3,-(SP)         ;;PUSH R3 ON STACK
014250  010446                         MOV     R4,-(SP)         ;;PUSH R4 ON STACK
014252  010546                         MOV     R5,-(SP)         ;;PUSH R5 ON STACK
014254  017746  164660                 MOV     @SWR,-(SP)       ;;PUSH @SWR ON STACK
014260  010637  014374                 MOV     SP,$SAVR6        ;;SAVE SP
014264  012737  014276  000024         MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
014272  000000                         HALT
014274  000776                         BR      .-2              ;;HANG UP

                                        ;;**********************************************************************
                                        ;POWER UP ROUTINE
014276  012737  014370  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
014304  013706  014374                 MOV     $SAVR6,SP        ;;GET SP
014310  005037  014374                 CLR     $SAVR6           ;;WAIT LOOP FOR THE TTY
014314  005237  014374         1$:     INC     $SAVR6           ;;WAIT FOR THE INC
014320  001375                         BNE     1$               ;;OF  WORD
014322  012677  164612                 MOV     (SP)+,@SWR       ;;POP STACK INTO @SWR
014326  012605                         MOV     (SP)+,R5         ;;POP STACK INTO R5
014330  012604                         MOV     (SP)+,R4         ;;POP STACK INTO R4
014332  012603                         MOV     (SP)+,R3         ;;POP STACK INTO R3
014334  012602                         MOV     (SP)+,R2         ;;POP STACK INTO R2
014336  012601                         MOV     (SP)+,R1         ;;POP STACK INTO R1
014340  012600                         MOV     (SP)+,R0         ;;POP STACK INTO R0
014342  012737  014224  000024         MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
014350  012737  000340  000026         MOV     #340,@#PWRVEC+2  ;;PRIO:7
014356  104401                         TYPE                     ;;REPORT THE POWER FAILURE
014360  014376                 $PWRMG: .WORD   PWRMSG           ;;POWER FAIL MESSAGE POINTER
014362  012716                         MOV     (PC)+,(SP)       ;;RESTART AT IOTEST
014364  003316                 $PWRAD: .WORD   IOTEST           ;;RESTART ADDRESS
014366  000002                         RTI
014370  000000                 $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
014372  000776                         BR      .-2              ;; BEFORE THE POWER DOWN WAS COMPLETE
014374  000000                 $SAVR6: 0                        ;;PUT THE SP HERE
014376  005015  042522  052123 PWRMSG: .ASCIZ  <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12>
014404  051101  044524  043516
014412  040440  052106  051105
014420  040440  050040  053517
014426  051105  043040  044501
014434  052514  042522  005015
014442      000
                014444                   .EVEN
```

```
2270
2271                                    .SBTTL  TYPE ROUTINE
2272
2273                                    ;;*********************************************************************
2274                                    ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2275                                    ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2276                                    ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2277                                    ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2278                                    ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2279                                    ;*
2280                                    ;*CALL:
2281                                    ;*1) USING A TRAP INSTRUCTION
2282                                    ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2283                                    ;*OR
2284                                    ;*      TYPE
2285                                    ;*      MESADR
2286                                    ;*
2287
2288  014444  105737  001157   $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
2289  014450  100002                  BPL     1$              ;;BR IF YES
2290  014452  000000                  HALT                    ;;HALT HERE IF NO TERMINAL
2291  014454  000407                  BR      3$              ;;LEAVE
2292  014456  010046           1$:    MOV     R0,-(SP)        ;;SAVE R0
2293  014460  017600  000002         MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
2294  014464  112046           2$:    MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
2295  014466  001005                  BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
2296  014470  005726                  TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
2297  014472  012600           60$:   MOV     (SP)+,R0        ;;RESTORE R0
2298  014474  062716  000002   3$:    ADD     #2,(SP)         ;;ADJUST RETURN PC
2299  014500  000002                  RTI                     ;;RETURN
2300  014502  122716  000011   4$:    CMPB    #HT,(SP)        ;;BRANCH IF <HT>
2301  014506  001430                  BEQ     8$
2302  014510  122716  000200         CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
2303  014514  001006                  BNE     5$
2304  014516  005726                  TST     (SP)+           ;;POP  <CR><LF> EQUIV
2305  014520  104401                  TYPE                    ;;TYPE A CR AND LF
2306  014522  001173                  $CRLF
2307  014524  105037  014660         CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
2308  014530  000755                  BR      2$              ;;GET NEXT CHARACTER
2309  014532  004737  014614   5$:    JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
2310  014536  123726  001156   6$:    CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
2311  014542  001350                  BNE     2$              ;;IF NO GO GET NEXT CHAR.
2312  014544  013746  001154         MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
2313                                                          ;;AND THE NULL CHAR.
2314  014550  105366  000001   7$:    DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2315  014554  002770                  BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2316  014556  004737  014614         JSR     PC,$TYPEC       ;;GO TYPE A NULL
2317  014562  105337  014660         DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2318  014566  000770                  BR      7$              ;;LOOP
2319
2320                                    ;HORIZONTAL TAB PROCESSOR
2321
2322  014570  112716  000040   8$:    MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2323  014574  004737  014614   9$:    JSR     PC,$TYPEC       ;;TYPE A SPACE
2324  014600  132737  000007  014660  BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2325  014606  001372                  BNE     9$              ;;TAB STOP
```

```
2326  014610  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
2327  014612  000724                        BR      2$              ;;GET NEXT CHARACTER
2328  014614  105777  164330        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2329  014620  100375                        BPL     $TYPEC
2330  014622  116677  000002  164322        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2331  014630  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2332  014636  001003                        BNE     1$              ;;BRANCH IF NO
2333  014640  105037  014660                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2334  014644  000406                        BR      $TYPEX          ;;EXIT
2335  014646  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2336  014654  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
2337  014656  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
2338  014660  000000        $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2339  014662  000207        $TYPEX: RTS     PC
2340
```

```
2341                                    .SBTTL  TTY INPUT ROUTINE
2342
2343                                    ;;**********************************************************************
2344                                    .ENABL  LSB
2345
2346                                    ;;**********************************************************************
2347                                    ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
2348                                    ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
2349                                    ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
2350                                    ;*WHEN OPERATING IN TTY FLAG MODE.
2351  014664  022737  000176  001140  $CKSWR: CMP     #$SWREG,SWR     ;;IS THE SOFT-SWR SELECTED?
2352  014672  001074                         BNE     15$             ;;BRANCH IF NO
2353  014674  105777  164244                 TSTB    @$TKS           ;;CHAR THERE?
2354  014700  100071                         BPL     15$             ;;IF NO, DON'T WAIT AROUND
2355  014702  117746  164240                 MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
2356  014706  042716  177600                 BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
2357  014712  022726  000007                 CMP     #7,(SP)+        ;;IS IT A CONTROL G?
2358  014716  001062                         BNE     15$             ;;NO, RETURN TO USER
2359  014720  123727  001134  000001         CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
2360  014726  001456                         BEQ     15$             ;;BRANCH IF YES
2361
2362  014730  104401  015536                 TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
2363  014734  104401  015543         $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
2364  014740  013746  000176                 MOV     $SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
2365  014744  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2366  014746  104401  015554                 TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
2367  014752  005046                 19$:    CLR     -(SP)           ;;CLEAR COUNTER
2368  014754  005046                         CLR     -(SP)           ;;THE NEW SWR
2369  014756  105777  164162         7$:     TSTB    @$TKS           ;;CHAR THERE?
2370  014762  100375                         BPL     7$              ;;IF NOT TRY AGAIN
2371
2372  014764  117746  164156                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
2373  014770  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
2374
2375
2376
2377  014774  021627  000025         9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
2378  015000  001005                         BNE     10$             ;;BRANCH IF NOT
2379  015002  104401  015531                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
2380  015006  062706  000006         20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
2381  015012  000757                         BR      19$             ;;LET'S TRY IT AGAIN
2382
2383
2384  015014  021627  000015         10$:    CMP     (SP),#15        ;;IS IT A <CR>?
2385  015020  001022                         BNE     16$             ;;BRANCH IF NO
2386  015022  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
2387  015026  001403                         BEQ     11$             ;;BRANCH IF YES
2388  015030  016677  000002  164102         MOV     2(SP),@SWR      ;;SAVE NEW SWR
2389  015036  062706  000006         11$:    ADD     #6,SP           ;;CLEAR UP STACK
2390  015042  104401  001173         14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
2391  015046  123727  001135  000001         CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
2392  015054  001003                         BNE     15$             ;;BRANCH IF NOT
2393  015056  012777  000100  164060         MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
2394  015064  000002                 15$:    RTI                     ;;RETURN
2395  015066  004737  014614         16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
2396  015072  021627  000060                 CMP     (SP),#60        ;;CHAR < 0?
```

```
2397  015076  002420                        BLT     18$             ;;BRANCH IF YES
2398  015100  021627  000067                CMP     (SP),#67        ;;CHAR > 7?
2399  015104  003015                        BGT     18$             ;;BRANCH IF YES
2400  015106  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
2401  015112  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
2402  015116  001403                        BEQ     17$             ;;BRANCH IF YES
2403  015120  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
2404  015122  006316                        ASL     (SP)            ;;   CHAR OVER TO MAKE
2405  015124  006316                        ASL     (SP)            ;;   ROOM FOR NEW ONE.
2406  015126  005266  000002        17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
2407  015132  056616  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
2408  015136  000707                        BR      7$              ;;GET THE NEXT ONE
2409  015140  104401  001172        18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
2410  015144  000720                        BR      20$             ;;SIMULATE CONTROL-U
2411                                .DSABL  LSB
2412
2413
2414                                ;;*****************************************************************
2415                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2416                                ;*CALL:
2417                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
2418                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
2419                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
2420                                ;
2421
2422  015146  011646                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
2423  015150  016666  000004  000002        MOV     4(SP),2(SP)     ;;SAVE THE PS
2424  015156  105777  163762        1$:     TSTB    @$TKS           ;;WAIT FOR
2425  015162  100375                        BPL     1$              ;;A CHARACTER
2426  015164  117766  163756  000004        MOVB    @$TKB,4(SP)     ;;READ THE TTY
2427  015172  042766  177600  000004        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
2428  015200  026627  000004  000023        CMP     4(SP),#23       ;;IS IT A CONTROL-S?
2429  015206  001013                        BNE     3$              ;;BRANCH IF NO
2430  015210  105777  163730        2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
2431  015214  100375                        BPL     2$              ;;LOOP UNTIL ITS THERE
2432  015216  117746  163724                MOVB    @$TKB,-(SP)     ;;GET CHARACTER
2433  015222  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
2434  015226  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
2435  015232  001366                        BNE     2$              ;;IF NOT DISCARD IT
2436  015234  000750                        BR      1$              ;;YES, RESUME
2437  015236  026627  000004  000140  3$:   CMP     4(SP),#140      ;;IS IT UPPER CASE?
2438  015244  002407                        BLT     4$              ;;BRANCH IF YES
2439  015246  026627  000004  000175        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
2440  015254  003003                        BGT     4$              ;;BRANCH IF YES
2441  015256  042766  000040  000004        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
2442  015264  000002                4$:     RTI                     ;;GO BACK TO USER
2443                                ;;*****************************************************************
2444                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2445                                ;*CALL:
2446                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
2447                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2448                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2449
2450  015266  010346                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
2451  015270  005046                        CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
2452  015272  012703  015522        1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
```

```
2453  015276  022703  015531          2$:     CMP     #$TTYIN+7,R3    ;;BUFFER FULL?
2454  015302  101456                          BLOS    4$              ;;BR IF YES
2455  015304  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2456  015306  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
2457  015310  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
2458  015314  001022                          BNE     5$              ;;BR IF NO
2459  015316  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
2460  015320  001007                          BNE     6$              ;;BR IF NO
2461  015322  112737  000134  015520        MOVB    #'\,9$          ;;TYPE A BACK SLASH
2462  015330  104401  015520                  TYPE    ,9$
2463  015334  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
2464  015340  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
2465  015342  020327  015522                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
2466  015346  103434                          BLO     4$              ;;BR IF YES
2467  015350  111337  015520                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
2468  015354  104401  015520                  TYPE    ,9$             ;;GO TYPE
2469  015360  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
2470  015362  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
2471  015364  001406                          BEQ     7$              ;;BR IF NO
2472  015366  112737  000134  015520        MOVB    #'\,9$          ;;TYPE A BACK SLASH
2473  015374  104401  015520                  TYPE    ,9$
2474  015400  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
2475  015402  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
2476  015406  001003                          BNE     8$              ;;BR IF NO
2477  015410  104401  015531                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
2478  015414  000726                          BR      1$              ;;GO START OVER
2479  015416  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
2480  015422  001011                          BNE     3$              ;;BRANCH IF NO
2481  015424  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
2482  015426  104401  001173                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
2483  015432  104401  015522                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
2484  015436  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
2485  015440  104401  001172          4$:     TYPE    ,$QUES          ;;TYPE A '?'
2486  015444  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
2487  015446  111337  015520          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2488  015452  104401  015520                  TYPE    ,9$
2489  015456  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2490  015462  001305                          BNE     2$              ;;LOOP IF NOT RETURN
2491  015464  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2492  015470  104401  001174                  TYPE    ,$LF            ;;TYPE A LINE FEED
2493  015474  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
2494  015476  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2495  015500  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2496  015502  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2497  015510  012766  015522  000004          MOV     #$TTYIN,4(SP)
2498  015516  000002                          RTI                     ;;RETURN
2499  015520     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2500  015521     000                          .BYTE   0               ;;TERMINATOR
2501  015522  000007          $TTYIN: .BLKB   7                       ;;RESERVE 7 BYTES FOR TTY INPUT
2502  015531     136  006525  000012 $CNTLU: .ASCIZ  /↑U/<15><12>     ;;CONTROL "U"
2503  015536  043536  005015     000 $CNTLG: .ASCIZ  /↑G/<15><12>     ;;CONTROL "G"
2504  015543     015  051412  051127 $MSWR:  .ASCIZ  <15><12>/SWR = /
2505  015550  036440  000040
2506  015554  020040  042516  020127 $MNEW:  .ASCIZ  /  NEW = /
2507  015562  020075     000
2508          015566                  .EVEN
```

```
                                        .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.

015566  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
015570  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
015574  005740                          TST     -(R0)           ;;BACKUP BY 2
015576  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
015600  006300                          ASL     R0              ;;POSITION FOR INDEXING
015602  016000  015622                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
015606  000200                          RTS     R0              ;;GO TO ROUTINE

                                        ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

015610  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
015612  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
015620  000002                          RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                        ;       -------
015622  015610                  $TRPAD: .WORD   $TRAP2
015624  014444                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
015626  014022                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
015630  013776                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
015632  014036                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
015634  011500                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

015636  014734                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

015640  014664                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
015642  015146                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
015644  015266                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE

015646  000000                  BUFFER: 0                       ;10 WORD BUFFER FOR THE COULTER INTERFACE TEST
        000001                  .END
```

```
BASEBA  001306            499#    536     720    1694
BASEIV  001310            500#    721    1695
BEGIN   001366            378     530#    696
BEGIN1  001372            379     532#
BIT0  = 000001            340#    830     846    1091    1127    1152    1523    1562
BIT00 = 000001            330#    340
BIT01 = 000002            329#    339
BIT02 = 000004            328#    338
BIT03 = 000010            327#    337
BIT04 = 000020            326#    336
BIT05 = 000040            325#    335
BIT06 = 000100            324#    334
BIT07 = 000200            323#    333
BIT08 = 000400            322#    332    2017
BIT09 = 001000            321#    331    2025    2081
BIT1  = 000002            339#
BIT10 = 002000            320#    994    1489    1519    1558    1591
BIT11 = 004000            319#   2032
BIT12 = 010000            318#    689    1059
BIT13 = 020000            317#   1466    2072
BIT14 = 040000            316#    945     946     982    1599    1605    1813    2003
BIT15 = 100000            315#    933     934     982    1360
BIT2  = 000004            338#
BIT3  = 000010            337#
BIT4  = 000020            336#
BIT5  = 000040            335#   1410    1439    1641    1660
BIT6  = 000100            334#    968     969     982    1438    1448    1497    1503    1840
BIT7  = 000200            333#    956     957     982    1345    1525
BIT8  = 000400            332#
BIT9  = 001000            331#
BPTVEC= 000014            347#
BRLEV1  001352            520#   1629*   1630*   1640
BRLEV2  001354            521#   1091*   1094    1114    1119*   1631*   1659
BRLEV3  001356            522#   1092*   1093*   1102    1112    1152*   1156    1163    1172    1173*   1523*   1527    1532
                         1533    1534    1544    1545   1552*   1562*   1566    1571    1572    1573    1583*   1632*   1633*
                         1984
BUFFER  015646           1839    1861    2553#
BYPASS  011260           1686#
BYPAS1  011346           1689    1693    1698#   1735
CKSWR = 104407           2002    2064    2080    2549#
COULTR  012032            381    1833#
CR    = 000015            255#   2331    2341
CRLF  = 000200            256#   2302    2341
DDISP = 177570            262#    410     579
DH1     012742            447    1951#
DH10    013123            489     495    1972#
DH2     013003            453    1957#
DH3     013044            459    1963#
DH4     013105            465     471     477     483    1969#
DIOBRL  001350            519#    750*   1623*   1626    1629    1631    1632
DISPLA  001142            410#    579*    587*   2046*   2067*
DISPRE  000174            375#    587
DRADD   001316            504#    720*    735     736     738     740    1690*   1694*   1981    1983    1984
DRBRL   001322            506#    750
DRIV    001320            505#    721*    742     743     745     747    1691*   1695*
DRT21   007406            996    1372#
```

```
DSWR   = 177570          261#    409     578
DT1      013176          448     454     460    1981#
DT10     013216          490     496    1984#
DT4      013210          466     472     478     484    1983#
EMTVEC=  000030          350#    562*    563*
EM1      012322          446    1899#
EM10     012602          488    1933#
EM11     012665          494    1942#
EM2      012353          452    1904#
EM3      012403          458    1909#
EM4      012434          464    1914#
EM5      012466          470    1919#
EM6      012521          476    1924#
EM7      012546          482    1928#
ERRVEC=  000004          343#    576     577*    588*    2008    2009*   2011*   2014*
EXTCNT   012026         1813*   1823*   1828#
EXTTMP   012024         1817*   1818    1821*   1822    1827#
EXTTST   011726          380    1812#
GNS    = ****** U        374     598     602     609     613     617     632     636     651     655     670     674     703
                         707    2541    2542    2543    2544    2545    2547    2549    2550    2551
GRBHIO   001332          512#    740*    741*    920*
GRDAI    001326          510#    736*    737*    764*    932*    999*   1002    1013*   1017    1028*   1032    1043*   1047
                        1065*   1067    1075*   1077    1101    1111    1129*   1138    1143*   1154*   1167    1172*   1185*
                        1189    1190*   1194    1195*   1199    1200*   1204    1205*   1209    1210*   1214    1215*   1219
                        1220*   1224    1225*   1229    1230*   1234    1235*   1239    1240*   1261*   1265    1266*   1270
                        1271*   1275    1276*   1280    1281*   1285    1286*   1290    1291*   1295    1296*   1300    1301*
                        1305    1306*   1310    1311*   1315    1316*   1332*   1336    1364    1386    1499    1530*   1545*
                        1569*   1601    1817    1818*   1821    1822*   1854    1856*
GRDIO    001330          511#    738*    739*    765*    772*    773     783*    785     794*    795     805*    806     818*
                         819     832*    833*    834     848*    849*    850     864*    865*    866*    867*    868*    869*
                         870*    871*    872*    873     882*    883*    884*    885*    886*    887*    888*    889*    890*
                         891     903*    904*    905     919*    921     931*    998*   1001*   1012*   1016*   1027*   1031*
                        1042*   1046*   1064*   1066*   1076*   1100*   1110*   1128*   1136*   1137*   1153*   1164*   1165*
                        1184*   1187*   1188*   1192*   1193*   1197*   1198*   1202*   1203*   1207*   1208*   1212*   1213*
                        1217*   1218*   1222*   1223*   1227*   1228*   1232*   1233*   1237*   1238*   1260*   1263*   1264*
                        1268*   1269*   1273*   1274*   1278*   1279*   1283*   1284*   1288*   1289*   1293*   1294*   1298*
                        1299*   1303*   1304*   1308*   1309*   1313*   1314*   1331*   1333*   1347*   1362*   1385*   1498*
                        1529*   1532*   1533*   1534*   1544*   1568*   1571*   1572*   1573*   1600*   1815*   1816*   1819*
                        1820*
GRIVA    001340          515#    742*    755*   1381*   1406*   1417*   1434*   1443*   1462*   1493*   1510*   1597*   1610*
                        1639*   1648*   1658*   1665*   1676*   1835*
GRIVB    001344          517#    745*    746*    757*   1383*   1408*   1436*   1464*   1473*   1495*   1512*   1595*   1612*
                        1650*   1678*
GRIVSA   001342          516#    743*    744*    755     756*   1382*   1407*   1435*   1462    1463*   1494*   1510    1511*
                        1597    1598*   1610    1611*   1648    1649*   1676    1677*   1836*
GRIVSB   001346          518#    747*    748*    757     758*   1384*   1408    1409*   1436    1437*   1465*   1495    1496*
                        1512    1513*   1596*   1612    1613*   1650    1651*   1678    1679*
GRSTAT   001324          509#    735*    763*    934*    935     944*    946*    947     957*    958     967*    969*    970
                         980*    982*    984    1346*   1349    1361*   1365    1379*   1391*   1405*   1410*   1433*   1438*
                        1439*   1447    1461*   1466*   1492*   1497*   1503*   1509*   1531*   1535*   1538    1547*   1548
                        1570*   1574*   1576    1594*   1599*   1605*   1615*   1638*   1641*   1657*   1660*   1834*   1840*
GTSWR  = 104406          625     644     663     682     715    2547#
HT     = 000011          253#   2300    2341
INTBIT   001336          514#    648*   1527    1566
IOTEST   003316          594     719#   2257
IOTST1   003372          732#
```

DZDRGC.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
IOTTS1  003530            734     750#
IOTVEC= 000020            348#    560*    561*
LF    = 000012            254#   2335    2341
MINSIN  001362            524#    667*    690     752    1073
MSGRUS  012212           1870*   1885*   1887    1894#
MSPNTI  012321           1871    1896#
NBEXT   001314            502#    722*   1686    1692*   1696*
NMBEXT  001312            501#    548*    604     722    1696
NOTLCH  001334            513#    629*   1062    1092    1097    1131    1156    1162    1181    1258
ODDJMP  001360            523#    751*    752*    753*   1003    1015    1018    1030    1033    1045    1048    1057    1060
                         1095    1133    1158    1182    1257    1337
PC    =%000007            274#    733*    749*   1716*   1719*   1729*   1734    1812*   1833*   1865*   1888*   2074*   2131*
                         2256    2309*   2316*   2323*   2337*   2339*   2395*
PIRQ  = 177772            260#
PIRQVE= 000240            354#
PR0   = 000000            277#
PR1   = 000040            278#
PR2   = 000100            279#
PR3   = 000140            280#
PR4   = 000200            281#
PR5   = 000240            282#
PR6   = 000300            283#
PR7   = 000340            284#
PS    = 177776            257#    258    1841*
PSW   = 177776            258#   1380*   1419*   1426*   1445*   1455*   1475*   1481*   1508*   1614*   1640*   1652*   1659*
                         1666*   1672*   1680*
PWRMSG  014376           2255    2262#
PWRVEC= 000024             349#    566*    567*   2224*   2225*   2234*   2240*   2252*   2253*
RBEG    001376             531     533#
RBEG1   001462             551#
RBEG2   003342             723#   1699
RDCHR = 104410            2455    2550#
RDLIN = 104411            2551#
RESVEC= 000010             344#
RUNMSG  012272           1838    1891#
R0    =%000000             265#    530*    532*    592    1183*   1187    1191*   1192    1196*   1197    1201*   1202    1206*
                         1207    1211*   1212    1216*   1217    1221*   1222    1226*   1227    1231*   1232    1236*   1237
                         1241*   1259*   1263    1267*   1268    1272*   1273    1277*   1278    1282*   1283    1287*   1288
                         1292*   1293    1297*   1298    1302*   1303    1307*   1308    1312*   1313    1317*   1364*   1386*
                         1499*   1601*   1698*   1726*   1729    1753    1763*   1767    1783    1784    1797*   1847*   1950*
                         1854*   1855    1857*   1862*   1863    1875*   1876*   1877*   1878*   1879    2103    2104*   2105*
                         2112*   2113*   2114*   2115*   2116*   2117    2122    2127*   2129*   2133    2135    2226    2251*
                         2292    2293*   2294    2297*   2518    2519*   2520    2521*   2522    2523*   2524*
R1    =%000001             266#    725*    726*    727*    728    1189*   1190    1194*   1195    1199*   1200    1204*   1205
                         1209*   1210    1214*   1215    1219*   1220    1224*   1225    1229*   1230    1234*   1235    1239*
                         1240    1243    1265*   1266    1270*   1271    1275*   1276    1280*   1281    1285*   1286    1290*
                         1291    1295*   1296    1300*   1301    1305*   1306    1310*   1311    1315*   1316    1319    1754
                         1767*   1768    1772    1796*   1839*   1855*   1859*   1879*   1880*   1881*   1882    2227    2250*
R2    =%000002             267#    536*    538     539*    724*    726     728*    729     730    1755    1766*   1770*   1773
                         1780*   1781*   1782    1787*   1795*   1871*   1882*   2228    2249*
R3    =%000003             268#    537*    540*    543     547*    548    1756    1764*   1765*   1779*   1782*   1791*   1792*
                         1794*   1861*   1862    2175    2184*   2190*   2191*   2194*   2199*   2200*   2201    2210*   2229
                         2248*   2450    2452*   2453    2456*   2457    2464*   2465    2467    2475    2479    2481*   2487
                         2489    2491*   2494*
R4    =%000004             269#   2176    2178*   2179*   2180*   2181    2182*   2196    2198*   2206*   2209*   2230    2247*
R5    =%000005             270#   1757    1759*   1761*   1768*   1772*   1787    1793*   2177    2183*   2185*   2187*   2188*
```

```
                        2189*  2190   2208*  2231   2246*
R6      =%000006         271#   273    554*   555*   556
R7      =%000007         272#   274
SETADD  003402           733    735#   1812   1833
SP      =%000C06         273#   534*   542    558*   576*   584*   588    604*   754*   1418   1424   1444   1453
                        1474   1479   1507   1609   1647   1671   1675   1723*  1753*  1754*  1755*  1756*  1757*
                        1758*  1759   1762*  1775   1777*  1779   1789   1791   1793   1794   1795   1796   1797
                        1799*  1800*  1845   2008*  2011   2013   2014   2042   2043   2047*  2069   2083*  2086*
                        2103*  2108*  2129   2133*  2167*  2168   2169   2170*  2175*  2176*  2177*  2183   2208
                        2209   2210   2211*  2212*  2226*  2227*  2228*  2229*  2230*  2231*  2232*  2233   2241*
                        2245   2246   2247   2248   2249   2250   2251   2256*  2292*  2293   2294*  2296   2297
                        2298*  2300   2302   2304   2310   2312*  2314*  2322*  2326   2330   2331   2335   2355*
                        2356*  2357   2364*  2367*  2368*  2372*  2373*  2377   2380*  2384   2386   2388   2389*
                        2396   2398   2400*  2401   2403*  2404*  2405*  2406*  2407*  2422*  2423*  2426*  2427*
                        2428   2432*  2433*  2434   2437   2439   2441*  2450*  2451*  2456   2459   2463*  2470
                        2474*  2493   2494   2495*  2496*  2497*  2518*  2519   2529*  2530*
STACK = 001100           248#   534    558    754
STKLMT= 177774           259#
SWR     001140           409#   556    578*   580    586*   623    629    642    648    661    667    680    686
                         713    994   1489   1519   1558   1591   1688   2003   2017   2019   2025   2032   2072
                        2077   2081   2232   2245*  2351   2388*
SWREG   000176           376#   586    623    642    661    680    713   2351   2364
SW0     = 000001         312#
SW00    = 000001         302#   312
SW01    = 000002         301#   311
SW02    = 000004         300#   310
SW03    = 000010         299#   309
SW04    = 000020         298#   308
SW05    = 000040         297#   307
SW06    = 000100         296#   306
SW07    = 000200         295#   305
SW08    = 000400         294#   304
SW09    = 001000         293#   303
SW1     = 000002         311#
SW10    = 002000         292#
SW11    = 004000         291#
SW12    = 010000         290#  1688
SW13    = 020000         289#
SW14    = 040000         288#
SW15    = 100000         287#
SW2     = 000004         310#
SW3     = 000010         309#
SW4     = 000020         308#
SW5     = 000040         307#
SW6     = 000100         306#
SW7     = 000200         305#
SW8     = 000400         304#
SW9     = 001000         303#
TBITVE= 000014           345#
TKVEC = 000060           352#
TPVEC = 000064           353#
TRANST  001364           525#   686*   692    753
TRAPVE= 000034           351#   564*   565*
TRTVEC= 000014           346#
TST1    003606           762#
TST10   004150           844#
```

```
TST11    004234            862#
TST12    004332            875     880#
TST13    004430            893     899#
TST14    004516            915#
TST15    004604            930#
TST16    004654            937     943#
TST17    004716            949     955#
TST2     003624            770#
TST20    004754            960     966#
TST21    005016            972     978#
TST22    005060            985     993#
TST23    005150           1005    1011#
TST24    005234           1020    1026#
TST25    005320           1035    1041#
TST26    005404           1050    1055#
TST27    005564           1058    1088#
TST3     003662            775     780#
TST30    005744           1125#
TST31    006062           1150#
TST32    006222           1178#
TST33    006616           1248    1254#
TST34    007212           1324    1329#
TST35    007270           1338    1344#
TST36    007340           1351    1359#
TST37    007406           1366    1376#
TST4     003720            786     792#
TST40    007520           1392    1402#
TST41    007626           1422    1430#
TST42    007762           1449    1451    1459#
TST43    010066           1478    1486#
TST44    010240           1491    1506    1518#
TST45    010430           1520    1557#
TST46    010562           1559    1588#
TST47    010734           1593    1608    1620#
TST5     003756            797     803#
TST50    011030           1637#
TST51    011124           1656#
TST52    011250           1674    1684#
TST53    011724           1811#
TST54    012030           1832#
TST6     004014            808     814#
TST7     004072            828#
TYPDS =  104405           1724    2545#
TYPE  =  104401            596     600     607     611     615     630     634     649     653     668     672     701     705
                          1722    1725    1798    1837    1886    2075    2102    2119    2121    2124    2126    2130    2137
                          2202    2254    2305    2362    2363    2366    2379    2390    2409    2462    2468    2473    2477
                          2482    2483    2485    2488    2492    2541#
TYPOC =  104402           2110    2134    2365    2542#
TYPON =  104404           2544#
TYPOS =  104403            605    2543#
$AUTOB   001134            406#    627*    646*    665*    684*    717*   2359    2508
$BDADR   001122            401#
$DDDAT   001126            403#    773*    774     785*    795*    796     806*    807     819*    820     834*    835     850*
                           851*    852     873*    874     891*    892     905*    906     921*    922     935*    936     947*
                           948     958*    959     970*    971     984*   1002*   1003*   1004    1017*   1018*   1019    1032*
                          1033*   1034    1047*   1048*   1049    1067*   1068    1077*   1078    1101*   1102*   1103    1111*
```

```
                        1112*   1114    1138*   1139    1167*   1168    1243*   1247    1319*   1323    1336*   1337*   1349*
                        1350    1365*   1447*   1448    1524*   1548*   1563*   1981    1984
$CHARC  014660          2307*   2317*   2324    2333*   2338#
$CKSWR  014664          2351#   2549
$CMTAG  001100           389#    553     554     562     568     569     570     730
$CM1   = 000002          421#    422#    423#
$CM2   = 000004          421#    422#    423#
$CM3   = 000002          419#    421
$CNTLG  015536          2362    2503#
$CNTLU  015531          2379    2477    2502#
$CRLF   001173           426#   2075    2093    2102    2121    2126    2130    2306    2341    2390    2482    2502
$DBLK   011714          1764    1798    1806#
$DOAGN  011454          1718    1727    1733#
$DTBL   011704          1767    1802#
$ENDAD  011444          1729#   2088
$ENDCT  011412           568    1720#
$ENDMG  011463          1722    1737#
$ENULL  011460          1725    1736#
$EOP    011356          1697    1710#
$EOPCT  011404           568*   1717#   1721
$ERFLG  001103           392#   1992    2021    2023    2029*   2050    2065*   2093
$ERMAX  001115           398#    571*   2023    2045*   2050
$ERROR  013504           562    2063#
$ERRPC  001116           399#   1981    1983    1984    2069*   2070*   2071    2093    2108
$ERRTB  001176           443#   2116
$ERRTY  013642          2074    2101#
$ERTTL  001112           396#   2068*   2093
$ESCAP  001170           424#    570*   2044*   2084    2086    2093
$FILLC  001156           417#   2310    2341
$FILLS  001155           416#   2341
$GDADR  001120           400#
$GDDAT  001124           402#    689*    690     692     698*    771*    774     782*    793*    796     804*    807     817*
                         818     820     823*    830*    833     835     839*    846*    849     852     856*    863*    864
                         874     881*    882     892     902*    903     904     906     909*    918*    919     920     922
                         925*    933*    936     945*    948     956*    959     968*    971     981*   1000*   1004   1014*
                        1015*   1016    1019    1029*   1030*   1031    1034    1044*   1045*   1046    1049    1059*   1060
                        1062    1064    1066    1068    1073    1076    1078    1083*   1094*   1095    1097    1100    1103
                        1110    1113*   1127*   1131    1133    1136    1139    1143    1144*   1158    1161*   1162*   1163*
                        1164    1168    1180*   1181*   1182*   1183    1247    1256*   1257*   1258*   1259    1323    1334*
                        1345*   1350    1360*   1446*   1525*   1546*   1564*   1981    1984
$GET42  011434          1726#
$GTSWR  014734          2363#   2547
$HD    = 000000          242
$ICNT   001104           393#   2036*   2037    2039*   2049
$ILLUP  014370          2224    2240    2259#
$INTAG  001135           407#   2391    2508
$ITEMB  001114           397#   2071*   2093    2105
$LF     001174           427#   2093    2341    2492    2502
$LPADR  001106           394#    572*   2027*   2042*   2047    2049
$LPERR  001110           395#    573*    816*    829*    845*    901*    917*   1089*   1126*   1151*   1522*   1561*   2027
                        2043*   2049    2083
$MAIL  = ****** U         590     623     642     661     680     713    2042    2077    2294
$MNEW   015554          2366    2506#
$MSWR   015543          2363    2504#
$MXCNT  013502          2040    2049#
$NULL   001154           415#   2312    2341
```

```
$NWTST= 000001          759#  767#  777#  789#  800#  811#  825#  841#  859#  877#  896#  912#  927#
                        940#  952#  963#  975#  990# 1008# 1023# 1038# 1052# 1085# 1122# 1147# 1175#
                       1251# 1326# 1341# 1356# 1373# 1399# 1427# 1456# 1483# 1515# 1554# 1585# 1617#
                       1634# 1653# 1681# 1808# 1829#
$OCNT   014220         2174* 2203* 2216#
$OMODE  014222         2169* 2173* 2178  2181* 2192* 2218#
$OVER   013466         2004  2020  2028  2038  2046#
$PASS   001100          390# 1714* 1715* 1723  1736  2034  2050
$PWRAD  014364         2257#
$PWRDN  014224          566  2224# 2252
$PWRNG  014360         2255#
$PWRUP  014276         2234  2240#
$QUES   001172          425# 2093  2341  2409  2485  2502
$RDCHR  015146         2422# 2550
$RDDEC= ****** U       2552
$RDLIN  015266         2450# 2551
$RDOCT= ****** U       2552
$RDSZ = 000007         2443#
$REGAD  001160          419#
$REG0   001162          421#
$REG1   001164          422#
$RTNAD  011456         1735#
$R2A  = ****** U       2552
$SAVRE= ****** U       2552
$SAVR6  014374         2233* 2241  2242* 2243* 2261#
$SCOPE  013232          560  2001#
$SETUP= 000137          481#  559   560   562   564   566   568   569   570   572   620   639   658
                        677   710  1712  2002  2064  2080  2088  2346  2508
$STUP = 177777          481#
$SVLAD  013440         2012  2041#
$SWR  = 165400          232#  242   359   360   361   362   363   364   365   366   423   424   425
                        569   570   572   573   763   771   781   793   804   815   829   845   863
                        881   900   916   931   944   956   967   979   994  1012  1027  1042  1056
                       1089  1126  1151  1179  1255  1330  1345  1360  1377  1403  1431  1460  1487
                       1519  1558  1589  1621  1638  1657  1685  1707  1713  1728  1734  1736  1812
                       1833  1993  1994  1995  1996  1997  2003  2015  2017  2018  2021  2022  2023
                       2030  2031  2032  2043  2046  2049  2056  2057  2058  2059  2068  2072  2077
                       2081  2093  2258
$SWRMK= 000000          366   367  1997  1998  2019
$TIMES  001166          423#  569*  781*  815*  900*  916*  979* 1056* 1330* 1377* 1621* 1685* 1713*
                       2030* 2037  2040* 2049
$TKB    001146          412# 2344  2355  2372  2426  2432
$TKS    001144          411# 2344  2353  2369  2393* 2424  2430
$TN   = 000055          232#  242   759   763#  767   771#  775   777   781#  786   789   793#  797
                        800   804#  808   811   815#  825   829#  841   845#  859   863#  875   877
                        881#  893   896   900#  912   916#  927   931#  937   940   944#  949   952
                        956#  960   963   967#  972   975   979#  985   990   994#  1005  1008  1012#
                       1020  1023  1027# 1035  1038  1042# 1050  1052  1056# 1058  1085  1089# 1122
                       1126# 1147  1151# 1175  1179# 1248  1251  1255# 1324  1326  1330# 1338  1341
                       1345# 1351  1356  1360# 1366  1373  1377# 1392  1399  1403# 1422  1427  1431#
                       1449  1451  1456  1460# 1478  1483  1487# 1491  1506  1515  1519# 1520  1554
                       1558# 1559  1585  1589# 1593  1608  1617  1621# 1634  1638# 1653  1657# 1674
                       1681  1685# 1808  1812# 1829  1833#
$TPB    001152          414# 2330* 2341
$TPFLG  001157          418# 2288  2341
$TPS    001150          413# 2328  2341
```

```
$TRAP   015566          564    2518#
$TRAP2  015610         2529#   2540
$TRP  = 000012         2533#   2542#  2543#  2544#  2545#  2546#  2547   2548#  2549   2550#  2551#  2552#
$TRPAD  015622         2523    2540#
$TSTNM  001102          391#   1712*  1992   2019   2041*  2046   2050   2067   2093
$TTYIN  015522         2452    2463   2465   2483   2497   2501#
$TYPBN= ****** U       2546
$TYPDS  011500         1752#   2545
$TYPE   014444         2288#   2533   2541
$TYPEC  014614         2309    2316   2323   2328#  2329   2395
$TYPEX  014662         2334    2336   2339#
$TYPOC  014022         2172#   2542
$TYPON  014036         2171    2174#  2544
$TYPOS  013776         2167#   2543
$XTSTR  013244         2006#
$$GET4= 000000         1728#
$OFILL  014221         2168*   2172*  2182   2217#
$40CAT= ****** U       2003    2074
.     = 015650          370#    374#   388#   428    557    572    573    599#   603#   704#   708#   1736   1740
                       1806#   1980#  2049   2050   2093   2140#  2236   2260   2269#  2341   2344   2501#  2502
                       2508#
```

```
CLERIN    990*   999   1013  1028  1043  1154  1185  1261  1332  1530  1569
CLEROT    989*   997   1012  1027  1042  1137  1153  1165  1184  1188  1193  1198  1203  1208  1213
         1218   1223  1228  1233  1238  1260  1264  1269  1274  1279  1284  1289  1294  1299  1304
         1309   1314  1331  1529  1568  1814  1819
CLRVCT    527*   755  1510  1610  1648  1676
COMMEN    355*
ENDCOM    355*
ERROR     249*   694   776   787   798   809   822   837   854   876   894   908   924   938   950
          961   973   986  1006  1021  1036  1051  1070  1080  1105  1116  1141  1170  1249  1325
         1339  1352  1367  1394  1397  1412  1425  1441  1450  1454  1468  1480  1505  1540  1550
         1578  1607  1625  1628  1646  1673
ESCAPE    355*
GETPRI    355*
GETSWR    232*   355*  620   639   658   677   710
MULT      355*
NEWTST    355*   759   767   777   789   800   811   825   841   859   877   896   912   927   940
          952   963   975   990  1008  1023  1038  1052  1085  1122  1147  1175  1251  1326  1341
         1356  1372  1399  1427  1456  1483  1515  1554  1585  1617  1634  1653  1681  1808  1829
POP       355*  1793  2245  2246
PUSH      355*  1752  2226  2232
REPORT    355*
SCOPE     250*   762   770   780   792   803   814   828   844   862   880   899   915   930   943
          955   966   978   993  1011  1026  1041  1055  1088  1125  1150  1178  1254  1329  1344
         1359  1376  1402  1430  1459  1486  1518  1557  1588  1620  1637  1656  1684  1711  1811
         1832
SETPRI    355*
SETTRA   2533*  2542  2543  2544  2545  2547  2549  2550  2551
SETUP     355*   552
SKIP      355*   775   786   797   808   821   875   893   937   949   960   972   985  1005  1020
         1035  1050  1058  1104  1115  1140  1169  1248  1324  1338  1351  1366  1392  1422  1449
         1451  1478  1491  1506  1520  1528  1539  1549  1559  1567  1577  1593  1608  1674
SLASH     355*
SPACE     355*
STARS     355*   384   428   759   761   767   769   777   779   789   791   800   802   811   813
          825   827   841   843   859   861   877   879   896   898   912   914   927   929   940
          942   952   954   963   965   975   977   990   992  1008  1010  1023  1025  1038  1040
         1052  1054  1085  1087  1122  1124  1147  1149  1175  1177  1251  1253  1326  1328  1341
         1343  1356  1358  1373  1375  1399  1401  1427  1429  1456  1458  1483  1485  1515  1517
         1554  1556  1585  1587  1617  1619  1634  1636  1653  1655  1631  1683  1703  1742  1808
         1810  1829  1831  1989  2052  2096  2144  2222  2238  2273  2343  2346  2414  2443  2512
SWRSU     355*   574*
TRMTRP   2533*
TYPBIN    355*
TYPDEC    355*  1723
TYPNAM    355*
TYPNUM    355*
TYPOCS    355*
TYPOCT    355*  2108  2132  2364
TYPTXT    355*   595   600   607   611   615   630   634   649   653   668   672   701   705
$$CMRE    382*   421   422
$$CMTM    382*
$$ESCA    355*
$$NEWT    355*   759   767   777   789   800   811   825   841   859   877   896   912   927   940
          952   963   975   990  1008  1023  1038  1052  1085  1122  1147  1175  1251  1326  1341
         1356  1373  1399  1427  1456  1483  1515  1554  1585  1617  1634  1653  1681  1808  1829
$$SET    2533*  2542  2543  2544  2545  2547  2549  2550  2551
```

DZDRGC.P11   CROSS REFERENCE TABLE -- MACRO NAMES

```
$$SKIP     355*    775     786     797     808     875     893     937     949     960     972     985    1005    1020    1035
          1050    1058    1248    1324    1338    1351    1366    1392    1422    1449    1451    1478    1491    1506    1520
          1559    1593    1608    1674
.EQUAT     232*    245
.HEADE     232*
.SETUP     232*    481
.SWRHI     232*    355
.SWRLO     367*
.$CATC     232*    368
.$CMTA     232*    382
.$EOP      232*   1701
.$ERRO     232*   2050
.$ERRT     232*   2094
.$PARM     232*
.$POWE     232*   2220
.$READ     232*   2341
.$SAVE     232*
.$SCOP     232*   1987
.$SPAC     232*
.$SWDO     232*
.$TRAP     232*   2510
.$TYPD     232*   1740
.$TYPE     232*   2271
.$TYPO     232*   2142
```

DZDRGC.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
ADD      737   739   741   744   746   748  1633  1691  1772  2116  2170  2180  2298  2380  2389
ASL      839   856  1119  1144  1173  1552  1583  2113  2114  2115  2403  2404  2405  2522
ASLB    1777
BCC      699  1084  1778
BEQ      593   691   693   775   786   797   808   821   836   853   875   893   907   923   960
         972   985   995  1005  1020  1035  1050  1058  1061  1069  1079  1098  1104  1115  1140
        1169  1248  1324  1338  1351  1449  1490  1528  1592  1687  1727  1864  2018  2020  2022
        2026  2035  2066  2082  2085  2118  2123  2136  2197  2301  2336  2360  2387  2402  2471
BGE     2038
BGT     1718  1786  2204  2399  2440
BHI     2024
BIC      849  1003  1015  1018  1030  1033  1045  1048  1066  1102  1110  1112  1162  1163  1181
        1182  1257  1258  1337  1503  1533  1544  1572  1605  1623  1715  1857  1880  2194  2356
        2373  2400  2427  2433  2441
BIS      752   753   833  1076  1143  1172  1190  1195  1200  1205  1210  1215  1220  1225  1230
        1235  1240  1266  1271  1276  1281  1286  1291  1296  1301  1306  1311  1316  1410  1439
        1466  1532  1534  1545  1571  1573  1641  1660  1780  1781  1881  2199  2200  2407
BISB    2105
BIT      690   692   936   948   994  1060  1062  1073  1095  1097  1114  1131  1133  1156  1158
        1448  1489  1519  1527  1558  1566  1591  1688  2003  2017  2025  2032  2072  2081
BITB    2324
BLO     2466
BLOS    2454
BLT     1769  1785  2205  2315  2397  2438
BMI     1366  1539  1776
BNE      544   557   581   591   622   624   641   643   660   662   679   681   712   714   731
         824   840   857   910   926   937   949  1063  1074  1096  1120  1132  1134  1145  1157
        1159  1174  1520  1553  1559  1567  1584  1624  1627  1689  1774  1824  1851  1853  1858
        1884  2004  2033  2073  2089  2106  2128  2195  2244  2295  2303  2311  2325  2332  2352
        2358  2378  2385  2392  2429  2435  2458  2460  2476  2480  2490
BPL     1549  1577  1760  1790  2078  2193  2289  2329  2354  2370  2425  2431
BR       531   541   583   597   601   608   612   616   626   631   635   645   650   654   664
         669   673   683   702   706   716   734  1392  1422  1451  1478  1491  1506  1593  1608
        1670  1674  1693  1771  1788  1825  1843  1866  1873  2006  2012  2015  2028  2031  2111
        2138  2171  2186  2207  2236  2260  2291  2308  2318  2327  2334  2381  2408  2410  2436
        2469  2478  2484  2486
CLR      530   537   550   555   569   570   727   751   756   758   763   764   765   782   817
         832   931   944   967   980   981   998  1000  1012  1027  1042  1113  1128  1137  1153
        1165  1184  1188  1193  1198  1203  1208  1213  1218  1223  1228  1233  1238  1260  1264
        1269  1274  1279  1284  1289  1294  1299  1304  1309  1314  1331  1334  1346  1361  1379
        1380  1391  1405  1419  1426  1433  1445  1446  1455  1461  1475  1481  1492  1496  1508
        1509  1511  1513  1524  1529  1531  1535  1546  1547  1564  1568  1570  1574  1594  1598
        1611  1613  1614  1615  1638  1649  1651  1652  1657  1666  1672  1677  1679  1680  1712
        1713  1763  1766  1815  1819  1834  1841  2030  2044  2104  2184  2242  2367  2368  2451
        2474
CLRB    1792  2029  2307  2333  2481  2491
CMP      542   556   580   623   642   661   680   713   730   774   796   807   820   835   852
         874   892   906   922   959   971  1004  1019  1034  1049  1068  1078  1103  1139  1168
        1247  1323  1348  1350  1363  1418  1424  1444  1453  1474  1479  1507  1609  1626  1647
        1671  1675  1784  1845  1863  2013  2037  2088  2351  2357  2377  2384  2396  2398  2428
        2434  2437  2439  2453  2465
CMPB    2019  2023  2300  2302  2310  2331  2335  2359  2391  2457  2475  2479  2489
COM      851   865   866   867   868   869   870   871   872   883   884   885   886   887   888
         889   890  1093  1191  1196  1201  1206  1211  1216  1221  1226  1231  1236  1241  1267
        1272  1277  1282  1287  1292  1297  1302  1307  1312  1317
DEC      547  1692  1716  1823  1850  1852  1883  2112  2464
```

```
DECB    2192  2203  2314  2317
EMT      249
HALT     374   545   619   638   657   676   709  2079  2090  2235  2259  2290
INC      540   823  1714  1770  2036  2068  2198  2206  2243  2406
INCB     909   925  2041  2065  2337
IOT      250
JMP      378   379   380   381   594   696   996  1697  1699  1734
JSR      733  1729  1812  1833  1865  2074  2309  2316  2323  2395
MOV      532   534   535   536   548   549   554   558   560   561   562   563   564   565   566
         567   568   572   573   576   577   578   579   584   586   587   588   604   629   648
         667   686   689   720   721   722   724   725   726   728   735   736   738   740   742
         743   745   747   750   754   755   757   771   772   773   781   783   785   793   794
         795   804   805   806   815   816   818   819   829   830   834   845   846   848   850
         863   864   873   881   882   891   900   901   902   903   905   916   917   918   919
         921   932   933   934   935   945   946   947   956   957   958   968   969   970   979
         982   984   999  1001  1002  1013  1014  1016  1017  1028  1029  1031  1032  1043  1044
        1046  1047  1056  1059  1064  1065  1067  1075  1077  1089  1091  1092  1094  1100  1101
        1111  1126  1127  1129  1136  1138  1151  1152  1154  1161  1164  1167  1180  1183  1185
        1187  1189  1192  1194  1197  1199  1202  1204  1207  1209  1212  1214  1217  1219  1222
        1224  1227  1229  1232  1234  1237  1239  1243  1256  1259  1261  1263  1265  1268  1270
        1273  1275  1278  1280  1283  1285  1288  1290  1293  1295  1298  1300  1303  1305  1308
        1310  1313  1315  1319  1330  1332  1333  1336  1345  1347  1349  1360  1362  1364  1365
        1377  1381  1382  1383  1384  1385  1386  1406  1407  1408  1409  1417  1434  1435  1436
        1437  1438  1443  1447  1462  1463  1464  1465  1473  1493  1494  1495  1497  1498  1499
        1510  1512  1522  1523  1525  1530  1561  1562  1563  1569  1595  1596  1597  1599  1600
        1601  1610  1612  1621  1629  1631  1632  1639  1640  1648  1650  1658  1659  1665  1676
        1678  1685  1694  1695  1696  1698  1719  1723  1726  1753  1754  1755  1756  1757  1758
        1759  1764  1767  1787  1793  1794  1795  1796  1797  1799  1800  1813  1816  1817  1818
        1820  1821  1822  1835  1836  1839  1840  1846  1847  1854  1855  1856  1859  1861  1862
        1870  1871  1872  1879  2008  2009  2011  2014  2027  2039  2040  2042  2043  2046  2047
        2067  2069  2083  2086  2103  2108  2117  2122  2127  2129  2133  2167  2175  2176  2177
        2183  2190  2208  2209  2210  2211  2212  2224  2225  2226  2227  2228  2229  2230  2231
        2232  2233  2234  2240  2241  2245  2246  2247  2248  2249  2250  2251  2252  2253  2256
        2292  2293  2297  2312  2364  2388  2393  2422  2423  2450  2452  2463  2494  2495  2496
        2497  2518  2519  2523  2529  2530
MOVB     571   627   646   665   684   717   904   920  1548  1762  1765  1779  1782  1791  1882
        2045  2071  2168  2169  2172  2173  2174  2178  2181  2182  2201  2294  2322  2330  2355
        2372  2426  2432  2456  2461  2467  2472  2487  2521
NEG     1761  2179
NOP     1244  1245  1246  1320  1321  1322  1387  1388  1389  1390  1403  1404  1411  1420  1421
        1431  1432  1440  1460  1467  1476  1477  1487  1488  1500  1501  1502  1504  1589  1590
        1602  1603  1604  1606  1642  1643  1644  1645  1661  1662  1663  1667  1668  1669  1730
        1731  1732
RESET    533   551   719   723   732   784   983  1335  1378  1622  1728
ROL      698  1083  2185  2187  2188  2189  2191
ROR     1875  1876  1877  1878
RTI      585  1395  1398  1801  2048  2092  2213  2258  2299  2394  2442  2498  2531
RTS      749  1888  2131  2339  2524
SUB      539  1630  1690  1768  2070
SWAB    1885
TRAP    2533  2542  2543  2544  2545  2547  2549  2550  2551
TST      538   543   590   592   621   640   659   678   711   729  1057  1686  1773  1783  2010
        2034  2077  2084  2135  2196  2296  2304  2326  2386  2401  2459  2470  2493  2520
TSTB    1538  1576  1775  1789  2021  2288  2328  2353  2369  2424  2430
WAIT    1842
.ASCII   425   426  1894
```

```
.ASCIZ   427  599  603  610  614  618  633  637  652  656  671  675  704  708 1737
        1891 1899 1904 1909 1914 1919 1924 1928 1933 1942 1951 1957 1963 1969 1972
        2139 2262 2502 2503 2504 2506
.BLKB   2501
.BLKW   1806
.BYTE    391  392  397  398  406  407  415  416  417  418 1736 1696 2214 2215 2216
        2217 2499 2500
.DSABL  2411
.ENABL   232 2344
.END    2554
.ENDC    237  249  341  355  363  365  366  367  379  385  389  391  419  423  424
         425  429  481  558  559  562  564  566  568  569  570  572  574  590  599
         603  610  614  618  623  629  633  637  642  648  652  656  661  667  671
         675  680  686  704  708  713  719  760  761  762  763  768  769  770  771
         776  778  779  780  781  782  787  790  791  792  793  798  801  802  803
         804  809  812  813  814  815  816  822  826  827  828  829  842  843  844
         845  860  861  862  863  876  878  879  880  881  894  897  898  899  900
         901  913  914  915  916  917  928  929  930  931  938  941  942  943  344
         950  953  954  955  956  961  964  965  966  967  973  976  977  978  979
         980  986  991  992  993  994 1006 1009 1010 1011 1012 1021 1024 1025 1026
        1027 1036 1039 1040 1041 1042 1051 1053 1054 1055 1056 1057 1059 1086 1087
        1088 1089 1105 1116 1123 1124 1125 1126 1141 1148 1149 1150 1151 1170 1176
        1177 1178 1179 1249 1252 1253 1254 1255 1325 1327 1328 1329 1330 1331 1339
        1342 1343 1344 1345 1352 1357 1358 1359 1360 1367 1374 1375 1376 1377 1378
        1393 1400 1401 1402 1403 1423 1428 1429 1430 1431 1450 1452 1457 1458 1459
        1460 1479 1484 1485 1486 1487 1492 1507 1516 1517 1518 1519 1521 1529 1540
        1550 1555 1556 1557 1558 1560 1568 1578 1586 1587 1588 1589 1594 1609 1618
        1619 1620 1621 1622 1635 1636 1637 1638 1654 1655 1656 1657 1675 1682 1683
        1684 1685 1686 1704 1706 1707 1709 1712 1718 1721 1722 1726 1728 1734 1736
        1737 1740 1743 1809 1810 1811 1812 1830 1831 1832 1833 1990 1993 1998 2003
        2005 2016 2019 2020 2021 2023 2025 2032 2036 2041 2042 2046 2049 2050 2053
        2056 2065 2069 2074 2075 2076 2077 2088 2092 2093 2097 2112 2141 2145 2223
        2232 2233 2239 2245 2246 2256 2258 2262 2274 2294 2344 2345 2347 2375 2411
        2415 2443 2444 2452 2454 2457 2485 2502 2508 2513 2519 2522 2541 2542 2543
        2544 2545 2546 2547 2548 2549 2550 2551 2552
.EQUIV   249  250  258  273  274  303  304  305  306  307  308  309  310  311  312
         331  332  333  334  335  336  337  338  339  340
.EVEN    599  603  610  614  618  633  637  652  656  671  675  704  708 1897 1980
        2140 2269 2508
.IF      233  247  313  341  362  364  365  366  367  377  384  388  390  419  423
         424  425  428  429  481  553  558  560  562  564  566  568  569  570  572
         590  598  602  609  613  617  620  623  632  636  639  642  651  655  658
         661  670  674  677  680  703  707  710  713  759  761  763  767  769  771
         775  777  779  781  782  786  789  791  793  797  800  802  804  808  811
         813  815  816  821  825  827  829  841  843  845  859  861  863  875  877
         879  881  893  896  898  900  901  912  914  916  917  927  929  931  937
         940  942  944  949  952  954  956  960  963  965  967  972  975  977  979
         980  985  990  992  994 1005 1008 1010 1012 1020 1023 1025 1027 1035 1038
        1040 1042 1050 1052 1054 1056 1057 1058 1085 1087 1089 1104 1115 1122 1124
        1126 1140 1147 1149 1151 1169 1175 1177 1179 1248 1251 1253 1255 1324 1326
        1328 1330 1331 1338 1341 1343 1345 1351 1356 1358 1360 1366 1373 1375 1377
        1378 1392 1399 1401 1403 1422 1427 1429 1431 1449 1451 1456 1458 1460 1478
        1483 1485 1487 1491 1506 1515 1517 1519 1520 1528 1539 1549 1554 1556 1558
        1559 1567 1577 1585 1587 1589 1593 1608 1617 1619 1621 1622 1634 1636 1638
        1653 1655 1657 1674 1681 1683 1685 1686 1703 1704 1705 1706 1707 1708 1709
        1711 1717 1720 1722 1726 1728 1734 1736 1737 1742 1808 1810 1812 1829 1831
```

	1833	1989	1992	1997	2003	2015	2017	2018	2019	2021	2022	2023	2032	2034	2042
	2043	2048	2049	2050	2052	2055	2065	2068	2072	2074	2075	2077	2081	2088	2092
	2093	2096	2111	2127	2144	2222	2232	2233	2238	2245	2246	2254	2256	2258	2262
	2273	2294	2343	2345	2346	2347	2375	2414	2415	2443	2451	2453	2457	2458	2501
	2502	2508	2512	2518	2522	2533	2542	2543	2544	2545	2546	2547	2549	2550	2551
	2552														
.IFF	247	362	364	366	367	385	388	390	419	429	558	760	761	762	763
	768	769	770	771	776	778	779	780	781	787	790	791	792	793	798
	801	802	803	804	809	812	813	814	815	821	826	827	828	829	842
	843	844	845	860	861	862	863	876	878	879	880	881	894	897	898
	899	900	913	914	915	916	928	929	930	931	938	941	942	943	944
	950	953	954	955	956	961	964	965	966	967	973	976	977	978	979
	986	991	992	993	994	1006	1009	1010	1011	1012	1021	1024	1025	1026	1027
	1036	1039	1040	1041	1042	1051	1053	1054	1055	1056	1057	1059	1086	1087	1088
	1089	1104	1115	1123	1124	1125	1126	1140	1148	1149	1150	1151	1169	1176	1177
	1178	1179	1249	1252	1253	1254	1255	1325	1327	1328	1329	1330	1339	1342	1343
	1344	1345	1352	1357	1358	1359	1360	1367	1374	1375	1376	1377	1393	1400	1401
	1402	1403	1423	1428	1429	1430	1431	1450	1452	1457	1458	1459	1460	1479	1484
	1485	1486	1487	1492	1507	1516	1517	1518	1519	1521	1528	1539	1549	1555	1556
	1557	1558	1560	1567	1577	1586	1587	1588	1589	1594	1609	1618	1619	1620	1621
	1622	1635	1636	1637	1638	1654	1655	1656	1657	1675	1682	1683	1684	1685	1686
	1704	1708	1712	1717	1720	1736	1743	1809	1810	1811	1812	1830	1831	1832	1833
	1990	2016	2019	2020	2023	2049	2050	2053	2055	2069	2088	2093	2097	2112	2141
	2145	2223	2239	2254	2274	2344	2347	2415	2417	2422	2443	2444	2453	2485	2501
	2513	2519													
.IFT	599	603	610	614	618	633	637	652	656	671	675	704	708	2031	2075
	2417	2422													
.IFTF	599	603	610	614	618	633	637	652	656	671	675	704	708	2029	2074
	2362	2415	2418												
.IIF	232	237	242	359	360	361	363	366	367	374	428	559	562	568	569
	570	572	573	1706	1712	1713	1724	1736	1740	1993	1994	1995	1996	1997	1998
	2002	2030	2031	2046	2049	2050	2056	2057	2058	2059	2064	2080	2088	2093	2109
	2134	2341	2344	2365	2493	2502	2508	2541	2542	2543	2544	2545	2547	2549	2550
	2551														
.IRP	481	759	767	777	789	800	811	825	841	859	877	896	912	927	940
	952	963	975	990	1008	1023	1038	1052	1085	1122	1147	1175	1251	1326	1341
	1356	1373	1399	1427	1456	1483	1515	1554	1585	1617	1634	1653	1681	1753	1793
	1808	1829	2226	2232	2245	2246									
.LIST	232	355	366	374	419	421	422	423	481	574	599	603	610	614	618
	633	637	652	656	671	675	704	708	759	763	767	771	777	781	789
	793	800	804	811	815	825	829	841	845	859	863	877	881	896	900
	912	916	927	931	940	944	952	956	963	967	975	979	990	994	1008
	1012	1023	1027	1038	1042	1052	1056	1085	1089	1122	1126	1147	1151	1175	1179
	1251	1255	1326	1330	1341	1345	1356	1360	1373	1377	1399	1403	1427	1431	1456
	1460	1483	1487	1515	1519	1554	1558	1585	1589	1617	1621	1634	1638	1653	1657
	1681	1685	1712	1728	1808	1812	1829	1833	1997	2088	2443	2533	2541	2542	2543
	2544	2545	2546	2547	2548	2549	2550	2551	2552						
.MACRO	367	382	527	989	990	2533									
.MCALL	232	355	574												
.NLIST	232	355	366	374	419	421	422	423	481	574	599	603	610	614	618
	633	637	652	656	671	675	704	708	759	763	767	771	777	781	789
	793	800	804	811	815	825	829	841	845	859	863	877	881	896	900
	912	916	927	931	940	944	952	956	963	967	975	979	990	994	1008
	1012	1023	1027	1038	1042	1052	1056	1085	1089	1122	1126	1147	1151	1175	1179
	1251	1255	1326	1330	1341	1345	1356	1360	1373	1377	1399	1403	1427	1431	1456
	1460	1483	1487	1515	1519	1554	1558	1585	1589	1617	1621	1634	1638	1653	1657

DZDRGC.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
          1681    1685    1712    1728    1808    1812    1829    1833    1997    2088    2443    2533    2541    2542    2543
          2544    2545    2546    2547    2548    2549    2550    2551    2552
.PAGE      355     382     429
.REM         1
.REPT      374     421     865     883    1187    1263
.SBTTL     245     355     368     377     382     429     552     620     639     658     677     688     710     759     767
           777     789     800     811     825     841     859     877     896     912     927     940     952     963     975
           990    1008    1023    1038    1052    1085    1122    1147    1175    1251    1326    1341    1356    1373    1399
          1427    1456    1483    1515    1554    1585    1617    1634    1653    1681    1701    1740    1808    1829    1987
          2050    2094    2142    2220    2271    2341    2510    2533
.TITLE     232
.WCRD      374     375     376     390     393     394     395     396     399     400     401     402     403     404     405
           408     409     410     419     421     422     606    1717    1720    1735    2120    2125    2218    2255    2257
          2338    2540
```

```
ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*,DZDRGC.SEQ/SOL/CRF=DZDRGC.P11
RUN-TIME:  50 31 6 SECONDS
RUN-TIME RATIO:  317/88=3.5
CORE USED:  24K  (47 PAGES)
```

Spooler runtime 11 Seconds, 51 KCS, 299 disk reads, 4 disk writes, 74 pages

[redacted]

