IIST

STAND-ALONE SPECIAL
CRIIAA0

AH-F173A-MC

FICHE 1 OF 1

JAN 1979
digital
MADE IN USA

```
        .NLIST  SEQ,LOC,BIN,TOC
        .REM_
```

IDENTIFICATION

PRODUCT CODE: AC-F172A-MC

PRODUCT NAME: CRIIAA0 :IST STND-ALN SP

DATE CREATED: 15 NOVEMBER 1978

MAINTAINER: C.S.S. LOW VOLUME PRODUCTS GROUP

AUTHOR: ROBERT J. COLLINS / W. WEISKE

1.0 ABSTRACT

THE DIP11-A STAND-ALONE DIAGNOSTIC IS USED TO EXERCISE
THE LOGIC OF A DIP11-A INTERFACE (IIST) IN STANDALONE MODE
AND REPORT ANY MALFUNCTIONS ON THE CONSOLE TERMINAL.
MOST OF THE TESTING IS PERFORMED IN WRAPAROUND MAINTENENCE
MODE, BUT SOME TESTS CAN BE EXECUTED IN NORMAL MODE.
THE INTERFACE UNDER TEST MAY BE CONNECTED TO THE IIST
INTERCONNECTING BUS AS LONG AS OTHER INTERFACES ARE NOT
TRANSMITTING INFORMATION ADDRESSED TO THE INTERFACE UNDER
TEST.

2.0 MINIMUM EQUIPMENT

A. PDP-11 WITH 8K AND A CONSOLE TERMINAL
B. DIP11-A (IIST) INTERFACE

3.0 PRELIMINARY OPERATIONS

IF THE INTERFACE IS CONNECTED TO THE IIST BUS, THE OPERATOR
MAY WISH TO PREPARE THE OTHER INTERFACES FOR THE RUNNING OF THE
DIAGNOSTIC ON THIS INTERFACE. SUCH PREPARATION MIGHT INCLUDE
INITIALIZATION OF THE OTHER INTERFACES OR ACTIVATION OF THE
VARIOUS EXTERNAL INHIBIT CONTROLS (IF AVAILABLE) IN ORDER TO
PREVENT INTERFERENCE WITH THE ONLINE INTERFACES.

4.0 LOADING PROCEDURE

IF A PAPER TAPE IS SUPPLIED, LOAD WITH AN ABS LOADER.
IF THE PROGRAM IS AVAILABLE ON XXDP MEDIUM, FOLLOW THE
LOADING PROCEDURES FOR THE MEDIUM.

5.0 CONSOLE SWITCH SETTINGS

SR15 (1)	HALT ON ERROR
SR14 (1)	LOOP ON TEST
SR13 (1)	INHIBIT TYPEOUTS
SR10 (1)	BELL ON ERROR
SR09 (1)	LOOP ON ERROR
SR00 (1)	PRINT ALL IIST REGISTERS ON THE CONSOLE TERMINAL

COMPUTERS WITHOUT A HARDWARE SWITCH REGISTER HAVE A SOFTWARE
SWITCH REGISTER LOCATION IN MEMORY CALLED "SWREG" (LOCATION 176).
THIS LOCATION CAN BE CHANGED MANUALLY OR BY TYPING THE "CNTRL & G"
KEYS AND RESPONDING TO THE RESULTING TERMINAL DIALOGUE.

6.0 STARTING PROCEDURE

LOADING ADDRESS 200(8) AND STARTING WILL IDENTIFY THE PROGRAM, INITIALIZE THE SYSTEM, AND BEGIN TESTING USING THE ADDRESS AND CONFIGURATION PARAMETERS ALREADY STORED WITHIN THE PROGRAM (SUCH AS THE DEFAULT PARAMETERS SUPPLIED UPON INITIAL LOAD, OR THE PARAMETERS SUPPLIED VIA THE CONSOLE ON A PREVIOUS START FROM LOCATION 204).

LOADING ADDRESS 204(8) AND STARTING WILL INITIATE A CONSOLE DIALOG IN WHICH THE OPERATOR MAY SUPPLY NEW OPERATING PARAMETERS (DEVICE ADDRESS, INTERRUPT VECTOR AND LEVEL, AND HARDWARE CONFIGURATION).

7.0 OPERATION

ONCE STARTED, THE PROGRAM RUNS CONTINUOUSLY AND PERIODICALLY PRINTS END OF PASS MESSAGES. THE HARDWARE SWITCH SELF ID CODE IS REPORTED ON PASS 1, AS IS THE SELECTED SANITY-TIMER COUNT RATE. THE OPERATOR SHOULD VERIFY THAT THESE ARE THE EXPECTED PARAMETERS.

8.0 ERRORS

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE AND RELEVANT TEST DATA IN TABULAR FORM. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED, IF NEEDED, FROM THE COMMENT AT THE ERROR PC OR FROM THE TEST ITSELF.

9.0 LISTING

```
140 -                           .LIST   LOC,BIN,SEQ
```

```
                                .ENABLE ABS,AMA
                        .TITLE  MAINDEC-11-CRIIA
                        ;*COPYRIGHT (C) 1977
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY ROBERT J. COLLINS
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C1),MAR 24, 1976.
                        ;*
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        11            INHIBIT ITERATIONS
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        .SBTTL  BASIC DEFINITIONS

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

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

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

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
```

```
(1)  000140                PR3=    140             ;;PRIORITY LEVEL 3
(1)  000200                PR4=    200             ;;PRIORITY LEVEL 4
(1)  000240                PR5=    240             ;;PRIORITY LEVEL 5
(1)  000300                PR6=    300             ;;PRIORITY LEVEL 6
(1)  000340                PR7=    340             ;;PRIORITY LEVEL 7
(1)
(1)                        ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
(1)  100000                SW15=   100000
(1)  040000                SW14=   40000
(1)  020000                SW13=   20000
(1)  010000                SW12=   10000
(1)  004000                SW11=   4000
(1)  002000                SW10=   2000
(1)  001000                SW09=   1000
(1)  000400                SW08=   400
(1)  000200                SW07=   200
(1)  000100                SW06=   100
(1)  000040                SW05=   40
(1)  000020                SW04=   20
(1)  000010                SW03=   10
(1)  000004                SW02=   4
(1)  000002                SW01=   2
(1)  000001                SW00=   1
(1)                        .EQUIV  SW09,SW9
(1)                        .EQUIV  SW08,SW8
(1)                        .EQUIV  SW07,SW7
(1)                        .EQUIV  SW06,SW6
(1)                        .EQUIV  SW05,SW5
(1)                        .EQUIV  SW04,SW4
(1)                        .EQUIV  SW03,SW3
(1)                        .EQUIV  SW02,SW2
(1)                        .EQUIV  SW01,SW1
(1)                        .EQUIV  SW00,SW0
(1)
(1)                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)  100000                BIT15=  100000
(1)  040000                BIT14=  40000
(1)  020000                BIT13=  20000
(1)  010000                BIT12=  10000
(1)  004000                BIT11=  4000
(1)  002000                BIT10=  2000
(1)  001000                BIT09=  1000
(1)  000400                BIT08=  400
(1)  000200                BIT07=  200
(1)  000100                BIT06=  100
(1)  000040                BIT05=  40
(1)  000020                BIT04=  20
(1)  000010                BIT03=  10
(1)  000004                BIT02=  4
(1)  000002                BIT01=  2
(1)  000001                BIT00=  1
(1)                        .EQUIV  BIT09,BIT9
(1)                        .EQUIV  BIT08,BIT8
(1)                        .EQUIV  BIT07,BIT7
(1)                        .EQUIV  BIT06,BIT6
(1)                        .EQUIV  BIT05,BIT5
```

```
(1)                             .EQUIV  BIT04,BIT4
(1)                             .EQUIV  BIT03,BIT3
(1)                             .EQUIV  BIT02,BIT2
(1)                             .EQUIV  BIT01,BIT1
(1)                             .EQUIV  BIT00,BIT0
(1)
(1)                             ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
(1)           000004            ERRVEC= 4                 ;;TIME OUT AND OTHER ERRORS
(1)           000010            RESVEC= 10                ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)           000014            TBITVEC=14                ;;''T'' BIT
(1)           000014            TRTVEC= 14                ;;TRACE TRAP
(1)           000014            BPTVEC= 14                ;;BREAKPOINT TRAP (BPT)
(1)           000020            IOTVEC= 20                ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)           000024            PWRVEC= 24                ;;POWER FAIL
(1)           000030            EMTVEC= 30                ;;EMULATOR TRAP (EMT) **ERROR**
(1)           000034            TRAPVEC=34                ;;''TRAP'' TRAP
(1)           000060            TKVEC=  60                ;;TTY KEYBOARD VECTOR
(1)           000064            TPVEC=  64                ;;TTY PRINTER VECTOR
(1)           000240            PIRQVEC=240               ;;PROGRAM INTERRUPT REQUEST VECTOR
```

```
157
158                                     .SBTTL  MEMORY MANAGEMENT DEFINITIONS
(1)
(1)                                     ;*KT11 VECTOR ADDRESS
(1)
(1)             000250                  MMVEC=  250
(1)
(1)                                     ;*KT11 STATUS REGISTER ADDRESSES
(1)
(1)             177572                  SR0=    177572
(1)             177574                  SR1=    177574
(1)             177576                  SR2=    177576
(1)             172516                  SR3=    172516
(1)
(1)                                     ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
(1)
(1)             172300                  KIPDR0= 172300
(1)             172302                  KIPDR1= 172302
(1)             172304                  KIPDR2= 172304
(1)             172306                  KIPDR3= 172306
(1)             172310                  KIPDR4= 172310
(1)             172312                  KIPDR5= 172312
(1)             172314                  KIPDR6= 172314
(1)             172316                  KIPDR7= 172316
(1)
(1)                                     ;*KERNEL "I" PAGE ADDRESS REGISTERS
(1)
(1)             172340                  KIPAR0= 172340
(1)             172342                  KIPAR1= 172342
(1)             172344                  KIPAR2= 172344
(1)             172346                  KIPAR3= 172346
(1)             172350                  KIPAR4= 172350
(1)             172352                  KIPAR5= 172352
(1)             172354                  KIPAR6= 172354
(1)             172356                  KIPAR7= 172356
(1)
159                                     .SBTTL  TRAP CATCHER
(1)
(1)             000000                          .=0
(1)                                     ;*ALL UNUSED LOCATIONS OF THE VECTOR AREA CONTAIN
(1)                                     ;*A ".+2, IOT" SEQUENCE TO CATCH AND PROCESS ILLEGAL
(1)                                     ;*TRAPS AND INTERRUPTS THAT MIGHT OCCUR.
(1)                                     ;*THE IOT TRAP WHICH IS TAKEN ON THE ILLEGAL TRAP/INT
(1)                                     ;*TRAPS TO THE $SCOPE ROUTINE WHICH (IF THE RETURN PC IS
(1)                                     ;*LESS THAN 1002) JUMPS TO THE $ERROR ROUTINE.
(1)                                     ;*THE $ERROR ROUTINE WILL REPORT THE ERROR AS FOLLOWS:
(1)                                     ;*      PC=YYYYYY UNEXPECTED TRAP TO XXX
(1)                                     ;*AND RETURN TO THE PROGRAM AT PC=YYYYYY+2
(1)                                     ;*WHERE XXX=LOCATION OF ILLEGAL TRAP
(1)                                     ;*      YYYYYY=PC AT TIME OF TRAP
(1)                                     ;*NOTE: IF THE PROCESSOR IS NOT AN 11/05 THE PROGRAM
(1)                                     ;*      CAN BE STARTED AT ADDRESS 0 AS WELL AS ADDRESS 200.
(1)
(1)  000000  000000                     $40CAT: HALT                    ;;HALT
(1)  000002  000737                             BR      .-100           ;;BRANCH TO 177700 & TIME OUT (NOT ON
(1)                                                                     ;;11/05)
```

```
    (1)  000004  002502                    .WORD   NOPAR           ;;VECTOR TO STARTING ADDRESS
    (1)  000006  000340                    .WORD   340             ;;WITH PRIORITY LEVEL 7
    (1)          000174                    .=174
    (1)  000174  000000          DISPREG:  .WORD   0               ;;SOFTWARE DISPLAY REGISTER
    (1)  000176  000000          SWREG:    .WORD   0               ;;SOFTWARE SWITCH REGISTER
    (1)                          .SBTTL  STARTING ADDRES(ES)
    (1)  000200  000137  002502            JMP     @#NOPAR ;;GO TO START OF PROGRAM
    160  000204  000137  002512            JMP     SELPAR          ;GO TO SELECT PARAMETERS
    161
```

```
                                        .SBTTL  ACT11 HOOKS

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

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

(1)  001000                             $APTHD:
(1)  001000  000000                     $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(1)  001002  001230                     $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
(1)  001004  000017                     $TSTM:  .WORD   15.     ;;RUN TIM OF LONGEST TEST
(1)  001006  000074                     $PASTM: .WORD   60.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(1)  001010  000000                     $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(1)  001012  000052                             .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
   168                                  .SBTTL  COMMON TAGS
   (1)
   (2)                                  ;;**********************************************************************
   (1)                                  ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
   (1)                                  ;*USED IN THE PROGRAM.
   (1)
   (1)          001100                          .=1100
   (1)  001100                          $CMTAG:                 ;;START OF COMMON TAGS
   (1)  001100  000000                          .WORD   0
   (1)  001102     000                  $TSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
   (1)  001103     000                  $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
   (1)  001104  000000                  $ICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
   (1)  001106  000000                  $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
   (1)  001110  000000                  $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
   (1)  001112  000000                  $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
   (1)  001114     000                  $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
   (1)  001115     001                  $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
   (1)  001116  000000                  $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
   (1)  001120  000000                  $GDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
   (1)  001122  000000                  $BDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA.
   (1)  001124  000000                  $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
   (1)  001126  000000                  $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
   (1)  001130  000000                          .WORD   0       ;;RESERVED--NOT TO BE USED
   (1)  001132  000000                          .WORD   0
   (1)  001134     000                  $AUTOB: .BYTE   0       ;;AUTOMATIC MODE INDICATOR
   (1)  001135     000                  $INTAG: .BYTE   0       ;;INTERRUPT MODE INDICATOR
   (1)  001136  000000                          .WORD   0
   (1)  001140  177570                  SWR:    .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
   (1)  001142  177570                  DISPLAY: .WORD  DDISP   ;;ADDRESS OF DISPLAY REGISTER
   (1)  001144  177560                  $TKS:   177560          ;;TTY KBD STATUS
   (1)  001146  177562                  $TKB:   177562          ;;TTY KBD BUFFER
   (1)  001150  177564                  $TPS:   177564          ;;TTY PRINTER STATUS REG. ADDRESS
   (1)  001152  177566                  $TPB:   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
   (1)  001154     000                  $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
   (1)  001155     002                  $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
   (1)  001156     012                  $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
   (1)  001157     000                  $TPFLG: .BYTE   0       ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
   (1)  001160  000000                  $REGAD: .WORD   0       ;;CONTAINS THE ADDRESS FROM
   (1)                                                          ;;WHICH  ($REG0) WAS OBTAINED
   (3)  001162  000000                  $REG0:  .WORD   0       ;;CONTAINS (($REGAD)+0)
   (3)  001164  000000                  $REG1:  .WORD   0       ;;CONTAINS (($REGAD)+2)
   (3)  001166  000000                  $REG2:  .WORD   0       ;;CONTAINS (($REGAD)+4)
   (3)  001170  000000                  $REG3:  .WORD   0       ;;CONTAINS (($REGAD)+6)
   (3)  001172  000000                  $REG4:  .WORD   0       ;;CONTAINS (($REGAD)+10)
   (3)  001174  000000                  $REG5:  .WORD   0       ;;CONTAINS (($REGAD)+12)
   (3)  001176  000000                  $TMP0:  .WORD   0       ;;USER DEFINED
   (3)  001200  000000                  $TMP1:  .WORD   0       ;;USER DEFINED
   (3)  001202  000000                  $TMP2:  .WORD   0       ;;USER DEFINED
   (3)  001204  000000                  $TMP3:  .WORD   0       ;;USER DEFINED
   (3)  001206  000000                  $TMP4:  .WORD   0       ;;USER DEFINED
   (3)  001210  000000                  $TMP5:  .WORD   0       ;;USER DEFINED
   (3)  001212  000000                  $TMP6:  .WORD   0       ;;USER DEFINED
   (1)  001214  000000                  $TIMES: 0               ;;MAX. NUMBER OF ITERATIONS
   (1)  001216  000000                  $ESCAPE:0               ;;ESCAPE ON ERROR ADDRESS
   (1)  001220  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
   (1)  001224     077                  $QUES:  .ASCII  /?/     ;;QUESTION MARK
```

```
(1)  001225  015                $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
(1)  001226  000012             $LF:    .ASCIZ  <12>            ;;LINE FEED
(2)                             ;;*****************************************************************
(2)                             .SBTTL  APT MAILBOX-ETABLE
(2)
(3)                             ;;*****************************************************************
(2)                             .EVEN
(2)  001230                     $MAIL:                          ;;APT MAILBOX
(2)  001230  000000             $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
(2)  001232  000000             $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
(2)  001234  000000             $TESTN: .WORD   ATESTN  ;;TEST NUMBER
(2)  001236  000000             $PASS:  .WORD   APASS   ;;PASS COUNT
(2)  001240  000000             $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
(2)  001242  000000             $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
(2)  001244  000000             $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
(2)  001246  000000             $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
(2)  001250                     $ETABLE:                        ;;APT ENVIRONMENT TABLE
(2)  001250  000                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
(2)  001251  000                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
(2)  001252  000000             $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
(2)  001254  000000             $USWR:  .WORD   AUSWR   ;;USER SWITCHES
(2)  001256  000000             $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
(2)                             ;*                      BITS 15-11=CPU TYPE
(2)                             ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(2)                             ;*                              11/70=06,PDQ=07,Q=10
(2)                             ;*                      BIT 10=REAL TIME CLOCK
(2)                             ;*                      BIT  9=FLOATING POINT PROCESSOR
(2)                             ;*                      BIT  8=MEMORY MANAGEMENT
(2)  001260  000                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
(2)  001261  000                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
(2)                             ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
(2)                             ;*                              900 NSEC CORE=001
(2)                             ;*                              300 NSEC BIPOLAR=002
(2)                             ;*                              500 NSEC MOS=003
(2)  001262  000000             $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
(2)                             ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
(2)  001264  000                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
(2)  001265  000                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
(2)  001266  000000             $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
(2)  001270  000                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
(2)  001271  000                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
(2)  001272  000000             $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
(2)  001274  000                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
(2)  001275  000                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
(2)  001276  000000             $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
(2)  001300  000000             $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)  001302  000000             $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2)  001304  000000             $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)  001306  000000             $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
(2)  001310  000000             $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
(2)  001312  000000             $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
(2)  001314  000000             $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
(2)  001316  000000             $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
(2)  001320  000000             $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
(2)  001322  000000             $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
(2)  001324  000000             $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
```

```
(2)  001326  000000                $DDW5:   .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
(2)  001330  000000                $DDW6:   .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
(2)  001332  000000                $DDW7:   .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
(2)  001334  000000                $DDW8:   .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
(2)  001336  000000                $DDW9:   .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
(2)  001340  000000                $DDW10:  .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
(2)  001342  000000                $DDW11:  .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
(2)  001344  000000                $DDW12:  .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
(2)  001346  000000                $DDW13:  .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
(2)  001350  000000                $DDW14:  .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
(2)  001352  000000                $DDW15:  .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
(2)
(2)
(2)  001354                        $ETEND:
(2)
```

```
(1)                                 .SBTTL  ERROR POINTER TABLE
(1)
(1)                                 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1)                                 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1)                                 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1)                                 ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(1)                                 ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1)                                 ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(1)                                 ;*      DH              ;;POINTS TO THE DATA HEADER
(1)                                 ;*      DT              ;;POINTS TO THE DATA
(1)                                 ;*      DF              ;;POINTS TO THE DATA FORMAT
(1)
(1)  001354                         $ERRTB:
169                                 ;ERROR PARAMETER TABLE
170
171                                 ;ERROR  1
172  001354  030000                         EM1             ;ACR BIT3-0 LOAD-READER ERROR
173  001356  032635                         DH1             ;PASS PC EXPCTD ACTUAL ACR
174  001360  033712                         DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
175  001362  033760                         DAF1
176
177                                 ;ERROR  2
178  001364  030033                         EM2             ;ACR BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)
179  001366  032701                         DH2             ;PASS PC ACR
180  001370  033726                         DT2             ;$PASS,$ERRPC,DISPLY
181  001372  033760                         DAF1
182
183                                 ;ERROR  3
184  001374  030473                         EM3             ;PGTE BIT11-8 LOAD-READ ERROR
185  001376  032752                         DH4             ;PASS PC EXPCTD ACUTAL PGTE
186  001400  033712                         DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
187  001402  033760                         DAF1
188
189                                 ;ERROR  4
190  001404  030530                         EM4             ;PGTE BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)
191  001406  032725                         DH3             ;PASS PC PGTE
192  001410  033726                         DT2             ;$PASS,$ERRPC,DISPLY
193  001412  033760                         DAF1
194
195                                 ;ERROR  5
196  001414  030611                         EM5             ;PGTE BIT3-0 LOAD-READ ERROR
197  001416  032752                         DH4             ;PASS PC EXPCTD ACUTAL PGTE
198  001420  033712                         DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
199  001422  033760                         DAF1
200
201                                 ;ERROR  6
202  001424  030645                         EM6             ;PGTE BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)
203  001426  032725                         DH3             ;PASS PC PGTE
204  001430  033726                         DT2             ;$PASS,$ERRPC,DISPLY
205  001432  033760                         DAF1
```

```
                                ;ERROR  7
001434  030112                          EM7             ;IE (PGTE BIT 6) CANNOT BE SET
001436  032725                          DH3             ;PASS PC PGTE
001440  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001442  033760                          DAF1

                                ;ERROR  10
001444  030147                          EM10            ;IE (PGTE BIT6) CANNOT BE WRITTEN TO 0
001446  032725                          DH3             ;PASS PC PGTE
001450  033712                          DT1             ;$PASS,$ERRPC,DISPLY
001452  033760                          DAF1

                                ;ERROR  11
001454  030215                          EM11            ;IE (PGTE BIT6) CANNOT BE CLEARED BY RESET (ACR CLR)
001456  032725                          DH3             ;PASS PC PGTE
001460  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001462  033760                          DAF1

                                ;ERROR  12
001464  030301                          EM12            ;PTP (PGTE BIT1) CANNOT BE SET
001466  032725                          DH3             ;PASS PC PGTE
001470  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001472  033760                          DAF1

                                ;ERROR  13
001474  030337                          EM13            ;PTP (PGTE BIT1) CANNOT BE WRITTEN TO 0
001476  032725                          DH3             ;PASS PC PGTE
001500  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001502  033760                          DAF1            ;

                                ;ERROR  14
001504  030406                          EM14            ;PTP (PGTE BIT1) CANNOT BE CLEARED BY RESET (ACR CLR)
001506  032725                          DH3             ;PASS PC PGTE
001510  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001512  033760                          DAF1

                                ;ERROR  15
001514  030725                          EM15            ;STTE BIT11-8 LOAD-READ ERROR
001516  033017                          DH5             ;PASS PC EXPCTD ACUTAL STTE
001520  033712                          DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001522  033760                          DAF1

                                ;ERROR  16
001524  030762                          EM16            ;STTE BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)
001526  033064                          DH6             ;PASS PC STTE
001530  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001532  033760                          DAF1

                                ;ERROR  17
001534  031043                          EM17            ;STTE BIT3-0 LOAD-READ ERROR
001536  033017                          DH5             ;PASS PC EXPCTD ACUTAL STTE
001540  033712                          DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001542  033760                          DAF1
```

```
                                ;ERROR  20
001544  031077                          EM20            ;STTE BIT3-0 WERE NOT CLEARED BY RESET (ACR-CLR)
001546  033064                          DH6             ;PASS PC STTE
001550  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001552  033760                          DAF1

                                ;ERROR  21
001554  031157                          EM21            ;LKE (STCS BIT2) CANNOT BE SET
001556  033111                          DH7             ;PASS PC STCS
001560  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001562  033760                          DAF1

                                ;ERROR  22
001564  031215                          EM22            ;LKE (STCS BIT2) CANNOT BE WRITTEN TO 0
001566  033111                          DH7             ;PASS PC STCS
001570  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001572  033760                          DAF1

                                ;ERROR  23
001574  031264                          EM23            ;LKE (STCS BIT2) CANNOT BE CLEARED BY RESET (ACR CLR)
001576  033111                          DH7             ;PASS PC STCS
001600  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001602  033760                          DAF1

                                ;ERROR  24
001604  031351                          EM24            ;STP (STCS BIT1) CANNOT BE SET
001606  033111                          DH7             ;PASS PC STCS
001610  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001612  033760                          DAF1

                                ;ERROR  25
001614  031407                          EM25            ;STP (STCS BIT1) CANNOT BE WRITTEN TO 0
001616  033111                          DH7             ;PASS PC STCS
001620  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001622  033760                          DAF1

                                ;ERROR  26
001624  031456                          EM26            ;STP (STCS BIT1) CANNOT BE CLEARED BY RESET (ACR CLR)
001626  033111                          DH7             ;PASS PC STCS
001630  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001632  033760                          DAF1

                                ;ERROR  27
001634  031543                          EM27            ;ENB (STCS BIT0) CANNOT BE SET
001636  033111                          DH7             ;PASS PC STCS
001640  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001642  033760                          DAF1

                                ;ERROR  30
001644  031601                          EM30            ;ENB (STCS BIT0) CANNOT BE WRITTEN TO 0
001646  033111                          DH7             ;PASS PC STCS
001650  033726                          DT2             ;$PASS,$ERRPC,DISPLY
001652  033760                          DAF1
```

```
                                        ;ERROR  31
001654  031650                                  EM31            ;ENB (STCS BIT0) CANNOT BE CLEARED BY RESET (ACR CLR)
001656  033111                                  DH7             ;PASS PC STCS
001660  033726                                  DT2             ;$PASS,$ERRPC,DISPLY
001662  033760                                  DAF1

                                        ;ERROR  32
001664  031735                                  EM32            ;STCS BIT15-8 LOAD-READ ERROR
001666  033136                                  DH8             ;PASS PC EXPCTD ACUTAL STCS
001670  033712                                  DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001672  033760                                  DAF1

                                        ;ERROR  33
001674  031772                                  EM33            ;STCS BIT15-8 WERE NOT CLEARED BY RESET (ACR CLR)
001676  033111                                  DH7             ;PASS PC STCS
001700  033726                                  DT2             ;$PASS,$ERRPC,DISPLY
001702  033760                                  DAF1

                                        ;ERROR  34
001704  032053                                  EM34            ;IMSK BIT11-8 LOAD-READ ERROR
001706  033203                                  DH9             ;PASS PC EXPCTD ACUTAL IMSK
001710  033712                                  DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001712  033760                                  DAF1

                                        ;ERROR  35
001714  032110                                  EM35            ;IMSK BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)
001716  033250                                  DH10            ;PASS PC IMSK
001720  033726                                  DT2             ;$PASS,$ERRPC,DISPLY
001722  033760                                  DAF1

                                        ;ERROR  36
001724  032171                                  EM36            ;IMSK BIT3-0 LOAD-READ ERROR
001726  033203                                  DH9             ;PASS PC EXPCTD ACUTAL IMSK
001730  033712                                  DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001732  033760                                  DAF1

                                        ;ERROR  37
001734  032225                                  EM37            ;IMSK BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)
001736  033250                                  DH10            ;PASS PC IMSK
001740  033726                                  DT2             ;$PASS,$ERRPC,DISPLY
001742  033760                                  DAF1

                                        ;ERROR  40
001744  032305                                  EM40            ;MTCE BIT11-8 LOAD-READ ERROR
001746  033275                                  DH11            ;PASS PC EXPCTD ACUTAL MTCE
001750  033712                                  DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001752  033760                                  DAF1

                                        ;ERROR  41
001754  032341                                  EM41            ;MTCE BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)
001756  033342                                  DH12            ;PASS PC MTCE
001760  033726                                  DT2             ;$PASS,$ERRPC,DISPLY
001762  033760                                  DAF1
```

```
                                ;ERROR  42
001764  032421                          EM42            ;MTCE BIT3-0 LOAD-READ ERROR
001766  033275                          DH11            ;PASS PC EXPCTD ACUTAL MTCE
001770  033712                          DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001772  033760                          DAF1

                                ;ERROR  43
001774  032454                          EM43            ;MTCE BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)
001776  033342                          DH12            ;PASS PC MTCE
002000  033726                          DT2             ;$PASS,$ERRPC,DISPLY
002002  033760                          DAF1
                                ;ERROR  44
002004  032533                          EM44            ;IIST ERROR  - REFER TO THE LISTING AT 'PC''
                                                        ;TO DETERMINE THE ERROR.
002006  033367                          DH13            ;PASS PC LINE # EXPCTD ACTUAL DISPLY $TMP0 $TMP1
002010  033736                          DT3             ;$PASS,$ERRPC,LINNBR,$GDDAT,$BDDAT,DISPLY,$TMP0,$TMP1
002012  033760                          DAF1

                                        .SBTTL  REGISTER DEFINITIONS

                                ;ACR BIT DEFINITIONS

        100000                  CLR=BIT15       ;MASTER CLEAR
        000000                  PGTEE=0
        000001                  PGCSE=1
        000002                  STTEE=2
        000003                  STCSE=3
        000004                  IMSKE=4
        000005                  PGFE=5
        000006                  STFE=6
        000007                  DCFE=7
        000010                  EXCE=10
        000015                  MTCE=15

                                ;PGTE BIT DEFINITION

        004000                  PB3=BIT11       ;PROGRAM BOOT TRANSMIT ENABLE 3
        002000                  PB2=BIT10       ;PROGRAM BOOT TRANSMIT ENABLE 2
        001000                  PB1=BIT9        ;PROGRAM BOOT TRANSMIT ENABLE 1
        000400                  PB0=BIT8        ;PROGRAM BOOT TRANSMIT ENABLE 0
        000010                  PI3=BIT3        ;PROGRAM INTERRUPT TRANSMIT ENABLE 3
        000004                  PI2=BIT2        ;PROGRAM INTERRUPT TRANSMIT ENABLE 2
        000002                  PI1=BIT1        ;PROGRAM INTERRUPT TRANSMIT ENABLE 1
        000001                  PI0=BIT0        ;PROGRAM INTERRUPT TRANSMIT ENABLE 0

                                ;PGCS BIT DEFINITIONS

        100000                  ERR=BIT15       ;ERROR
        040000                  GRJ=BIT14       ;GO REJECT
        020000                  PGMR=BIT13      ;PGTE MODIFICATION REFUSED
        010000                  STMR=BIT12      ;STTE MODIFICATION REFUSED
        004000                  PRDY=BIT11      ;PG READY
        000000                  SID0=0          ;SELF ID 0
        000400                  SID1=BIT8       ;SELF ID 1
        001000                  SID2=BIT9       ;SELF ID 2
        001400                  SID3=BIT9+BIT8  ;SELF ID 3
```

```
000010                  IP=BIT3         ;INTERRUPT PENDING IP=ERR+FLAGS
000004                  IE=BIT2         ;IIST INTERRUPT ENABLE
000002                  PTP=BIT1        ;PGTE PARITY BIT
000001                  GO=BIT0         ;GENERATE BOOT/INTERRUPT

                        ;STTE BIT DEFINITIONS

004000                  SB3=BIT11       ;SANITY TIMER BOOT TRANSMIT ENABLE 3
002000                  SB2=BIT10       ;SANITY TIMER BOOT TRANSMIT ENABLE 2
001000                  SB1=BIT9        ;SANITY TIMER BOOT TRANSMIT ENABLE 1
000400                  SB0=BIT8        ;SANITY TIMER BOOT TRANSMIT ENABLE 0
000010                  SI3=BIT3        ;SANITY TIMER INTERRUPT TRANSMIT ENABLE 3
000004                  SI2=BIT2        ;SANITY TIMER INTERRUPT TRANSMIT ENABLE 2
000002                  SI2=BIT1        ;SANITY TIMER INTERRUPT TRANSMIT ENABLE 1
000001                  SI0=BIT0        ;SANITY TIMER INTERRUPT TRANSMIT ENABLE 0

                        ;STCS BIT DEFINITIONS

000010                  TMO=BIT3        ;TIMER EXPIRED
000004                  LKE=BIT2        ;LOCKUP ENABLE
000002                  STP=BIT1        ;STTE PARITY BIT
000001                  ENB=BIT0        ;SANITY TIMER ENABLE

                        ;IMSK BIT DEFINITIONS

004000                  BM3=BIT11       ;BOOT INHIBIT 3
002000                  BM2=BIT10       ;BOOT INHIBIT 2
001000                  BM1=BIT9        ;BOOT INHIBIT 1
000400                  BM0=BIT8        ;BOOT INHIBIT 0
000010                  IM3=BIT3        ;INTERRUPT INHIBIT 3
000004                  IM2=BIT2        ;INTERRUPT INHIBIT 2
000002                  IM1=BIT1        ;INTERRUPT INHIBIT 1
000001                  IM0=BIT0        ;INTERRUPT INHIBIT 0

                        ;PGF BIT DEFINITIONS

004000                  PBF3=BIT11      ;PROGRAM GENERATED BOOT FLAG 3
002000                  PBF2=BIT10      ;PROGRAM GENERATED BOOT FLAG 2
001000                  PBF1=BIT9       ;PROGRAM GENERATED BOOT FLAG 1
000400                  PBF0=BIT8       ;PROGRAM GENERATED BOOT FLAG 0
000010                  PIF3=BIT3       ;PROGRAM GENERATED INTERRUPT FLAG 3
000004                  PIF2=BIT2       ;PROGRAM GENERATED INTERRUPT FLAG 2
000002                  PIF1=BIT1       ;PROGRAM GENERATED INTERRUPT FLAG 1
000001                  PIF0=BIT0       ;PROGRAM GENERATED INTERRUPT FLAG 0

                        ;STF BIT DEFINITIONS

000010                  SBF3=BIT3       ;SANITY TIMER GENERATED BOOT FLAG 3
000004                  SBF2=BIT2       ;SANITY TIMER GENERATED BOOT FLAG 2
000002                  SBF1=BIT1       ;SANITY TIMER GENERATED BOOT FLAG 1
000001                  SBF0=BIT0       ;SANITY TIMER GENERATED BOOT FLAG 0
000010                  SIF3=BIT3       ;SANITY TIMER GENERATED INTERRUPT FLAG 3
000004                  SIF2=BIT2       ;SANITY TIMER GENERATED INTERRUPT FLAG 2
000002                  SIF1=BIT1       ;SANITY TIMER GENERATED INTERRUPT FLAG 1
000001                  SIF0=BIT0       ;SANITY TIMER GENERATED INTERRUPT FLAG 0
```

```
                        ;DCF BIT DEFINITIONS
                        
004000                  BRK3=BIT11      ;DISCONNECT OR POWER LOSS STATE OF IIST 3
002000                  BRK2=BIT10      ;DISCONNECT OR POWER LOSS STATE OF IIST 2
001000                  BRK1=BIT9       ;DISCONNECT OR POWER LOSS STATE OF IIST 1
000400                  BRK0=BIT8       ;DISCONNECT OR POWER LOSS STATE OF IIST0
000010                  DCF3=BIT3       ;DCLO/DISCONNECT INTERRUPT FLAG 3
000004                  DCF2=BIT2       ;DCLO/DISCONNECT INTERRUPT FLAG 2
000002                  DCF1=BIT1       ;DCLO/DISCONNECT INTERRUPT FLAG 1
000001                  DCF0=BIT0       ;DCLO/DISCONNECT INTERRUPT FLAG 0
                        
                        ;EXC BIT DEFINITIONS
                        
004000                  UI3=BIT11       ;UNEXPECTED VALID INPUT FLAG 3
002000                  UI2=BIT10       ;UNEXPECTED VALID INPUT FLAG 2
001000                  UI1=BIT9        ;UNEXPECTED VALID INPUT FLAG 1
000400                  UI0=BIT8        ;UNEXPECTED VALID INPUT FLAG 0
000010                  RTE3=BIT3       ;UNEXPECTED INVALID INPUT FLAG 3
000004                  RTE2=BIT2       ;UNEXPECTED INVALID INPUT FLAG 2
000002                  RTE1=BIT1       ;UNEXPECTED INVALID INPUT FLAG 1
000001                  RTE0=BIT0       ;UNEXPECTED INVALID INPUT FLAG 0
                        
                        ;MTC BIT DEFINITIONS
                        
004000                  MTYP=BIT11      ;MNT TYPE BIT.0=PG; 1=ST
002000                  MFRM=BIT10      ;MNT FRAME BIT.0=NORMAL; 1=FRAMING ERROR
000000                  MID0=0          ;MNT ID 0
000400                  MID1=BIT8       ;MNT ID 1
001000                  MID2=BIT9       ;MNT ID 2
001400                  MID3=BIT9+BIT8  ;MNT ID 3
000010                  DSBT=BIT3       ;DISABLE BOOT
000004                  MENB=BIT2       ;ENABLE B11-8 OF MNT
000002                  MLEN=BIT1       ;ENABLE MAINT. LOOP (IDLE XMIT DRIVERS)
000001                  MDSD=BIT0       ;DISABLE XMIT DRIVER (CAUSE A BREAK)
000016                  TMT=DSBT+MENB+MLEN
                        
                        ;OTHER DEFINITIONS
                        
005726                  POP=5726
022626                  POPPOP=22626
000200                  CRLF=200
```

```
                        .SBTTL  TEST MACRO DEFINITIONS
```

```
611  002014                                  START:
(1)                                          .SBTTL  INITIALIZE THE COMMON TAGS
(1)                                          ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
(1)  002014  012706  001100                          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
(1)  002020  005026                                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
(1)  002022  022706  001140                          CMP     #SWR,R6 ;;DONE?
(1)  002026  001374                                  BNE     .-6             ;;LOOP BACK IF NO
(1)  002030  012706  001100                          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
(1)                                          ;;INITIALIZE A FEW VECTORS
(1)  002034  012737  024012  000020                  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
(1)  002042  012737  000340  000022                  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
(1)  002050  012737  024264  000030                  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
(1)  002056  012737  000340  000032                  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
(1)  002064  012737  027272  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(1)  002072  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
(1)  002100  012737  027354  000024                  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
(1)  002106  012737  000340  000026                  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
(1)  002114  013737  023724  023716                  MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
(1)  002122  005037  001214                          CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
(1)  002126  005037  001216                          CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
(1)  002132  112737  000001  001115                  MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
(1)  002140  012737  002140  001106                  MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
(1)  002146  012737  002146  001110                  MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
(2)                                          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
(2)                                          ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
(2)  002154  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
(2)  002160  012737  002214  000004                  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
(2)  002166  012737  177570  001140                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
(2)  002174  012737  177570  001142                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
(2)  002202  022777  177777  176730                  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
(2)  002210  001012                                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
(2)                                                                          ;;AND  THE HARDWARE SWR IS NOT = -1
(2)  002212  000403                                  BR      65$             ;;BRANCH IF NO TIMEOUT
(2)  002214  012716  002222          64$:           MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
(2)  002220  000002                                  RTI
(2)  002222  012737  000176  001140  65$:           MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
(2)  002230  012737  000174  001142                  MOV     #DISPREG,DISPLAY
(2)  002236  012637  000004          66$:           MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
(1)
(2)  002242  005037  001236                          CLR     $PASS           ;;CLEAR PASS COUNT
(2)  002246  132737  000200  001251                  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
(2)  002254  001403                                  BEQ     67$             ;;YES,USE NON-APT SWITCH
(2)  002256  012737  001252  001140                  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
(2)  002264                          67$:
612                                          .SBTTL  TYPE PROGRAM NAME
(1)                                          ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
(1)  002264  005227  177777                          INC     #-1             ;;FIRST TIME?
(1)  002270  001062                                  BNE     68$             ;;BRANCH IF NO
(1)  002272  022737  023756  000042                  CMP     #$ENDAD,@#42    ;;ACT-11?
(1)  002300  001456                                  BEQ     68$             ;;BRANCH IF YES
(1)  002302  104401  002350                          TYPE    ,69$            ;;TYPE ASCIZ STRING
(2)                                          .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
(2)  002306  005737  000042                          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
(2)  002312  001012                                  BNE     70$             ;;BRANCH IF YES
(2)  002314  123727  001250  000001                  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
(2)  002322  001406                                  BEQ     70$             ;;BRANCH IF YES
```

```
(2)  002324  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
(2)  002332  001005                          BNE     71$             ;;BRANCH IF NO
(2)  002334  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
(2)  002336  000403                          BR      71$
(2)  002340  112737  000001  001134  70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
(2)  002346                          71$:
(1)  002346  000433                          BR      68$             ;;GET OVER THE ASCIZ
(1)                                  ;;69$:   .ASCIZ  <CRLF>#DIP11-A (IIST) MONOPORT DIAGNOSTIC MAINDEC-11-CRIIA#<CRLF>
(1)  002436                          68$:
613  002436  012737  000006  000004          MOV     #6,4            ;SET TO TRAP
614  002444  012737  000004  000006          MOV     #4,6            ;ON AN NXM
615                                  ;       JSR     PC,$SIZE        ;SIZE MEMORY
616  002452  005737  023656                  TST     PARFLG  ;SEE IF PARAMETERS TO BE INPUT
617  002456  001402                          BEQ     1$              ;SKIP IF NOT
618  002460  004737  022336                  JSR     PC,GETPAR       ;GET PARAMETERS
619  002464  012737  000340  177776  1$:     MOV     #340,PSW        ;RISE PRIORITY
620  002472  005037  023662                  CLR     STCCNT          ;INITIALIZE S.T. COUNT VALUE.
621  002476  000137  002524                  JMP     BEGIN           ;GO START TESTING
622
623
624                                  ;COME HERE ON START AT 200
625  002502  005037  023656          NOPAR:  CLR     PARFLG          ;CLEAR PARAMETER FLAG
626  002506  000137  002014                  JMP     START           ;GO TO STARTUP
627
628                                  ;COME HERE ON START AT 204
629  002512  012737  000001  023656  SELPAR: MOV     #1,PARFLG       ;SET PARAMETER FLAG
630  002520  000137  002014                  JMP     START
631
632  002524                          BEGIN:
```

```
                                        .SBTTL
                                        .SBTTL
                                        .SBTTL  BASIC REGISTER TESTS
                                        .SBTTL
                                ;;******************************************************************
                                ;*TEST 1         ACR (BIT3-0) WRITE-READ TEST USING A COUNT PATTERN
                                ;;******************************************************************
(2)  002524  000004             TST1:   SCOPE
640  002526  012777  100000  021036        MOV     #CLR,@ACR       ;CLEAR THE WORLD
641  002534  012737  002550  001110        MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
642  002542  012737  000017  001124        MOV     #17,$GDDAT      ;INIT EXPECTED TO 17
643  002550  013777  001124  021014  1$:   MOV     $GDDAT,@ACR     ;LOAD ACR WITH $GDDAT
644  002556  017737  021010  023610        MOV     @ACR,DISPLY     ;READ ACR
645  002564  013737  023610  001126        MOV     DISPLY,$BDDAT   ;COPY ACR INTO $BDDAT
646  002572  042737  177760  001126        BIC     #-<17+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
647  002600  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
648  002606  001403                        BEQ     2$              ;SKIP IF OK
649  002610  104001                        ERROR+  1               ;ACR BIT3-0 LOAD-READ ERROR
650  002612  004737  022234                JSR     PC,DMPREG
651  002616  005237  001124        2$:     INC     $GDDAT          ;BUMP EXPECTED
652  002622  042737  177760  001124        BIC     #-<17+1>,$GDDAT ;MASK TO BITS UNDER TEST
653  002630  023727  001124  000017        CMP     $GDDAT,#17      ;DONE ALL?
654  002636  001344                        BNE     1$              ;TEST AGAIN IF NO
655  002640  012737  002646  001110        MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
656  002646  012777  000017  020716  3$:   MOV     #17,@ACR        ;SET ALL BITS
657  002654  012777  100000  020710        MOV     #CLR,@ACR       ;CLEAR THE WORLD
658  002662  017737  020704  023610        MOV     @ACR,DISPLY     ;READ ACR
659  002670  032737  000017  023610        BIT     #17,DISPLY      ;CHECK BIT3-0 IN ACR FOR 0
660  002676  001403                        BEQ     TST2            ;;SKIP IF RESET CLEARED BIT3-0 IN ACR
661  002700  104002                        ERROR+  2               ;BIT3-0 IN ACR WERE NOT CLEARED BY RESET
662  002702  004737  022234                JSR     PC,DMPREG
```

```
667
(4)                                     ;;*******************************************************************
(4)                                     ;*TEST 2        PGTE (BIT11-8) WRITE-READ TEST USING PGTE COUNT PATTERN
(4)                                     ;;*******************************************************************
(3)  002706  000004                     TST2:   SCOPE
(1)  002710  012777  100000  020654             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  002716  012737  002732  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  002724  012737  007400  001124             MOV     #7400,$GDDAT    ;INIT EXPECTED TO 7400
(1)  002732  012777  000000  020632  1$:        MOV     #PGTEE,@ACR     ;SET ADR TO PGTE
(1)  002740  013777  001124  020630             MOV     $GDDAT,@PGTE    ;LOAD PGTE WITH $GDDAT
(1)  002746  012777  000000  020616             MOV     #PGTEE,@ACR     ;RESET ADR TO PGTE
(1)  002754  017737  020616  023610             MOV     @PGTE,DISPLY    ;READ PGTE
(1)  002762  013737  023610  001126             MOV     DISPLY,$BDDAT   ;COPY PGTE INTO $BDDAT
(1)  002770  042737  170377  001126             BIC     #-<7400+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  002776  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  003004  001403                             BEQ     2$              ;SKIP IF OK
(1)  003006  104003                             ERROR+  N               ;PGTE BIT11-8 LOAD-READ ERROR
(1)  003010  004737  022234                     JSR     PC,DMPREG
(1)  003014  062737  000400  001124  2$:        ADD     #BIT8,$GDDAT    ;BUMP EXPECTED
(1)  003022  042737  170377  001124             BIC     #-<7400+1>,$GDDAT       ;MASK TO BITS UNDER TEST
(1)  003030  023727  001124  007400             CMP     $GDDAT,#7400    ;DONE ALL?
(1)  003036  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  003040  012737  003046  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  003046  012777  000000  020516  3$:        MOV     #PGTEE,@ACR     ;RESET ADR TO PGTE
(1)  003054  012777  007400  020514             MOV     #7400,@PGTE             ;SET ALL BITS
(1)  003062  012777  100000  020502             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  003070  012777  000000  020474             MOV     #PGTEE,@ACR     ;SET ADDR TO PGTE
(1)  003076  017737  020474  023610             MOV     @PGTE,DISPLY    ;READ PGTE
(1)  003104  032737  007400  023610             BIT     #7400,DISPLY    ;CHECK BIT11-8 IN PGTE FOR 0
(1)  003112  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT11-8 IN PGTE
(1)  003114  104004                             ERROR+  N               ;BIT11-8 IN PGTE WERE NOT CLEARED BY RESET
(1)  003116  004737  022234                     JSR     PC,DMPREG
(1)  003122                             4$:
```

```
669
(4)                                     ;;*******************************************************************
(4)                                     ;*TEST 3        PGTE (BIT3-0) WRITE-READ TEST USING PGTE COUNT PATTERN
(4)                                     ;;*******************************************************************
(3)  003122  000004                     TST3:   SCOPE
(1)  003124  012777  100000  020440             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  003132  012737  003146  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  003140  012737  000017  001124             MOV     #17,$GDDAT      ;INIT EXPECTED TO 17
(1)  003146  012777  000000  020416  1$:        MOV     #PGTEE,@ACR     ;SET ADR TO PGTE
(1)  003154  013777  001124  020414             MOV     $GDDAT,@PGTE    ;LOAD PGTE WITH $GDDAT
(1)  003162  012777  000000  020402             MOV     #PGTEE,@ACR     ;RESET ADR TO PGTE
(1)  003170  017737  020402  023610             MOV     @PGTE,DISPLY    ;READ PGTE
(1)  003176  013737  023610  001126             MOV     DISPLY,$BDDAT   ;COPY PGTE INTO $BDDAT
(1)  003204  042737  177760  001126             BIC     #-<17+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  003212  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  003220  001403                             BEQ     2$              ;SKIP IF OK
(1)  003222  104005                             ERROR+  N               ;PGTE BIT3-0 LOAD-READ ERROR
(1)  003224  004737  022234                     JSR     PC,DMPREG
(1)  003230  062737  000001  001124  2$:        ADD     #BIT0,$GDDAT    ;BUMP EXPECTED
(1)  003236  042737  177760  001124             BIC     #-<17+1>,$GDDAT ;MASK TO BITS UNDER TEST
(1)  003244  023727  001124  000017             CMP     $GDDAT,#17      ;DONE ALL?
(1)  003252  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  003254  012737  003262  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  003262  012777  000000  020302  3$:        MOV     #PGTEE,@ACR     ;RESET ADR TO PGTE
(1)  003270  012777  000017  020300             MOV     #17,@PGTE               ;SET ALL BITS
(1)  003276  012777  100000  020266             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  003304  012777  000000  020260             MOV     #PGTEE,@ACR     ;SET ADDR TO PGTE
(1)  003312  017737  020260  023610             MOV     @PGTE,DISPLY    ;READ PGTE
(1)  003320  032737  000017  023610             BIT     #17,DISPLY      ;CHECK BIT3-0 IN PGTE FOR 0
(1)  003326  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT3-0 IN PGTE
(1)  003330  104006                             ERROR+  N               ;BIT3-0 IN PGTE WERE NOT CLEARED BY RESET
(1)  003332  004737  022234                     JSR     PC,DMPREG
(1)  003336                             4$:
```

```
671
(4)
(4)
(3)  003336  000004
(1)  003340  012777  100000  020224  TST4:   SCOPE
(1)  003346  012777  000001  020216          MOV     #CLR,@ACR       ;CLEAR THE SLATE
(1)  003354  012777  000004  020214          MOV     #PGCSE,@ACR     ;SET ADR TO PGCS
(1)  003362  012777  000001  020202          MOV     #IE,@PGCS               ;SET IE IN PGCS
(1)  003370  017737  020202  023610          MOV     #PGCSE,@ACR     ;RESET ADR TO PGCS
(1)  003376  032737  000004  023610          MOV     @PGCS,DISPLY    ;READ PGCS
(3)  003404  001003                          BIT     #IE,DISPLY      ;CHECK FOR IE=1
(1)  003406  104007                          BNE     TST5            ;;SKIP IF IE IS SET IN PGCS
(1)  003410  004737  022234                  ERROR+  N               ;IE (PGCS BIT2) CANNOT BE SET
(1)                                          JSR     PC,DMPREG
(1)
(5)
(4)
(4)
(3)  003414  000004  TST5:   SCOPE
(1)  003416  012777  000001  020146          MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  003424  012777  000004  020144          MOV     #IE,@PGCS               ;SET IE IN PGCS
(1)  003432  012777  000001  020132          MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  003440  042777  000004  020130          BIC     #IE,@PGCS               ;CLEAR IE IN PGCS
(1)  003446  012777  000001  020116          MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  003454  017737  020116  023610          MOV     @PGCS,DISPLY    ;READ PGCS
(1)  003462  032737  000004  023610          BIT     #IE,DISPLY      ;CHECK FOR IE=0
(3)  003470  001403                          BEQ     TST6            ;;SKIP IF IE IS CLEAR IN PGCS
(1)  003472  104010                          ERROR+  N               ;IE (PGCS BIT2) CANNOT BE CLEARED
(1)  003474  004737  022234                  JSR     PC,DMPREG
(1)                                                                  ;BY WRITING IT TO 0
(1)
(5)
(4)
(4)
(3)  003500  000004  TST6:   SCOPE
(1)  003502  012777  000001  020062          MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  003510  012777  000004  020060          MOV     #IE,@PGCS               ;SET IE IN PGCS
(1)  003516  012777  100000  020046          MOV     #CLR,@ACR       ;CLEAR THE INTERFACE
(1)  003524  012777  000001  020040          MOV     #PGCSE,@ACR     ;SET ADR TO PGCS
(1)  003532  017737  020040  023610          MOV     @PGCS,DISPLY    ;READ PGCS
(1)  003540  032737  000004  023610          BIT     #IE,DISPLY      ;CHECK FOR IE=0
(3)  003546  001403                          BEQ     TST7            ;;SKIP IS IE IS CLEARED BY RESET IN PGCS
(1)  003550  104011                          ERROR+  N               ;IE (PGCS BIT2) CANNOT BE CLEARED BY RESET
(1)  003552  004737  022234                  JSR     PC,DMPREG
```

```
673
 (4)                                    ;;*****************************************************************
 (4)                                    ;*TEST 7        TEST THAT PTP (PGCS BIT1) CAN BE SET
 (4)                                    ;;*****************************************************************
 (3)  003556  000004                    TST7:   SCOPE
 (1)  003560  012777  100000  020004            MOV     #CLR,@ACR       ;CLEAR THE SLATE
 (1)  003566  012777  000001  017776            MOV     #PGCSE,@ACR     ;SET ADR TO PGCS
 (1)  003574  012777  000002  017774            MOV     #PTP,@PGCS              ;SET PTP IN PGCS
 (1)  003602  012777  000001  017762            MOV     #PGCSE,@ACR     ;RESET ADR TO PGCS
 (1)  003610  017737  017762  023610            MOV     @PGCS,DISPLY    ;READ PGCS
 (1)  003616  032737  000002  023610            BIT     #PTP,DISPLY     ;CHECK FOR PTP=1
 (3)  003624  001003                            BNE     TST10           ;;SKIP IF PTP IS SET IN PGCS
 (1)  003626  104012                            ERROR+  N               ;PTP (PGCS BIT1) CANNOT BE SET
 (1)  003630  004737  022234                    JSR     PC,DMPREG
 (1)
 (5)                                    ;;*****************************************************************
 (4)                                    ;*TEST 10       TEST THAT PTP (PGCS BIT1) CAN BE CLEARED
 (4)                                    ;;*****************************************************************
 (3)  003634  000004                    TST10:  SCOPE
 (1)  003636  012777  000001  017726            MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
 (1)  003644  012777  000002  017724            MOV     #PTP,@PGCS              ;SET PTP IN PGCS
 (1)  003652  012777  000001  017712            MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
 (1)  003660  042777  000002  017710            BIC     #PTP,@PGCS              ;CLEAR PTP IN PGCS
 (1)  003666  012777  000001  017676            MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
 (1)  003674  017737  017676  023610            MOV     @PGCS,DISPLY    ;READ PGCS
 (1)  003702  032737  000002  023610            BIT     #PTP,DISPLY     ;CHECK FOR PTP=0
 (3)  003710  001403                            BEQ     TST11           ;;SKIP IF PTP IS CLEAR IN PGCS
 (1)  003712  104013                            ERROR+  N               ;PTP (PGCS BIT1) CANNOT BE CLEARED
 (1)  003714  004737  022234                    JSR     PC,DMPREG
 (1)                                                                    ;BY WRITING IT TO 0
 (1)
 (5)                                    ;;*****************************************************************
 (4)                                    ;*TEST 11       TEST THAT PTP (PGCS BIT1) CAN BE CLEARED BY RESET
 (4)                                    ;;*****************************************************************
 (3)  003720  000004                    TST11:  SCOPE
 (1)  003722  012777  000001  017642            MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
 (1)  003730  012777  000002  017640            MOV     #PTP,@PGCS              ;SET PTP IN PGCS
 (1)  003736  012777  100000  017626            MOV     #CLR,@ACR       ;CLEAR THE INTERFACE
 (1)  003744  012777  000001  017620            MOV     #PGCSE,@ACR     ;SET ADR TO PGCS
 (1)  003752  017737  017620  023610            MOV     @PGCS,DISPLY    ;READ PGCS
 (1)  003760  032737  000002  023610            BIT     #PTP,DISPLY     ;CHECK FOR PTP=0
 (3)  003766  001403                            BEQ     TST12           ;;SKIP IS PTP IS CLEARED BY RESET IN PGCS
 (1)  003770  104014                            ERROR+  N               ;PTP (PGCS BIT1) CANNOT BE CLEARED BY RESET
 (1)  003772  004737  022234                    JSR     PC,DMPREG
```

```
675
(4)
(4)
(3)  003776  000004
(1)  004000  012777  100000  017564  TST12:  SCOPE
(1)  004006  012737  004022  001110          MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  004014  012737  007400  001124          MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  004022  012777  000002  017542  1$:     MOV     #7400,$GDDAT    ;INIT EXPECTED TO 7400
(1)  004030  013777  001124  017540          MOV     #STTEE,@ACR     ;SET ADR TO STTE
(1)  004036  012777  000002  017526          MOV     $GDDAT,@STTE    ;LOAD STTE WITH $GDDAT
(1)  004044  017737  017526  023610          MOV     #STTEE,@ACR     ;RESET ADR TO STTE
(1)  004052  013737  023610  001126          MOV     @STTE,DISPLY    ;READ STTE
(1)  004060  042737  170377  001126          MOV     DISPLY,$BDDAT   ;COPY STTE INTO $BDDAT
(1)  004066  023737  001124  001126          BIC     #-<7400+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  004074  001403                          CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  004076  104015                          BEQ     2$              ;SKIP IF OK
(1)  004100  004737  022234                  ERROR+  N               ;STTE BIT11-8 LOAD-READ ERROR
(1)  004104  062737  000400  001124  2$:     JSR     PC,DMPREG
(1)  004112  042737  170377  001124          ADD     #BIT8,$GDDAT    ;BUMP EXPECTED
(1)  004120  023727  001124  007400          BIC     #-<7400+1>,$GDDAT       ;MASK TO BITS UNDER TEST
(1)  004126  001335                          CMP     $GDDAT,#7400    ;DONE ALL?
(1)  004130  012737  004136  001110          BNE     1$              ;TEST AGAIN IF NO
(1)  004136  012777  000002  017426  3$:     MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  004144  012777  007400  017424          MOV     #STTEE,@ACR     ;RESET ADR TO STTE
(1)  004152  012777  100000  017412          MOV     #7400,@STTE             ;SET ALL BITS
(1)  004160  012777  000002  017404          MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  004166  017737  017404  023610          MOV     #STTEE,@ACR     ;SET ADDR TO STTE
(1)  004174  032737  007400  023610          MOV     @STTE,DISPLY    ;READ STTE
(1)  004202  001403                          BIT     #7400,DISPLY    ;CHECK BIT11-8 IN STTE FOR 0
(1)  004204  104016                          BEQ     4$              ;SKIP IF RESET CLEARED BIT11-8 IN STTE
(1)  004206  004737  022234                  ERROR+  N               ;BIT11-8 IN STTE WERE NOT CLEARED BY RESET
(1)  004212                                  JSR     PC,DMPREG
                                     4$:
```

```
677
(4)
(4)
(3)  004212  000004                          TST13:  SCOPE
(1)  004214  012777  100000  017350                  MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  004222  012737  004236  001110                  MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  004230  012737  000017  001124                  MOV     #17,$GDDAT      ;INIT EXPECTED TO 17
(1)  004236  012777  000002  017326          1$:     MOV     #STTEE,@ACR     ;SET ADR TO STTE
(1)  004244  013777  001124  017324                  MOV     $GDDAT,@STTE    ;LOAD STTE WITH $GDDAT
(1)  004252  012777  000002  017312                  MOV     #STTEE,@ACR     ;RESET ADR TO STTE
(1)  004260  017737  017312  023610                  MOV     @STTE,DISPLY    ;READ STTE
(1)  004266  013737  023610  001126                  MOV     DISPLY,$BDDAT   ;COPY STTE INTO $BDDAT.
(1)  004274  042737  177760  001126                  BIC     #-<17+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  004302  023737  001124  001126                  CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  004310  001403                                  BEQ     2$              ;SKIP IF OK
(1)  004312  104017                                  ERROR+  N               ;STTE BIT3-0 LOAD-READ ERROR
(1)  004314  004737  022234                          JSR     PC,DMPREG
(1)  004320  062737  000001  001124          2$:     ADD     #BIT0,$GDDAT    ;BUMP EXPECTED
(1)  004326  042737  177760  001124                  BIC     #-<17+1>,$GDDAT ;MASK TO BITS UNDER TEST
(1)  004334  023727  001124  000017                  CMP     $GDDAT,#17      ;DONE ALL?
(1)  004342  001335                                  BNE     1$              ;TEST AGAIN IF NO
(1)  004344  012737  004352  001110                  MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  004352  012777  000002  017212          3$:     MOV     #STTEE,@ACR     ;RESET ADR TO STTE
(1)  004360  012777  000017  017210                  MOV     #17,@STTE               ;SET ALL BITS
(1)  004366  012777  100000  017176                  MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  004374  012777  000002  017170                  MOV     #STTEE,@ACR     ;SET ADDR TO STTE
(1)  004402  017737  017170  023610                  MOV     @STTE,DISPLY    ;READ STTE
(1)  004410  032737  000017  023610                  BIT     #17,DISPLY      ;CHECK BIT3-0 IN STTE FOR 0
(1)  004416  001403                                  BEQ     4$              ;SKIP IF RESET CLEARED BIT3-0 IN STTE
(1)  004420  104020                                  ERROR+  N               ;BIT3-0 IN STTE WERE NOT CLEARED BY RESET
(1)  004422  004737  022234                          JSR     PC,DMPREG
(1)  004426                                  4$:
```

```
679
(4)
(4)
(3)  004426  000004                    ;;*****************************************************************
(1)  004430  012777  100000  017134    ;*TEST 14       TEST THAT LKE (STCS BIT2) CAN BE SET
(1)  004436  012777  000003  017126    ;;*****************************************************************
(1)  004444  012777  000004  017124    TST14:  SCOPE
(1)  004452  012777  000003  017112            MOV     #CLR,@ACR       ;CLEAR THE SLATE
(1)  004460  017737  017112  023610            MOV     #STCSE,@ACR     ;SET ADR TO STCS
(1)  004466  032737  000004  023610            MOV     #LKE,@STCS              ;SET LKE IN STCS
(3)  004474  001003                            MOV     #STCSE,@ACR     ;RESET ADR TO STCS
(1)  004476  104021                            MOV     @STCS,DISPLY    ;READ STCS
(1)  004500  004737  022234                    BIT     #LKE,DISPLY     ;CHECK FOR LKE=1
(1)                                            BNE     TST15           ;;SKIP IF LKE IS SET IN STCS
(5)                                            ERROR+  N               ;LKE (STCS BIT2) CANNOT BE SET
(4)                                            JSR     PC,DMPREG
(4)                                    ;;*****************************************************************
(3)  004504  000004                    ;*TEST 15       TEST THAT LKE (STCS BIT2) CAN BE CLEARED
(1)  004506  012777  000003  017056    ;;*****************************************************************
(1)  004514  012777  000004  017054    TST15:  SCOPE
(1)  004522  012777  000003  017042            MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  004530  042777  000004  017040            MOV     #LKE,@STCS              ;SET LKE IN STCS
(1)  004536  012777  000003  017026            MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  004544  017737  017026  023610            BIC     #LKE,@STCS              ;CLEAR LKE IN STCS
(1)  004552  032737  000004  023610            MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(3)  004560  001403                            MOV     @STCS,DISPLY    ;READ STCS
(1)  004562  104022                            BIT     #LKE,DISPLY     ;CHECK FOR LKE=0
(1)  004564  004737  022234                    BEQ     TST16           ;;SKIP IF LKE IS CLEAR IN STCS
(1)                                            ERROR+  N               ;LKE (STCS BIT2) CANNOT BE CLEARED
(1)                                            JSR     PC,DMPREG
(5)                                                                    ;BY WRITING IT TO 0
(4)                                    ;;*****************************************************************
(4)                                    ;*TEST 16       TEST THAT LKE (STCS BIT2) CAN BE CLEARED BY RESET
(3)  004570  000004                    ;;*****************************************************************
(1)  004572  012777  000003  016772    TST16:  SCOPE
(1)  004600  012777  000004  016770            MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  004606  012777  100000  016756            MOV     #LKE,@STCS              ;SET LKE IN STCS
(1)  004614  012777  000003  016750            MOV     #CLR,@ACR       ;CLEAR THE INTERFACE
(1)  004622  017737  016750  023610            MOV     #STCSE,@ACR     ;SET ADR TO STCS
(1)  004630  032737  000004  023610            MOV     @STCS,DISPLY    ;READ STCS
(3)  004636  001403                            BIT     #LKE,DISPLY     ;CHECK FOR LKE=0
(1)  004640  104023                            BEQ     TST17           ;;SKIP IS LKE IS CLEARED BY RESET IN STCS
(1)  004642  004737  022234                    ERROR+  N               ;LKE (STCS BIT2) CANNOT BE CLEARED BY RESET
                                               JSR     PC,DMPREG
```

```
                                ;;*****************************************************************
                                ;*TEST 17        TEST THAT STP (STCS BIT1) CAN BE SET
                                ;;*****************************************************************
004646  000004                  TST17:  SCOPE
004650  012777  100000  016714          MOV     #CLR,@ACR       ;CLEAR THE SLATE
004656  012777  000003  016706          MOV     #STCSE,@ACR     ;SET ADR TO STCS
004664  012777  000002  016704          MOV     #STP,@STCS              ;SET STP IN STCS
004672  012777  000003  016672          MOV     #STCSE,@ACR     ;RESET ADR TO STCS
004700  017737  016672  023610          MOV     @STCS,DISPLY    ;READ STCS
004706  032737  000002  023610          BIT     #STP,DISPLY     ;CHECK FOR STP=1
004714  001003                          BNE     TST20           ;;SKIP IF STP IS SET IN STCS
004716  104024                          ERROR+  N               ;STP (STCS BIT1) CANNOT BE SET
004720  004737  022234                  JSR     PC,DMPREG

                                ;;*****************************************************************
                                ;*TEST 20        TEST THAT STP (STCS BIT1) CAN BE CLEARED
                                ;;*****************************************************************
004724  000004                  TST20:  SCOPE
004726  012777  000003  016636          MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
004734  012777  000002  016634          MOV     #STP,@STCS              ;SET STP IN STCS
004742  012777  000003  016622          MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
004750  042777  000002  016620          BIC     #STP,@STCS              ;CLEAR STP IN STCS
004756  012777  000003  016606          MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
004764  017737  016606  023610          MOV     @STCS,DISPLY    ;READ STCS
004772  032737  000002  023610          BIT     #STP,DISPLY     ;CHECK FOR STP=0
005000  001403                          BEQ     TST21           ;;SKIP IF STP IS CLEAR IN STCS
005002  104025                          ERROR+  N               ;STP (STCS BIT1) CANNOT BE CLEARED
005004  004737  022234                  JSR     PC,DMPREG
                                                                ;BY WRITING IT TO 0

                                ;;*****************************************************************
                                ;*TEST 21        TEST THAT STP (STCS BIT1) CAN BE CLEARED BY RESET
                                ;;*****************************************************************
005010  000004                  TST21:  SCOPE
005012  012777  000003  016552          MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
005020  012777  000002  016550          MOV     #STP,@STCS              ;SET STP IN STCS
005026  012777  100000  016536          MOV     #CLR,@ACR       ;CLEAR THE INTERFACE
005034  012777  000003  016530          MOV     #STCSE,@ACR     ;SET ADR TO STCS
005042  017737  016530  023610          MOV     @STCS,DISPLY    ;READ STCS
005050  032737  000002  023610          BIT     #STP,DISPLY     ;CHECK FOR STP=0
005056  001403                          BEQ     TST22           ;;SKIP IS STP IS CLEARED BY RESET IN STCS
005060  104026                          ERROR+  N               ;STP (STCS BIT1) CANNOT BE CLEARED BY RESET
005062  004737  022234                  JSR     PC,DMPREG
```

```
                                ;;*****************************************************************
                                ;*TEST 22        TEST THAT ENB (STCS BIT0) CAN BE SET
                                ;;*****************************************************************
005066  000004                  TST22:  SCOPE
005070  012777  100000  016474          MOV     #CLR,@ACR       ;CLEAR THE SLATE
005076  012777  000003  016466          MOV     #STCSE,@ACR     ;SET ADR TO STCS
005104  012777  000001  016464          MOV     #ENB,@STCS              ;SET ENB IN STCS
005112  012777  000003  016452          MOV     #STCSE,@ACR     ;RESET ADR TO STCS
005120  017737  016452  023610          MOV     @STCS,DISPLY    ;READ STCS
005126  032737  000001  023610          BIT     #ENB,DISPLY     ;CHECK FOR ENB=1
005134  001003                          BNE     TST23           ;;SKIP IF ENB IS SET IN STCS
005136  104027                          ERROR+  N               ;ENB (STCS BIT0) CANNOT BE SET
005140  004737  022234                  JSR     PC,DMPREG

                                ;;*****************************************************************
                                ;*TEST 23        TEST THAT ENB (STCS BIT0) CAN BE CLEARED
                                ;;*****************************************************************
005144  000004                  TST23:  SCOPE
005146  012777  000003  016416          MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
005154  012777  000001  016414          MOV     #ENB,@STCS              ;SET ENB IN STCS
005162  012777  000003  016402          MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
005170  042777  000001  016400          BIC     #ENB,@STCS              ;CLEAR ENB IN STCS
005176  012777  000003  016366          MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
005204  017737  016366  023610          MOV     @STCS,DISPLY    ;READ STCS
005212  032737  000001  023610          BIT     #ENB,DISPLY     ;CHECK FOR ENB=0
005220  001403                          BEQ     TST24           ;;SKIP IF ENB IS CLEAR IN STCS
005222  104030                          ERROR+  N               ;ENB (STCS BIT0) CANNOT BE CLEARED
005224  004737  022234                  JSR     PC,DMPREG
                                                                ;BY WRITING IT TO 0

                                ;;*****************************************************************
                                ;*TEST 24        TEST THAT ENB (STCS BIT0) CAN BE CLEARED BY RESET
                                ;;*****************************************************************
005230  000004                  TST24:  SCOPE
005232  012777  000003  016332          MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
005240  012777  000001  016330          MOV     #ENB,@STCS              ;SET ENB IN STCS
005246  012777  100000  016316          MOV     #CLR,@ACR       ;CLEAR THE INTERFACE
005254  012777  000003  016310          MOV     #STCSE,@ACR     ;SET ADR TO STCS
005262  017737  016310  023610          MOV     @STCS,DISPLY    ;READ STCS
005270  032737  000001  023610          BIT     #ENB,DISPLY     ;CHECK FOR ENB=0
005276  001403                          BEQ     TST25           ;;SKIP IS ENB IS CLEARED BY RESET IN STCS
005300  104031                          ERROR+  N               ;ENB (STCS BIT0) CANNOT BE CLEARED BY RESET
005302  004737  022234                  JSR     PC,DMPREG
```

```
685
(4)                                     ;;*********************************************************************
(4)                                     ;*TEST 25      STCS (BIT15-8) WRITE-READ TEST USING STCS COUNT PATTERN
(4)                                     ;;*********************************************************************
(3)  005306  000004                     TST25:  SCOPE
(1)  005310  012777  100000  016254             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  005316  012737  005332  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  005324  012737  177400  001124             MOV     #177400,$GDDAT  ;INIT EXPECTED TO 177400
(1)  005332  012777  000003  016232  1$:        MOV     #STCSE,@ACR     ;SET ADR TO STCS
(1)  005340  013777  001124  016230             MOV     $GDDAT,@STCS    ;LOAD STCS WITH $GDDAT
(1)  005346  012777  000003  016216             MOV     #STCSE,@ACR     ;RESET ADR TO STCS
(1)  005354  017737  016216  023610             MOV     @STCS,DISPLY    ;READ STCS
(1)  005362  013737  023610  001126             MOV     DISPLY,$BDDAT   ;COPY STCS INTO $BDDAT
(1)  005370  042737  000377  001126             BIC     #-<177400+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  005376  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  005404  001403                             BEQ     2$              ;SKIP IF OK
(1)  005406  104032                             ERROR+  N               ;STCS BIT15-8 LOAD-READ ERROR
(1)  005410  004737  022234                     JSR     PC,DMPREG
(1)  005414  062737  000400  001124  2$:        ADD     #BIT8,$GDDAT    ;BUMP EXPECTED
(1)  005422  042737  000377  001124             BIC     #-<177400+1>,$GDDAT     ;MASK TO BITS UNDER TEST
(1)  005430  023727  001124  177400             CMP     $GDDAT,#177400  ;DONE ALL?
(1)  005436  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  005440  012737  005446  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  005446  012777  000003  016116  3$:        MOV     #STCSE,@ACR     ;RESET ADR TO STCS
(1)  005454  012777  177400  016114             MOV     #177400,@STCS           ;SET ALL BITS
(1)  005462  012777  100000  016102             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  005470  012777  000003  016074             MOV     #STCSE,@ACR     ;SET ADDR TO STCS
(1)  005476  017737  016074  023610             MOV     @STCS,DISPLY    ;READ STCS
(1)  005504  032737  177400  023610             BIT     #177400,DISPLY  ;CHECK BIT15-8 IN STCS FOR 0
(1)  005512  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT15-8 IN STCS
(1)  005514  104033                             ERROR+  N               ;BIT15-8 IN STCS WERE NOT CLEARED BY RESET
(1)  005516  004737  022234                     JSR     PC,DMPREG
(1)  005522                             4$:
```

```
687                                     ;;*********************************************************************
(4)                                     ;*TEST 26       IMSK (BIT11-8) WRITE-READ TEST USING IMSK COUNT PATTERN
(4)                                     ;;*********************************************************************
(3)  005522  000004                     TST26:  SCOPE
(1)  005524  012777  100000  016040             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  005532  012737  005546  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  005540  012737  007400  001124             MOV     #7400,$GDDAT    ;INIT EXPECTED TO 7400
(1)  005546  012777  000004  016016  1$:        MOV     #IMSKE,@ACR     ;SET ADR TO IMSK
(1)  005554  013777  001124  016014             MOV     $GDDAT,@IMSK    ;LOAD IMSK WITH $GDDAT
(1)  005562  012777  000004  016002             MOV     #IMSKE,@ACR     ;RESET ADR TO IMSK
(1)  005570  017737  016002  023610             MOV     @IMSK,DISPLY    ;READ IMSK
(1)  005576  013737  023610  001126             MOV     DISPLY,$BDDAT   ;COPY IMSK INTO $BDDAT
(1)  005604  042737  170377  001126             BIC     #-<7400+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  005612  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  005620  001403                             BEQ     2$              ;SKIP IF OK
(1)  005622  104034                             ERROR+  N               ;IMSK BIT11-8 LOAD-READ ERROR
(1)  005624  004737  022234                     JSR     PC,DMPREG
(1)  005630  062737  000400  001124  2$:        ADD     #BIT8,$GDDAT    ;BUMP EXPECTED
(1)  005636  042737  170377  001124             BIC     #-<7400+1>,$GDDAT       ;MASK TO BITS UNDER TEST
(1)  005644  023727  001124  007400             CMP     $GDDAT,#7400    ;DONE ALL?
(1)  005652  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  005654  012737  005662  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  005662  012777  000004  015702  3$:        MOV     #IMSKE,@ACR     ;RESET ADR TO IMSK
(1)  005670  012777  007400  015700             MOV     #7400,@IMSK             ;SET ALL BITS
(1)  005676  012777  100000  015666             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  005704  012777  000004  015660             MOV     #IMSKE,@ACR     ;SET ADDR TO IMSK
(1)  005712  017737  015660  023610             MOV     @IMSK,DISPLY    ;READ IMSK
(1)  005720  032737  007400  023610             BIT     #7400,DISPLY    ;CHECK BIT11-8 IN IMSK FOR 0
(1)  005726  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT11-8 IN IMSK
(1)  005730  104035                             ERROR+  N               ;BIT11-8 IN IMSK WERE NOT CLEARED BY RESET
(1)  005732  004737  022234                     JSR     PC,DMPREG
(1)  005736                             4$:
```

```
689
(4)                                     ;;*********************************************************************
(4)                                     ;*TEST 27       IMSK (BIT3-0) WRITE-READ TEST USING IMSK COUNT PATTERN
(4)                                     ;;*********************************************************************
(3)  005736  000004                     TST27:  SCOPE
(1)  005740  012777  100000  015624             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  005746  012737  005762  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  005754  012737  000017  001124             MOV     #17,$GDDAT      ;INIT EXPECTED TO 17
(1)  005762  012777  000004  015602  1$:        MOV     #IMSKE,@ACR     ;SET ADR TO IMSK
(1)  005770  013777  001124  015600             MOV     $GDDAT,@IMSK    ;LOAD IMSK WITH $GDDAT
(1)  005776  012777  000004  015566             MOV     #IMSKE,@ACR     ;RESET ADR TO IMSK
(1)  006004  017737  015566  023610             MOV     @IMSK,DISPLY    ;READ IMSK
(1)  006012  013737  023610  001126             MOV     DISPLY,$BDDAT   ;COPY IMSK INTO $BDDAT
(1)  006020  042737  177760  001126             BIC     #-<17+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  006026  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  006034  001403                             BEQ     2$              ;SKIP IF OK
(1)  006036  104036                             ERROR+  N               ;IMSK BIT3-0 LOAD-READ ERROR
(1)  006040  004737  022234                     JSR     PC,DMPREG
(1)  006044  062737  000001  001124  2$:        ADD     #BIT0,$GDDAT    ;BUMP EXPECTED
(1)  006052  042737  177760  001124             BIC     #-<17+1>,$GDDAT ;MASK TO BITS UNDER TEST
(1)  006060  023727  001124  000017             CMP     $GDDAT,#17      ;DONE ALL?
(1)  006066  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  006070  012737  006076  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  006076  012777  000004  015466  3$:        MOV     #IMSKE,@ACR     ;RESET ADR TO IMSK
(1)  006104  012777  000017  015464             MOV     #17,@IMSK               ;SET ALL BITS
(1)  006112  012777  100000  015452             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  006120  012777  000004  015444             MOV     #IMSKE,@ACR     ;SET ADDR TO IMSK
(1)  006126  017737  015444  023610             MOV     @IMSK,DISPLY    ;READ IMSK
(1)  006134  032737  000017  023610             BIT     #17,DISPLY      ;CHECK BIT3-0 IN IMSK FOR 0
(1)  006142  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT3-0 IN IMSK
(1)  006144  104037                             ERROR+  N               ;BIT3-0 IN IMSK WERE NOT CLEARED BY RESET
(1)  006146  004737  022234                     JSR     PC,DMPREG
(1)  006152                             4$:
```

```
691
(4)                                     ;;*****************************************************************
(4)                                     ;*TEST 30       MTC (BIT11-8) WRITE-READ TEST USING MTC COUNT PATTERN
(4)                                     ;;*****************************************************************
(3)  006152  000004                     TST30:  SCOPE
(1)  006154  012777  100000  015410             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  006162  012737  006176  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  006170  012737  007400  001124             MOV     #7400,$GDDAT    ;INIT EXPECTED TO 7400
(1)  006176  012777  000015  015366     1$:     MOV     #MTCE,@ACR      ;SET ADR TO MTC
(1)  006204  013777  001124  015364             MOV     $GDDAT,@MTC     ;LOAD MTC WITH $GDDAT
(1)  006212  012777  000015  015352             MOV     #MTCE,@ACR      ;RESET ADR TO MTC
(1)  006220  017737  015352  023610             MOV     @MTC,DISPLY     ;READ MTC
(1)  006226  013737  023610  001126             MOV     DISPLY,$BDDAT   ;COPY MTC INTO $BDDAT
(1)  006234  042737  170377  001126             BIC     #-<7400+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  006242  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  006250  001403                             BEQ     2$              ;SKIP IF OK
(1)  006252  104040                             ERROR+  N               ;MTC BIT11-8 LOAD-READ ERROR
(1)  006254  004737  022234                     JSR     PC,DMPREG
(1)  006260  062737  000400  001124     2$:     ADD     #BIT8,$GDDAT    ;BUMP EXPECTED
(1)  006266  042737  170377  001124             BIC     #-<7400+1>,$GDDAT       ;MASK TO BITS UNDER TEST
(1)  006274  023727  001124  007400             CMP     $GDDAT,#7400    ;DONE ALL?
(1)  006302  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  006304  012737  006312  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  006312  012777  000015  015252     3$:     MOV     #MTCE,@ACR      ;RESET ADR TO MTC
(1)  006320  012777  007400  015250             MOV     #7400,@MTC              ;SET ALL BITS
(1)  006326  012777  100000  015236             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  006334  012777  000015  015230             MOV     #MTCE,@ACR      ;SET ADDR TO MTC
(1)  006342  017737  015230  023610             MOV     @MTC,DISPLY     ;READ MTC
(1)  006350  032737  007400  023610             BIT     #7400,DISPLY    ;CHECK BIT11-8 IN MTC FOR 0
(1)  006356  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT11-8 IN MTC
(1)  006360  104041                             ERROR+  N               ;BIT11-8 IN MTC WERE NOT CLEARED BY RESET
(1)  006362  004737  022234                     JSR     PC,DMPREG
(1)  006366                             4$:
```

```
693                                     ;;*********************************************************************
(4)                                     ;*TEST 31       MTC (BIT3-0) WRITE-READ TEST USING MTC COUNT PATTERN
(4)                                     ;;*********************************************************************
(3)  006366  000004                     TST31:  SCOPE
(1)  006370  012777  100000  015174             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  006376  012737  006412  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  006404  012737  000017  001124             MOV     #17,$GDDAT      ;INIT EXPECTED TO 17
(1)  006412  012777  000015  015152  1$:        MOV     #MTCE,@ACR      ;SET ADR TO MTC
(1)  006420  013777  001124  015150             MOV     $GDDAT,@MTC     ;LOAD MTC WITH $GDDAT
(1)  006426  012777  000015  015136             MOV     #MTCE,@ACR      ;RESET ADR TO MTC
(1)  006434  017737  015136  023610             MOV     @MTC,DISPLY     ;READ MTC
(1)  006442  013737  023610  001126             MOV     DISPLY,$BDDAT   ;COPY MTC INTO $BDDAT
(1)  006450  042737  177760  001126             BIC     #-<17+1>,$BDDAT ;CLEAR ALL BUT BITS UNDER TEST
(1)  006456  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  006464  001403                             BEQ     2$              ;SKIP IF OK
(1)  006466  104042                             ERROR+  N               ;MTC BIT3-0 LOAD-READ ERROR
(1)  006470  004737  022234                     JSR     PC,DMPREG
(1)  006474  062737  000001  001124  2$:        ADD     #BIT0,$GDDAT    ;BUMP EXPECTED
(1)  006502  042737  177760  001124             BIC     #-<17+1>,$GDDAT ;MASK TO BITS UNDER TEST
(1)  006510  023727  001124  000017             CMP     $GDDAT,#17      ;DONE ALL?
(1)  006516  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  006520  012737  006526  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  006526  012777  000015  015036  3$:        MOV     #MTCE,@ACR      ;RESET ADR TO MTC
(1)  006534  012777  000017  015034             MOV     #17,@MTC                ;SET ALL BITS
(1)  006542  012777  100000  015022             MOV     #CLR,@ACR       ;CLEAR THE WORLD
(1)  006550  012777  000015  015014             MOV     #MTCE,@ACR      ;SET ADDR TO MTC
(1)  006556  017737  015014  023610             MOV     @MTC,DISPLY     ;READ MTC
(1)  006564  032737  000017  023610             BIT     #17,DISPLY      ;CHECK BIT3-0 IN MTC FOR 0
(1)  006572  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT3-0 IN MTC
(1)  006574  104043                             ERROR+  N               ;BIT3-0 IN MTC WERE NOT CLEARED BY RESET
(1)  006576  004737  022234                     JSR     PC,DMPREG
(1)  006602                             4$:
694
695  006602  004737  023474                     JSR     PC,GETSID               ;DUMP SID ON PASS 1
```

```
                                ;DUAL ADDRESSING TEST
                                ;A COUNT PATTERN IS DRIVEN THROUGH THE REGISTER UNDER TEST
                                ;AND ALL OTHER REGISTERS ARE CHECKED FOR ANY CHANGE
                                ;ALL THE IIST REGISTERS ARE TESTED SEQUENTIALLY
                                ;THIS TEST IS NOT PERFORMED IF THERE ARE OTHER TRANSMITTING
                                ;;NON-INHIBITED UNITS ON THE IIST BUS (CONFIG=3), SINCE THE
                                ;REGISTER SNAPSHOT COULD BE INVALID.

                                ;;*****************************************************************
(3)                             ;*TEST 32       DUAL ADDRESSING TEST OF ALL REGISTERS
(3)                             ;;*****************************************************************
(2) 006606 000004               TST32:  SCOPE
706 006610 023727 023660 000003         CMP     CONFIG,#3       ;SEE IF CONFIG #3 (OTHER ACTIVE UNITS)
707 006616 001002                       BNE     11$             ;SKIP IF NO
708 006620 000137 007234                JMP     99$             ;DON'T DO TEST IF YES
709 006624 012705 000020       11$:     MOV     #16.,R5         ;SET COUNTER FOR 16 REGISTERS
710 006630 012704 000000               MOV     #0,R4           ;SET POINTER FOR REGISTER ADDRESS
711 006634 012703 023616               MOV     #SNAP0,R3       ;SET POINTER FOR INITIAL REGISTER DATA
712 006640 012777 100000 014724        MOV     #CLR,@ACR       ;CLEAR THE IIST
713 006646 005037 001176               CLR     $TMP0           ;CLEAR THE REGISTER NUMBER
714 006652 110477 014714       1$:     MOVB    R4,@ACR         ;SELECT REGISTER FROM TABLE
715 006656 017723 014714               MOV     @ADR,(3)+       ;PUT REGISTER CONTENTS INTO SNAP
716 006662 005204                      INC     R4              ;BUMP ADR CODE
717 006664 005305                      DEC     R5              ;DONE 10?
718 006666 001371                      BNE     1$              ;DO 10
719 006670 005000                      CLR     R0              ;CLEAR TEST DATA WORD
720 006672 010077 014674       2$:     MOV     R0,@ACR         ;LOAD DATA INTO REGISTER
721 006676 005200                      INC     R0              ;BUMP DATA
722 006700 100374                      BPL     2$              ;DO 0-77777. (DO NOT SET ''CLR'')
723 006702 012705 000020               MOV     #16.,R5         ;SET COUNTER FOR 16 REGISTERS
724 006706 012704 000000               MOV     #0,R4           ;SET POINTER FOR REGISTER ADDRESS TABLE
725 006712 012703 023616               MOV     #SNAP0,R3       ;SET POINTER FOR INITIAL REGISTER DATA
726 006716 110437 001176       3$:     MOVB    R4,$TMP0        ;SELECT REGISTER FROM TABLE
727 006722 013777 001176 014642        MOV     $TMP0,@ACR      ;LOAD ACR WITH ADDRESS
728 006730 017737 014642 001126        MOV     @ADR, $BDDAT    ;READ REGISTER INTO $BDDAT
729 006736 012337 001124               MOV     (3)+,$GDDAT     ;FETCH ORIGINAL REGISTER DATA
730 006742 023737 001124 001126        CMP     $GDDAT,$BDDAT   ;CHECK FOR ANY CHANGE IN REGISTER
731 006750 001403                      BEQ     4$              ;SKIP IF NO CHANGE
732 006752 104044                      ERROR+  44              ;ADDRESSING ACR ALSO ADDRESSED REGISTER IN $TMP0
733 006754 004737 022234               JSR     PC,DMPREG
734 006760 005204              4$:     INC     R4              ;BUMP ADR CODE
735 006762 005305                      DEC     R5              ;CHECKED ALL REGISTERS?
736 006764 001354                      BNE     3$              ;CHECK ALL REGISTERS
737 006766 012705 000020               MOV     #16.,R5         ;SET COUNTER FOR 16 REGISTERS
738 006772 012704 000000               MOV     #0,R4           ;SET POINTER FOR REGISTER ADDRESS
739 006776 012737 007010 001110        MOV     #5$,$LPERR      ;SET SCOPE LOOP POINT
740 007004 005037 001204               CLR     $TMP3           ;INITIALIZE $TMP3
741 007010 012777 100000 014554 5$:    MOV     #CLR,@ACR       ;CLEAR THE IIST
742 007016 012703 023616               MOV     #SNAP0,R3       ;SET POINTER FOR INITIAL REGISTER DATA
743 007022 012701 000020               MOV     #16.,R1         ;SET COUNTER TO READ 16 REGISTERS
744 007026 012702 000000               MOV     #0,R2           ;SET POINTER FOR REGISTER ADDRESS
745 007032 110277 014534       6$:     MOVB    R2,@ACR         ;SELECT REGISTER
746 007036 017723 014534               MOV     @ADR,(3)+       ;READ INITIAL REGISTER DATA
747 007042 005202                      INC     R2              ;BUMP ADR CODE
748 007044 005301                      DEC     R1              ;DONE 16?
749 007046 001371                      BNE     6$              ;DO 16
```

```
   750  007050  110437  001176          MOVB    R4,$TMP0        ;$TMP0=REGISTER BEING WRITTEN
   751  007054  005000                  CLR     R0              ;START AT 0
```

```
753  007056  113777  001176  014506  7$:     MOVB    $TMP0,@ACR      ;SELECT REGISTER
754  007064  010001                          MOV     R0,R1           ;COPY R0
755  007066  120427  000001                  CMPB    R4,#PGCSE       ;PGCS?
756  007072  001411                          BEQ     10$             ;SKIP IF YES
757  007074  120427  000003                  CMPB    R4,#STCSE       ;STCS?
758  007100  001406                          BEQ     10$             ;SKIP IF YES
759  007102  120427  000015                  CMPB    R4,#MTCE        ;MTC?
760  007106  001005                          BNE     20$             ;SKIP IF NO
761  007110  042701  000007                  BIC     #7,R1           ;CLEAR MAINT BITS
762  007114  000402                          BR      20$             ;SKIP
763  007116  042701  000001          10$:    BIC     #1,R1           ;DO NOT SET GO
764  007122  010177  014450          20$:    MOV     R1,@ADR         ;LOAD DATA INTO REGISTER
765  007126  005200                          INC     R0              ;BUMP DATA
766  007130  001352                          BNE     7$              ;LOAD 0-77777 INTO REGISTER
767  007132  012702  000000                  MOV     #0,R2           ;SET POINTER FOR REGISTER ADDRESS TABLE
768  007136  012701  000020                  MOV     #16.,R1         ;SET COUNTER FOR 16 REGISTERS
769  007142  012703  023616                  MOV     #SNAP0,R3       ;SET POINTER FOR INITIAL CONTENTS TABLE
770  007146  110277  014420          8$:     MOVB    R2,@ACR         ;SELECT A REGISTER
771  007152  012337  001124                  MOV     (3)+,$GDDAT     ;FETCH INITIAL CONTENTS OF REGISTER
772  007156  127737  014410  001176          CMPB    @ACR,$TMP0      ;IS SELECTED REGISTER THE ONE WRITTEN?
773  007164  001415                          BEQ     9$              ;SKIP TESTING IF YES
774  007166  017737  014404  001126          MOV     @ADR,$BDDAT     ;FETCH REGISTER DATA
775  007174  117737  014372  001200          MOVB    @ACR,$TMP1      ;SAVE ADDRESS OF REGISTER CHECKED IN $TMP1
776  007202  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK REGISTER FOR ANY CHANGES
777  007210  001403                          BEQ     9$              ;SKIP IF NO CHANGE
778  007212  104044                          ERROR+  44              ;ADDRESSING REGISTER IN $TMP0 ALSO
779  007214  004737  022234                  JSR     PC,DMPREG
780                                                                  ;ADDRESSED REGISTER IN $TMP1
781  007220  005202                  9$:     INC     R2              ;BUMP ADR CODE
782  007222  005301                          DEC     R1              ;CHECKED ALL REGISTERS?
783  007224  001350                          BNE     8$              ;CHECK ALL REGISTERS
784  007226  005204                          INC     R4              ;BUMP POINTER FOR REGISTER ADDRESS
785  007230  005305                          DEC     R5              ;DONE ALL?
786  007232  001266                          BNE     5$              ;DO ALL
787  007234                          99$:
```

```
                                        .SBTTL
                                        .SBTTL
                                        .SBTTL  INDIVIDUAL LINE TESTS
                                        .SBTTL

007234  012704  000000      LINE0:  MOV     #0,R4           ;SET INDEX TO 0
007240  004737  007324              JSR     PC,LINTST       ;TEST LINE
007244  012704  000001              MOV     #1,R4           ;SET INDEX TO 1
007250  004737  007324              JSR     PC,LINTST       ;TEST LINE
007254  012704  000002              MOV     #2,R4           ;SET INDEX TO 2
007260  004737  007324              JSR     PC,LINTST       ;TEST LINE
007264  012704  000003              MOV     #3,R4           ;SET INDEX TO 3
007270  004737  007324              JSR     PC,LINTST       ;TEST LINE
007274  013704  023570      LINE1:  MOV     SIDNUM,R4       ;SET INDEX TO ID #
007300  006304                      ASL     R4              ;SHIFT
007302  004737  016072              JSR     PC,PGONL        ;TEST PGF ONLINE
007306  013704  023570      LINE2:  MOV     SIDNUM,R4       ;SET INDEX TO ID #
007312  006304                      ASL     R4              ;SHIFT
007314  004737  017674              JSR     PC,STONL        ;TEST STF ONLINE
007320  000137  020546              JMP     ENDTST          ;SKIP OVER SUBROUTINE
```

```
                                LINTST: MOV     R4,LINMBR       ;SAVE LINE UNDER TEST
                                        ASL     R4              ;MAKE INDEX INTO A WORD POINTER

                                ;ISSUE A PROGRAM INTERRUPT REQUEST AND CHECK FOR:
                                ;RDY TO RESET
                                ;RDY TO SET
                                ;PIF BIT TO SET
                                ;DCF REGISTER TO STAY RESET
                                ;EXC REGISTER TO STAY RESET

                                ;;*****************************************************************
(3)                             ;*TEST 33       CHECK A LEGAL PGM PI REQUEST FOR PROPER OPERATION
(3)                             ;;*****************************************************************
(2) 007332  000004              TST33:  SCOPE
007334  005037  177776                  CLR     PSW             ;LOWER PRIORITY TO ALLOW INTERUPTS
007340  004737  022042                  JSR     PC,SETMNT       ;ENABLE MAINT LOOPBACK
007344  012777  000000  014220          MOV     #PGTEE,@ACR     ;SELECT PGTE
007352  016477  020526  014216          MOV     PGITAB(4),@PGTE ;LOAD PIE
007360  012777  000001  014210          MOV     #GO,@PGCS       ;TRANSMIT
007366  012777  000001  014176          MOV     #PGCSE,@ACR     ;SELECT PGCS
007374  017737  014176  001126          MOV     @PGCS,$BDDAT    ;READ PGCS
007402  032737  004000  001126          BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
007410  001406                          BEQ     1$              ;=0
007412  016437  020516  001124          MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED PGCS
007420  104044                          ERROR+  44              ;SETTING PGCS GO DID NOT CLEAR RDY
007422  004737  022234                  JSR     PC,DMPREG
007426  005000                  1$:     CLR     R0              ;INITIALIZE TIMER
007430  012777  000001  014134  2$:     MOV     #PGCSE,@ACR     ;SELECT PGCS
007436  017737  014134  001126          MOV     @PGCS,$BDDAT    ;READ PGCS
007444  032737  004000  001126          BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
007452  001013                          BNE     3$              ;SKIP IF SET
007454  005300                          DEC     R0              ;TIMER =0?
007456  001364                          BNE     2$              ;TRY AGAIN IF NO
007460  016437  020516  001124          MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED WITH
007466  062737  004000  001124          ADD     #PRDY,$GDDAT    ;SID + RDY
007474  104044                          ERROR+  44              ;PGCS RDY DID NOT SET ON A PGM PI
007476  004737  022234                  JSR     PC,DMPREG
007502  012777  000005  014062  3$:     MOV     #PGFE,@ACR      ;SELECT PGF
007510  017737  014062  001126          MOV     @PGF,$BDDAT     ;READ PGF
007516  012737  000017  001124          MOV     #17,$GDDAT      ;FETCH EXPECTED FLAG BITS
007524  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK FOR CORRECT FLAG
007532  001403                          BEQ     4$              ;SKIP IF OK
007534  104044                          ERROR+  44              ;PGM PI DID NOT SET CORRECT FLAG
007536  004737  022234                  JSR     PC,DMPREG
007542  012777  000007  014022  4$:     MOV     #DCFE,@ACR      ;SELECT DCF
007550  017737  014022  023610          MOV     @DCF,DISPLY     ;READ DCF
007556  001403                          BEQ     5$              ;SKIP IF 0
007560  104044                          ERROR+  44              ;PGM PI SET A DCF CONDITION
007562  004737  022234                  JSR     PC,DMPREG
007566  012777  000010  013776  5$:     MOV     #EXCE,@ACR              ;SELECT EXC
007574  017737  013776  023610          MOV     @EXC,DISPLY     ;READ EXC
007602  001403                          BEQ     6$              ;SKIP IF 0
007604  104044                          ERROR+  44              ;PGM PI SET AN EXC CONDITION
007606  004737  022234                  JSR     PC,DMPREG
007612  012777  000006  013752  6$:     MOV     #STFE,@ACR      ;SELECT STF
007620  017737  013752  023610          MOV     @STF,DISPLY     ;READ STF
```

```
  863  007626  001403                    BEQ     7$              ;SKIP IF CLEAR
  864  007630  104044                    ERROR+  44              ;PGM PI SET AN STF CONDITION
  865  007632  004737  022234            JSR     PC,DMPREG
```

```
867  007636  012700  000017          7$:   MOV   #17,R0          ;SET ALL 4 LINES
868  007642  046400  020526                BIC   PGITAB(4),R0    ;CLEAR LINE UNDER TEST
869  007646  012777  000005  013716        MOV   #PGFE,@ACR      ;SELECT PGF
870  007654  010077  013716                MOV   R0,@PGF         ;CLEAR ALL PIFS EXCEPT LINE UNDER TEST
871  007660  012777  000005  013704        MOV   #PGFE,@ACR      ;SELECT PGF
872  007666  017737  013704  001202        MOV   @PGF,$TMP2      ;READ PGF
873  007674  012777  000001  013670        MOV   #PGCSE,@ACR     ;SELECT PGCS
874  007702  017737  013670  023610        MOV   @PGCS,DISPLY    ;READ PGCS
875  007710  032737  000010  023610        BIT   #IP,DISPLY      ;CHECK FOR IP
876  007716  001003                        BNE   8$              ;SKIP IF SET
877  007720  104044                        ERROR+ 44             ;IP DID NOT SET ON PGF
878  007722  004737  022234                JSR   PC,DMPREG
879  007726  017746  013652          8$:   MOV   @IIIP,-(SP)    ;SAVE OLD
880  007732  017746  013644                MOV   @IIIV,-(SP)     ;PI INFORMATION
881  007736  012777  007770  013636        MOV   #9$,@IIIV       ;LOAD NEW PI VECTOR
882  007744  012777  000340  013632        MOV   #340,@IIIP      ;LOAD NEW INT. LEVEL
883  007752  005037  177776                CLR   PS              ;ALLOW PI
884  007756  000240                        NOP                   ;STALL
885  007760  012737  000340  177776        MOV   #340,PS         ;LOCKOUT PI
886  007766  000404                        BR    10$             ;SKIP OVER ERROR
887  007770  022626          9$:   POPPOP                        ;FIX STACK
888  007772  104044                        ERROR+ 44             ;PGF PI WITH IE=0
889  007774  004737  022234                JSR   PC,DMPREG
890  010000  012777  010074  013574  10$:   MOV   #12$,@IIIV     ;LOAD NEW PI VECTOR
891  010006  012777  000001  013556        MOV   #PGCSE,@ACR     ;SELECT PGCS
892  010014  012777  000004  013554        MOV   #IE,@PGCS       ;SET IE
893  010022  012737  000340  001212        MOV   #340,$TMP6      ;INIT PI LEVEL
894  010030  162737  000040  001212  11$:   SUB   #40,$TMP6      ;DROP PI LEVEL BY 1
895  010036  013737  001212  177776        MOV   $TMP6,PS        ;LOWER PSW
896  010044  000240                        NOP                   ;STALL
897  010046  023727  001212  000140        CMP   $TMP6,#140      ;TRIED ALL LEVELS?
898  010054  001365                        BNE   11$             ;BR IF NO
899  010056  012737  000340  177776        MOV   #340,PS         ;LOCKOUT PI
900  010064  104044                        ERROR+ 44             ;NO PGF PI WITH IE=1
901  010066  004737  022234                JSR   PC,DMPREG
902  010072  000422                        BR    13$             ;SKIP NEXT TEST
903  010074  022626          12$:  POPPOP                        ;FIX STACK
904  010076  006337  001212                ASL   $TMP6           ;RIGHT JUSTIFY
905  010102  006337  001212                ASL   $TMP6
906  010106  006337  001212                ASL   $TMP6
907  010112  000337  001212                SWAB  $TMP6           ;THE PI LEVEL
908  010116  005237  001212                INC   $TMP6           ;ADD 1 TO MAKE CORECT LEVEL
909  010122  023737  001212  023606        CMP   $TMP6,IIIL      ;CHECK AGAINST KNOWN LEVEL
910  010130  001403                        BEQ   13$             ;SKIP IF OK
911  010132  104044                        ERROR+ 44             ;PGF PI TO THE WRONG LEVEL
912  010134  004737  022234                JSR   PC,DMPREG
913  010140  012677  013436          13$:  MOV   (SP)+,@IIIV     ;RESTORE
914  010144  012677  013434                MOV   (SP)+,@IIIP     ;OLD PI INFO
915  010150  012777  000005  013414  14$:   MOV   #PGFE,@ACR      ;SELECT PGF
916  010156  016437  020526  001124        MOV   PIFTAB(4),$GDDAT;FETCH FLAG BIT
917  010164  013777  001124  013404        MOV   $GDDAT,@PGF     ;CLEAR FLAG
918  010172  005037  001124                CLR   $GDDAT          ;CLEAR EXPECTED
919  010176  012777  000005  013366        MOV   #PGFE,@ACR      ;SELECT PGF
920  010204  017737  013366  001126        MOV   @PGF,$BDDAT     ;READ PGF
921  010212  001403                        BEQ   15$             ;SKIP IF OK
922  010214  104044                        ERROR+ 44             ;WRITING A 1 TO PGF FLAG DID NOT
```

```
923  010216  004737  022234                  JSR     PC,DMPREG
924
925  010222                          15$:                        ;CLEAR IT
```

```
                                ;ISSUE A PROGRAM BOOT REQUEST AND CHECK FOR:
                                ;RDY TO SET
                                ;PBF BIT TO SET
                                ;DCF REGISTER TO STAY RESET
                                ;EXC REGISTER TO STAY RESET

                                ;;*****************************************************************
(3)                             ;*TEST 34       CHECK A LEGAL PGM BOOT REQUEST FOR PROPER OPERATION
(3)                             ;;*****************************************************************
(2) 010222 000004               TST34:  SCOPE
934 010224 004737 022042                JSR     PC,SETMNT       ;ENABLE MAINT LOOPBACK
935 010230 012777 000000 013334         MOV     #PGTEE,@ACR     ;SELECT PGTE
936 010236 016477 020536 013332         MOV     PGBTAB(4),@PGTE ;LOAD PIB
937 010244 012777 000001 013324         MOV     #GO,@PGCS       ;TRANSMIT
938 010252 005000               1$:     CLR     R0              ;INITIALIZE TIMER
939 010254 012777 000001 013310 2$:     MOV     #PGCSE,@ACR     ;SELECT PGCS
940 010262 117737 013310 001126         MOV     @PGCS,$BDDAT    ;READ PGCS
941 010270 032737 004000 001126         BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
942 010276 001013                       BNE     3$              ;SKIP IF SET
943 010300 005300                       DEC     R0              ;TIMER =0?
944 010302 001364                       BNE     2$              ;TRY AGAIN IF NO
945 010304 016437 020516 001124         MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED WITH
946 010312 062737 004000 001124         ADD     #PRDY,$GDDAT    ;SID + RDY
947 010320 104044                       ERROR+  44              ;PGCS RDY DID NOT SET ON A PGM BOOT
948 010322 004737 022234                JSR     PC,DMPREG
949 010326 012777 000005 013236 3$:     MOV     #PGFE,@ACR      ;SELECT PGF
950 010334 017737 013236 001126         MOV     @PGF,$BDDAT     ;READ PGF
951 010342 012737 007400 001124         MOV     #7400,$GDDAT    ;FETCH EXPECTED FLAG BITS
952 010350 023737 001124 001126         CMP     $GDDAT,$BDDAT   ;CHECK FOR CORRECT FLAG
953 010356 001403                       BEQ     4$              ;SKIP IF OK
954 010360 104044                       ERROR+  44              ;PGM BOOT DID NOT SET CORRECT FLAG
955 010362 004737 022234                JSR     PC,DMPREG
956 010366 012777 000007 013176 4$:     MOV     #DCFE,@ACR      ;SELECT DCF
957 010374 017737 013176 023610         MOV     @DCF,DISPLY     ;READ DCF
958 010402 001403                       BEQ     5$              ;SKIP IF 0
959 010404 104044                       ERROR+  44              ;PGM BOOT SET A DCF CONDITION
960 010406 004737 022234                JSR     PC,DMPREG
961 010412 012777 000010 013152 5$:     MOV     #EXCE,@ACR      ;SELECT EXC
962 010420 017737 013152 023610         MOV     @EXC,DISPLY     ;READ EXC
963 010426 001403                       BEQ     6$              ;SKIP IF 0
964 010430 104044                       ERROR+  44              ;PGM BOOT SET AN EXC CONDITION
965 010432 004737 022234                JSR     PC,DMPREG
966 010436 012777 000006 013126 6$:     MOV     #STFE,@ACR      ;SELECT STF
967 010444 017737 013126 023610         MOV     @STF,DISPLY     ;READ STF
968 010452 001403                       BEQ     7$              ;SKIP IF 0
969 010454 104044                       ERROR+  44              ;PGM BOOT SET AN STF CONDITION
970 010456 004737 022234                JSR     PC,DMPREG
971 010462 012777 000005 013102 7$:     MOV     #PGFE,@ACR      ;SELECT PGF
972 010470 016437 020536 001124         MOV     PBFTAB(4),$GDDAT;FETCH FLAG BIT
973 010476 013777 001124 013072         MOV     $GDDAT,@PGF     ;CLEAR FLAG
974 010504 005137 001124                COM     $GDDAT          ;FLIP GDDAT
975 010510 042737 170377 001124         BIC     #170377,$GDDAT  ;MASK OUT JUNK
976 010516 012777 000005 013046         MOV     #PGFE,@ACR      ;SELECT PGF
977 010524 017737 013046 001126         MOV     @PGF,$BDDAT     ;READ PGF
978 010532 023737 001124 001126         CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
979 010540 001403                       BEQ     8$              ;SKIP IF OK
```

```
   980  010542  104044                    ERROR+  44              ;WRITING A 1 TO PGF FLAG DID NOT
   981  010544  004737  022234            JSR     PC,DMPREG
   982                                                             ;CLEAR IT
   983  010550                    8$:
```

```
                                ;ISSUE A SANITY TIMER INTERRUPT REQUEST AND CHECK FOR:
                                ;RDY TO RESET
                                ;RDY TO SET
                                ;SIF BIT TO SET
                                ;DCF REGISTER TO STAY RESET
                                ;EXC REGISTER TO STAY RESET
                                ;PGF REGISTER TO STAY RESET
                                ;SIF BIT TO BE CLEARED BY WRITING A 1 TO IT

                                ;;*****************************************************************
                                ;*TEST 35       CHECK A LEGAL ST INTR REQUEST FOR PROPER OPERATION
                                ;;*****************************************************************
(2) 010550 000004               TST35:  SCOPE
 995 010552 004737 022204               JSR     PC,SSTMNT       ;ENABLE MAINT LOOPBACK
 996 010556 012777 000000 013006        MOV     #PGTEE,@ACR     ;SELECT PGTE
 997 010564 016477 020526 013004        MOV     PGITAB(4),@PGTE ;LOAD PIE
 998 010572 012777 000003 012776        MOV     #PTP+GO,@PGCS   ;TRANSMIT
 999 010600 012777 000001 012764        MOV     #PGCSE,@ACR     ;SELECT PGCS
1000 010606 017737 012764 001126        MOV     @PGCS,$BDDAT    ;READ PGCS
1001 010614 032737 004000 001126        BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
1002 010622 001406                      BEQ     1$              ;=0
1003 010624 016437 020516 001124        MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED PGCS
1004 010632 104044                      ERROR+  44              ;SETTING PGCS GO DID NOT CLEAR RDY
1005 010634 004737 022234               JSR     PC,DMPREG
1006 010640 005000              1$:     CLR     R0              ;INITIALIZE TIMER
1007 010642 012777 000001 012722 2$:    MOV     #PGCSE,@ACR     ;SELECT PGCS
1008 010650 017737 012722 001126        MOV     @PGCS,$BDDAT    ;READ PGCS
1009 010656 032737 004000 001126        BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
1010 010664 001013                      BNE     3$              ;SKIP IF SET
1011 010666 005300                      DEC     R0              ;TIMER =0?
1012 010670 001364                      BNE     2$              ;TRY AGAIN IF NO
1013 010672 016437 020516 001124        MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED WITH
1014 010700 062737 004000 001124        ADD     #PRDY,$GDDAT    ;SID + RDY
1015 010706 104044                      ERROR+  44              ;PGCS RDY DID NOT SET ON A PGM PI
1016 010710 004737 022234               JSR     PC,DMPREG
1017 010714 012777 000006 012650 3$:    MOV     #STFE,@ACR      ;SELECT STF
1018 010722 017737 012650 001126        MOV     @STF,$BDDAT     ;READ STF
1019 010730 012737 000017 001124        MOV     #17,$GDDAT      ;FETCH EXPECTED FLAG BITS
1020 010736 023737 001124 001126        CMP     $GDDAT,$BDDAT   ;CHECK FOR CORRECT FLAG
1021 010744 001403                      BEQ     4$              ;SKIP IF OK
1022 010746 104044                      ERROR+  44              ;ST INTR DID NOT SET CORRECT FLAG
1023 010750 004737 022234               JSR     PC,DMPREG
1024 010754 012777 000007 012610 4$:    MOV     #DCFE,@ACR      ;SELECT DCF
1025 010762 017737 012610 023610        MOV     @DCF,DISPLY     ;READ DCF
1026 010770 001403                      BEQ     5$              ;SKIP IF 0
1027 010772 104044                      ERROR+  44              ;ST INTR SET A DCF CONDITION
1028 010774 004737 022234               JSR     PC,DMPREG
1029 011000 012777 000010 012564 5$:    MOV #EXCE,@ACR          ;SELECT EXC
1030 011006 017737 012564 023610        MOV     @EXC,DISPLY     ;READ EXC
1031 011014 001403                      BEQ     6$              ;SKIP IF 0
1032 011016 104044                      ERROR+  44              ;ST INTR SET AN EXC CONDITION
1033 011020 004737 022234               JSR     PC,DMPREG
1034 011024 012777 000005 012540 6$:    MOV     #PGFE,@ACR      ;SELECT PGF
1035 011032 017737 012540 023610        MOV     @PGF,DISPLY     ;READ PGF
1036 011040 001403                      BEQ     7$              ;SKIP IF CLEAR
1037 011042 104044                      ERROR+  44              ;ST INTR SET A PGF CONDITION
```

```
                                                    K 4
1038  011044  004737  022234                JSR     PC,DMPREG
```

```
1040  011050  012700  000017          7$:   MOV   #17,R0          ;SET ALL 4 LINES
1041  011054  046400  020526                BIC   STITAB(4),R0    ;CLEAR LINE UNDER TEST
1042  011060  012777  000006  012504        MOV   #STFE,@ACR      ;SELECT STF
1043  011066  010077  012504               MOV   R0,@STF         ;CLEAR ALL SIFS EXCEPT LINE UNDER TEST
1044  011072  012777  000006  012472        MOV   #STFE,@ACR      ;SELECT STF
1045  011100  017737  012472  001202        MOV   @STF,$TMP2      ;READ STF
1046  011106  012777  000001  012456        MOV   #PGCSE,@ACR     ;SELECT PGCS
1047  011114  017737  012456  023610        MOV   @PGCS,DISPLY    ;READ PGCS
1048  011122  032737  000010  023610        BIT   #IP,DISPLY      ;CHECK FOR IP
1049  011130  001003                       BNE   8$              ;SKIP IF SET
1050  011132  104044                       ERROR+ 44             ;IP DID NOT SET ON STF
1051  011134  004737  022234               JSR   PC,DMPREG
1052  011140  017746  012440          8$:   MOV   @IIIP,-(SP)    ;SAVE OLD
1053  011144  017746  012432                MOV   @IIIV,-(SP)     ;PI INFORMATION
1054  011150  012777  011202  012424        MOV   #9$,@IIIV       ;LOAD NEW PI VECTOR
1055  011156  012777  000340  012420        MOV   #340,@IIIP      ;LOAD NEW INT. LEVEL
1056  011164  005037  177776               CLR   PS              ;ALLOW PI
1057  011170  000240                       NOP                   ;STALL
1058  011172  012737  000340  177776        MOV   #340,PS         ;LOCKOUT PI
1059  011200  000404                       BR    10$             ;SKIP OVER ERROR
1060  011202  022626                9$:    POPPOP                ;FIX STACK
1061  011204  104044                       ERROR+ 44             ;STF PI WITH IE=0
1062  011206  004737  022234               JSR   PC,DMPREG
1063  011212  012777  011260  012362  10$:   MOV   #11$,@IIIV      ;LOAD NEW PI VECTOR
1064  011220  012777  000001  012344        MOV   #PGCSE,@ACR     ;SELECT PGCS
1065  011226  012777  000004  012342        MOV   #IE,@PGCS       ;SET IE
1066  011234  005037  177776               CLR   PS              ;ALLOW PI
1067  011240  000240                       NOP                   ;STALL
1068  011242  012737  000340  177776        MOV   #340,PS         ;LOCKOUT PI
1069  011250  104044                       ERROR+ 44             ;NO STF PI WITH IE=1
1070  011252  004737  022234               JSR   PC,DMPREG
1071  011256  000401                       BR    12$             ;SKIP NEXT TEST
1072  011260  022626                11$:   POPPOP                ;FIX STACK
1073  011262  012677  012314          12$:   MOV   (SP)+,@IIIV     ;RESTORE
1074  011266  012677  012312                MOV   (SP)+,@IIIP     ;OLD PI INFO
1075  011272  012777  000006  012272  13$:   MOV   #STFE,@ACR      ;SELECT STF
1076  011300  016437  020526  001124        MOV   SIFTAB(4),$GDDAT;FETCH FLAG BIT
1077  011306  013777  001124  012262        MOV   $GDDAT,@STF     ;CLEAR FLAG
1078  011314  005037  001124               CLR   $GDDAT          ;CLEAR EXPECTED
1079  011320  012777  000006  012244        MOV   #STFE,@ACR      ;SELECT STF
1080  011326  017737  012244  001126        MOV   @STF,$BDDAT     ;READ STF
1081  011334  001403                       BEQ   14$             ;SKIP IF OK
1082  011336  104044                       ERROR+ 44             ;WRITING A 1 TO STF FLAG DID NOT
1083  011340  004737  022234               JSR   PC,DMPREG
1084                                                             ;CLEAR IT
1085  011344                         14$:
```

```
                                        ;ISSUE A SANITY TIMER BOOT REQUEST AND CHECK FOR:
                                        ;RDY TO SET
                                        ;SBF BIT TO SET
                                        ;DCF REGISTER TO STAY RESET
                                        ;EXC REGISTER TO STAY RESET

                                        ;;*****************************************************************
                                        ;*TEST 36       CHECK A LEGAL ST BOOT REQUEST FOR PROPER OPERATION
                                        ;;*****************************************************************
011344  000004                  TST36:  SCOPE
011346  004737  022204                  JSR     PC,SSTMNT       ;ENABLE MAINT LOOPBACK
011352  012777  000000  012212          MOV     #PGTEE,@ACR     ;SELECT PGTE
011360  016477  020536  012210          MOV     PGBTAB(4),@PGTE ;LOAD PIB
011366  012777  000003  012202          MOV     #PTP+GO,@PGCS   ;TRANSMIT
011374  005000                  1$:     CLR     R0              ;INITIALIZE TIMER
011376  012777  000001  012166  2$:     MOV     #PGCSE,@ACR     ;SELECT PGCS
011404  017737  012166  001126          MOV     @PGCS,$BDDAT    ;READ PGCS
011412  032737  004000  001126          BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
011420  001013                          BNE     3$              ;SKIP IF SET
011422  005300                          DEC     R0              ;TIMER =0?
011424  001364                          BNE     2$              ;TRY AGAIN IF NO
011426  016437  020516  001124          MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED WITH
011434  062737  004000  001124          ADD     #PRDY,$GDDAT    ;SID + RDY
011442  104044                          ERROR+  44              ;PGCS RDY DID NOT SET ON A ST BOOT
011444  004737  022234                  JSR     PC,DMPREG
011450  012777  000006  012114  3$:     MOV     #STFE,@ACR      ;SELECT STF
011456  017737  012114  001126          MOV     @STF,$BDDAT     ;READ STF
011464  012737  007400  001124          MOV     #7400,$GDDAT    ;FETCH EXPECTED FLAG BITS
011472  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK FOR CORRECT FLAG
011500  001403                          BEQ     4$              ;SKIP IF OK
011502  104044                          ERROR+  44              ;ST BOOT DID NOT SET CORRECT FLAG
011504  004737  022234                  JSR     PC,DMPREG
011510  012777  000007  012054  4$:     MOV     #DCFE,@ACR      ;SELECT DCF
011516  017737  012054  023610          MOV     @DCF,DISPLY     ;READ DCF
011524  001403                          BEQ     5$              ;SKIP IF 0
011526  104044                          ERROR+  44              ;ST BOOT SET A DCF CONDITION
011530  004737  022234                  JSR     PC,DMPREG
011534  012777  000010  012030  5$:     MOV     #EXCE,@ACR      ;SELECT EXC
011542  017737  012030  023610          MOV     @EXC,DISPLY     ;READ EXC
011550  001403                          BEQ     6$              ;SKIP IF 0
011552  104044                          ERROR+  44              ;ST BOOT SET AN EXC CONDITION
011554  004737  022234                  JSR     PC,DMPREG
011560  012777  000005  012004  6$:     MOV     #PGFE,@ACR      ;SELECT PGF
011566  017737  012004  023610          MOV     @PGF,DISPLY     ;READ PGF
011574  001403                          BEQ     7$              ;SKIP IF 0
011576  104044                          ERROR+  44              ;ST BOOT SET A PGF CONDITION
011600  004737  022234                  JSR     PC,DMPREG
011604  012777  000006  011760  7$:     MOV     #STFE,@ACR      ;SELECT STF
011612  016437  020536  001124          MOV     SBFTAB(4),$GDDAT;FETCH FLAG BIT
011620  013777  001124  011750          MOV     $GDDAT,@STF     ;CLEAR FLAG
011626  005137  001124                  COM     $GDDAT          ;FLIP GDDAT
011632  042737  170377  001124          BIC     #170377,$GDDAT  ;MASK OUT JUNK
011640  012777  000006  011724          MOV     #STFE,@ACR      ;SELECT STF
011646  017737  011724  001126          MOV     @STF,$BDDAT     ;READ STF
011654  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
011662  001403                          BEQ     8$              ;SKIP IF OK
```

```
1140  011664  104044                          ERROR+  44              ;WRITING A 1 TO STF FLAG DID NOT
1141  011666  004737  022234                   JSR     PC,DMPREG
1142                                                                  ;CLEAR IT
1143  011672                          8$:
```

```
                                ;ISSUE A PROGRAM INTERRUPT REQUEST WITH IMSK BIT =1 AND CHECK:
                                ;PIF BIT TO STAY 0
                                ;UI BIT TO SET
                                ;UI BIT TO CLEAR WHEN WRITTEN TO A 1

                                ;;*****************************************************************
                                ;*TEST 37       CHECK THE IMSK BIT FOR PROPER OPERATION
                                ;;*****************************************************************
(2) 011672  000004              TST37:  SCOPE
1151 011674  004737  022042             JSR     PC,SETMNT       ;ENABLE MAINT. LOOP
1152 011700  012777  000000  011664     MOV     #PGTEE,@ACR     ;SELECT PGTE
1153 011706  016477  020526  011662     MOV     PGITAB(4),@PGTE ;LOAD PIB
1154 011714  012777  000004  011650     MOV     #IMSKE,@ACR     ;SELECT IMSK
1155 011722  016477  020526  011646     MOV     IMITAB(4),@IMSK;INHIBIT LINE
1156 011730  012777  000001  011634     MOV     #PGCSE,@ACR     ;SELECT PGCS
1157 011736  012777  000001  011632     MOV     #GO,@PGCS       ;TRANSMIT
1158 011744  012777  000001  011620  1$: MOV     #PGCSE,@ACR     ;SELECT PGCS
1159 011752  032777  004000  011616     BIT     #PRDY,@PGCS     ;CHECK FOR
1160 011760  001771                     BEQ     1$              ;READY
1161 011762  012777  000005  011602     MOV     #PGFE,@ACR      ;SELECT PGF
1162 011770  017737  011602  001126     MOV     @PGF,$BDDAT     ;READ PGF
1163 011776  012737  000017  001124     MOV     #17,$GDDAT      ;FETCH EXPECTED FLAGS
1164 012004  046437  020526  001124     BIC     PGITAB(4),$GDDAT;MINUS EXPECTED FLAG
1165 012012  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
1166 012020  001403                     BEQ     2$              ;SKIP IF OK
1167 012022  104044                     ERROR+  44              ;IMSK BIT DID NOT INHIBIT ITS PGF BIT
1168 012024  004737  022234             JSR     PC,DMPREG
1169 012030  016437  020536  001124  2$: MOV     UVITAB(4),$GDDAT;LOAD EXPECTED UI BIT
1170 012036  012777  000010  011526     MOV     #EXCE,@ACR      ;SELECT EXC
1171 012044  017737  011526  001126     MOV     @EXC,$BDDAT     ;READ EXC
1172 012052  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;CHECK THE UI BIT
1173 012060  001403                     BEQ     3$              ;SKIP IF SET
1174 012062  104044                     ERROR+  44              ;UI DID NOT SET ON A PIF WITH IMSK=1
1175 012064  004737  022234             JSR     PC,DMPREG
1176 012070  005037  001124  3$:        CLR     $GDDAT          ;EXPECT 0
1177 012074  012777  000010  011470     MOV     #EXCE,@ACR      ;SELECT EXC
1178 012102  016477  020536  011466     MOV     UVITAB(4),@EXC  ;CLEAR UI BIT
1179 012110  012777  000010  011454     MOV     #EXCE,@ACR      ;SELECT EXC
1180 012116  017737  011454  023610     MOV     @EXC,DISPLY     ;READ EXC
1181 012124  013737  023610  001126     MOV     DISPLY,$BDDAT   ;TEST EXC
1182 012132  001403                     BEQ     4$              ;SKIP IF 0
1183 012134  104044                     ERROR+  44              ;WRITING A 1 TO UI BIT DID NOT CLEAR IT
1184 012136  004737  022234             JSR     PC,DMPREG
1185 012142  012777  000006  011422  4$: MOV     #STFE,@ACR      ;SELECT STF REGISTER
1186 012150  017737  011422  001126     MOV     @STF,$BDDAT     ;GET AND TEST STF REGISTER
1187 012156  001403                     BEQ     5$              ;STF SHOULD BE 0
1188 012160  104044                     ERROR+  44              ;STF NOT ZERO ON A PGI WITH IMSK=1
1189 012162  004737  022234             JSR     PC,DMPREG
1190 012166                      5$:
```

```
                                        ;ISSUE A PROGRAM BOOT REQUEST WITH BIMSK BIT =1 AND CHECK:
                                        ;PBF BIT TO STAY 0
                                        ;UI BIT TO SET

                                        ;;*****************************************************************
                                        ;*TEST 40       CHECK THE IMSK BIT FOR PROPER OPERATION ON PGM BOOT
                                        ;;*****************************************************************
 (2) 012166  000004                     TST40:  SCOPE
1197 012170  004737  022042                     JSR     PC,SETMNT       ;ENABLE MAINT. LOOP
1198 012174  012777  000000  011370             MOV     #PGTEE,@ACR     ;SELECT PGTE
1199 012202  016477  020536  011366             MOV     PGBTAB(4),@PGTE ;LOAD PBB
1200 012210  012777  000004  011354             MOV     #IMSKE,@ACR     ;SELECT IMSK
1201 012216  016477  020536  011352             MOV     BMITAB(4),@IMSK ;INHIBIT LINE
1202 012224  012777  000001  011340             MOV     #PGCSE,@ACR     ;SELECT PGCS
1203 012232  012777  000001  011336             MOV     #GO,@PGCS       ;TRANSMIT
1204 012240  012777  000001  011324  1$:        MOV     #PGCSE,@ACR     ;SELECT PGCS
1205 012246  032777  004000  011322             BIT     #PRDY,@PGCS     ;CHECK FOR
1206 012254  001771                             BEQ     1$              ;READY
1207 012256  012777  000005  011306             MOV     #PGFE,@ACR      ;SELECT PGF
1208 012264  017737  011306  001126             MOV     @PGF,$BDDAT     ;READ PGF
1209 012272  012737  007400  001124             MOV     #7400,$GDDAT    ;FETCH EXPECTED FLAGS
1210 012300  046437  020536  001124             BIC     PGBTAB(4),$GDDAT;MINUS EXPECTED FLAG
1211 012306  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
1212 012314  001403                             BEQ     2$              ;SKIP IF OK
1213 012316  104044                             ERROR+  44              ;IMSK BIT DID NOT INHIBIT ITS PGF BIT
1214 012320  004737  022234                     JSR     PC,DMPREG
1215 012324  016437  020536  001124  2$:        MOV     UVITAB(4),$GDDAT;LOAD EXPECTED UI BIT
1216 012332  012777  000010  011232             MOV     #EXCE,@ACR      ;SELECT EXC
1217 012340  017737  011232  001126             MOV     @EXC,$BDDAT     ;READ EXC
1218 012346  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK THE UI BIT
1219 012354  001403                             BEQ     3$      ;SKIP IF SET
1220 012356  104044                             ERROR+  44              ;UI DID NOT SET ON A PIF WITH IMSK=1
1221 012360  004737  022234                     JSR     PC,DMPREG
1222 012364  005037  001124  3$:        CLR     $GDDAT                  ;EXPECT STF TO BE 0
1223 012370  012777  000006  011174             MOV     #STFE,@ACR      ;SELECT STF
1224 012376  017737  011174  001126             MOV     @STF,$BDDAT     ;GET AND TEST STF
1225 012404  001403                             BEQ     4$              ;IT SHOULD BE 0
1226 012406  104044                             ERROR+  44
1227 012410  004737  022234                     JSR     PC,DMPREG
1228 012414                     4$:
```

```
                                ;ISSUE A SANITY TIMER INTERRUPT REQUEST WITH IMSK BIT =1 AND CHECK:
                                ;SIF BIT TO STAY 0
                                ;UI BIT TO SET
                                ;UI BIT TO CLEAR WHEN WRITTEN TO A 1

                                ;;*********************************************************************
                                ;*TEST 41        CHECK THE IMSK BIT FOR PROPER OPERATION ON ST INTR
                                ;;*********************************************************************
 (2) 012414  000004             TST41:  SCOPE
1236 012416  004737  022204             JSR     PC,SSTMNT       ;ENABLE MAINT. LOOP
1237 012422  012777  000000  011142     MOV     #PGTEE,@ACR     ;SELECT PGTE
1238 012430  016477  020526  011140     MOV     PGITAB(4),@PGTE ;LOAD PIB
1239 012436  012777  000004  011126     MOV     #IMSKE,@ACR     ;SELECT IMSK
1240 012444  016477  020526  011124     MOV     IMITAB(4),@IMSK;INHIBIT LINE
1241 012452  012777  000001  011112     MOV     #PGCSE,@ACR     ;SELECT PGCS
1242 012460  012777  000003  011110     MOV     #PTP+GO,@PGCS   ;TRANSMIT
1243 012466  012777  000001  011076 1$:  MOV     #PGCSE,@ACR     ;SELECT PGCS
1244 012474  032777  004000  011074     BIT     #PRDY,@PGCS     ;CHECK FOR
1245 012502  001771                     BEQ     1$              ;READY
1246 012504  012777  000006  011060     MOV     #STFE,@ACR      ;SELECT STF
1247 012512  017737  011060  001126     MOV     @STF,$BDDAT     ;READ STF
1248 012520  012737  000017  001124     MOV     #17,$GDDAT      ;FETCH EXPECTED FLAGS
1249 012526  046437  020526  001124     BIC     PGITAB(4),$GDDAT;MINUS EXPECTED FLAG
1250 012534  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
1251 012542  001403                     BEQ     2$              ;SKIP IF OK
1252 012544  104044                     ERROR+  44              ;IMSK BIT DID NOT INHIBIT ITS SIF BIT
1253 012546  004737  022234             JSR     PC,DMPREG
1254 012552  016437  020536  001124 2$:  MOV     UVITAB(4),$GDDAT;LOAD EXPECTED UI BIT
1255 012560  012777  000010  011004     MOV     #EXCE,@ACR      ;SELECT EXC
1256 012566  017737  011004  001126     MOV     @EXC,$BDDAT     ;READ EXC
1257 012574  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;CHECK THE UI BIT
1258 012602  001403                     BEQ     3$              ;SKIP IF SET
1259 012604  104044                     ERROR+  44              ;UI DID NOT SET ON A SIF WITH IMSK=1
1260 012606  004737  022234             JSR     PC,DMPREG
1261 012612  005037  001124      3$:    CLR     $GDDAT          ;EXPECT 0
1262 012616  012777  000010  010746     MOV     #EXCE,@ACR      ;SELECT EXC
1263 012624  016477  020536  010744     MOV     UVITAB(4),@EXC  ;CLEAR UI BIT
1264 012632  012777  000010  010732     MOV     #EXCE,@ACR      ;SELECT EXC
1265 012640  017737  010732  023610     MOV     @EXC,DISPLY     ;READ EXC
1266 012646  013737  023610  001126     MOV     DISPLY,$BDDAT   ;TEST EXC
1267 012654  001403                     BEQ     4$              ;SKIP IF 0
1268 012656  104044                     ERROR+  44              ;WRITING A 1 TO UI BIT DID NOT CLEAR IT
1269 012660  004737  022234             JSR     PC,DMPREG
1270 012664  012777  000005  010700 4$:  MOV     #PGFE,@ACR      ;SELECT PGF REGIPGER
1271 012672  017737  010700  001126     MOV     @PGF,$BDDAT     ;GET AND TEST STF REGISTER
1272 012700  001403                     BEQ     5$              ;PGF SHOULD BE 0
1273 012702  104044                     ERROR+  44              ;PGF NOT ZERO ON A STI WITH IMSK=1
1274 012704  004737  022234             JSR     PC,DMPREG
1275 012710                      5$:
```

```
1277                                    ;ISSUE A SANITY TIMER BOOT REQUEST WITH BIMSK BIT =1 AND CHECK:
1278                                    ;SBF BIT TO STAY 0
1279                                    ;UI BIT TO SET
1280
1281                                    ;;*****************************************************************
 (3)                                    ;*TEST 42       CHECK THE IMSK BIT FOR PROPER OPERATION ON ST BOOT
 (3)                                    ;;*****************************************************************
 (2) 012710  000004                     TST42:  SCOPE
1282 012712  004737  022204                     JSR     PC,SSTMNT       ;ENABLE MAINT. LOOP
1283 012716  012777  000000  010646             MOV     #PGTEE,@ACR     ;SELECT PGTE
1284 012724  016477  020536  010644             MOV     PGBTAB(4),@PGTE ;LOAD PBB
1285 012732  012777  000004  010632             MOV     #IMSKE,@ACR     ;SELECT IMSK
1286 012740  016477  020536  010630             MOV     BMITAB(4),@IMSK ;INHIBIT LINE
1287 012746  012777  000001  010616             MOV     #PGCSE,@ACR     ;SELECT PGCS
1288 012754  012777  000003  010614             MOV     #PTP+GO,@PGCS   ;TRANSMIT
1289 012762  012777  000001  010602  1$:        MOV     #PGCSE,@ACR     ;SELECT PGCS
1290 012770  032777  004000  010600             BIT     #PRDY,@PGCS     ;CHECK FOR
1291 012776  001771                             BEQ     1$              ;READY
1292 013000  012777  000006  010564             MOV     #STFE,@ACR      ;SELECT STF
1293 013006  017737  010564  001126             MOV     @STF,$BDDAT     ;READ STF
1294 013014  012737  007400  001124             MOV     #7400,$GDDAT    ;FETCH EXPECTED FLAGS
1295 013022  046437  020536  001124             BIC     PGBTAB(4),$GDDAT;MINUS EXPECTED FLAG
1296 013030  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
1297 013036  001403                             BEQ     2$              ;SKIP IF OK
1298 013040  104044                             ERROR+  44              ;IMSK BIT DID NOT INHIBIT ITS SBF BIT
1299 013042  004737  022234                     JSR     PC,DMPREG
1300 013046  016437  020536  001124  2$:        MOV     UVITAB(4),$GDDAT;LOAD EXPECTED UI BIT
1301 013054  012777  000010  010510             MOV     #EXCE,@ACR      ;SELECT EXC
1302 013062  017737  010510  001126             MOV     @EXC,$BDDAT     ;READ EXC
1303 013070  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK THE UI BIT
1304 013076  001403                             BEQ     3$      ;SKIP IF SET
1305 013100  104044                             ERROR+  44              ;UI DID NOT SET ON A SBF WITH IMSK=1
1306 013102  004737  022234                     JSR     PC,DMPREG
1307 013106  005037  001124  3$:        CLR     $GDDAT                  ;EXPECT PGF TO BE 0
1308 013112  012777  000005  010452             MOV     #PGFE,@ACR      ;SELECT PGF
1309 013120  017737  010452  001126             MOV     @PGF,$BDDAT     ;GET AND TEST PGF
1310 013126  001403                             BEQ     4$              ;IT SHOULD BE 0
1311 013130  104044                             ERROR+  44
1312 013132  004737  022234                     JSR     PC,DMPREG
1313 013136                             4$:
```

```
                                        ;TEST OPERATION OF THE TIMER LOGIC
                                        ;;*****************************************************************
                                        ;*TEST 43       CHECK THE TIMER INTERRUPT LOGIC AND COUNT REGISTER
                                        ;;*****************************************************************
(2) 013136 000004                       TST43:  SCOPE
1318 013140 004737 022204                       JSR     PC,SSTMNT       ;SET MAINT. LOOPBACK
1319 013144 005000                              CLR     R0              ;INIT. TIMER COUNTER
1320 013146 012777 000002 010416                MOV     #STTEE,@ACR     ;SELECT STTE
1321 013154 016477 020526 010414                MOV     SIFTAB(4),@STTE ;ENABLE SI LINE
1322 013162 012777 000001 010406                MOV     #ENB,@STCS      ;ENABLE SANITY TIMER
1323 013170 012777 000003 010374 1$:            MOV     #STCSE,@ACR     ;SELECT STCS
1324 013176 017737 010374 023610                MOV     @STCS,DISPLY    ;READ STCS
1325 013204 032737 177410 023610                BIT     #177410,DISPLY  ;CHECK FOR TMO OR COUNT=-1
1326 013212 001007                              BNE     2$              ;SKIP IF ANY BITS SET
1327 013214 005200                              INC     R0              ;COUNT BACKGROUND TIMER
1328 013216 001364                              BNE     1$              ;KEEP COUNTING UNTIL OVERFLOW
1329 013220 104044                              ERROR+  44              ;TIMER DID NOT COUNTDOWN WHEN ENABLED
1330 013222 004737 022234                       JSR     PC,DMPREG
1331 013226 012700 177777                       MOV     #-1,R0          ;SET TIME TO MAX
1332 013232 010037 023612               2$:     MOV     R0,TIMPER       ;SAVE TIME PERIOD CONSTANT
1333 013236 006200                              ASR     R0              ;DIVIDE BY
1334 013240 006200                              ASR     R0              ;8
1335 013242 006200                              ASR     R0
1336 013244 060037 023612                       ADD     R0,TIMPER       ;ADD 12%
1337 013250 012777 000003 010314                MOV     #STCSE,@ACR     ;;**
1338 013256 017737 010314 023610                MOV     @STCS,DISPLY    ;;**
1339 013264 005037 001126                       CLR     $BDDAT          ;INIT. DISPLAY
1340 013270 113737 023611 001126                MOVB    DISPLY+1,$BDDAT ;FETCH COUNT
1341 013276 012737 000377 001124                MOV     #377,$GDDAT     ;LOAD EXPECTED
1342 013304 023737 001124 001126                CMP     $GDDAT,$BDDAT   ;CHECK FOR TIMER = -1
1343 013312 001403                              BEQ     3$              ;SKIP IF OK
1344 013314 104044                              ERROR+  44              ;TIMER DID NOT COUNT FROM 0 TO -1
1345 013316 004737 022234                       JSR     PC,DMPREG
1346 013322 012737 000011 001124 3$:            MOV     #TMO+ENB,$GDDAT ;LOAD EXPECTED
1347 013330 113737 023610 001126                MOVB    DISPLY,$BDDAT   ;LOAD ACTUAL
1348 013336 023737 001124 001126                CMP     $GDDAT,$BDDAT   ;CHECK FOR TMO
1349 013344 001403                              BEQ     4$              ;SKIP IF SET
1350 013346 104044                              ERROR+  44              ;TMO DID NOT SET ON TIMER UNDERFLOW
1351 013350 004737 022234                       JSR     PC,DMPREG
1352 013354 012777 000001 010210 4$:            MOV     #PGCSE,@ACR     ;SELECT PGCS
1353 013362 017737 010210 023610                MOV     @PGCS,DISPLY    ;READ PGCS
1354 013370 032737 000010 023610                BIT     #IP,DISPLY      ;TEST FOR IP
1355 013376 001003                              BNE     5$              ;SKIP IF SET
1356 013400 104044                              ERROR+  44              ;SIF DID NOT SET IP
1357 013402 004737 022234                       JSR     PC,DMPREG
1358 013406                             5$:
```

```
1360                                    ;;*****************************************************************
 (3)                                    ;*TEST 44       CHECK THE TIMER BOOT LOGIC AND COUNT REGISTER
 (3)                                    ;;*****************************************************************
 (2) 013406 000004                      TST44:  SCOPE
1361 013410 004737 022204                       JSR     PC,SSTMNT       ;SET MAINT. LOOPBACK
1362 013414 012777 000002 010150                MOV     #STTEE,@ACR     ;SELECT STTE
1363 013422 016477 020536 010146                MOV     SBFTAB(4),@STTE ;ENABLE SB LINE
1364 013430 012777 000001 010140                MOV     #ENB,@STCS      ;ENABLE SANITY TIMER
1365 013436 005000                              CLR     R0              ;CLR COUNTER
1366 013440 012777 000003 010124 1$:    MOV     #STCSE,@ACR     ;SELECT STCS
1367 013446 017737 010124 023610                MOV     @STCS,DISPLY    ;READ STCS
1368 013454 032737 177410 023610                BIT     #177410,DISPLY  ;CHECK FOR TMO OR COUNT=-1
1369 013462 001005                              BNE     2$              ;SKIP IF ANY BITS SET
1370 013464 005200                              INC     R0              ;COUNT BACKGROUND TIMER
1371 013466 001364                              BNE     1$              ;KEEP COUNTING UNTIL OVERFLOW
1372 013470 104044                              ERROR+  44              ;TIMER DID NOT COUNTDOWN WHEN ENABLED
1373 013472 004737 022234                       JSR     PC,DMPREG
1374 013476 012777 000003 010066 2$:    MOV     #STCSE,@ACR     ;;**
1375 013504 017737 010066 023610                MOV     @STCS,DISPLY    ;;**
1376 013512 005037 001126                       CLR     $BDDAT          ;INIT. DISPLAY
1377 013516 113737 023611 001126                MOVB    DISPLY+1,$BDDAT ;FETCH COUNT
1378 013524 012737 000377 001124                MOV     #377,$GDDAT     ;LOAD EXPECTED
1379 013532 023737 001124 001126                CMP     $GDDAT,$BDDAT   ;CHECK FOR TIMER = -1
1380 013540 001403                              BEQ     3$              ;SKIP IF OK
1381 013542 104044                              ERROR+  44              ;TIMER DID NOT COUNT FROM 0 TO -1
1382 013544 004737 022234                       JSR     PC,DMPREG
1383 013550 012737 000011 001124 3$:    MOV     #TMO+ENB,$GDDAT ;LOAD EXPECTED
1384 013556 113737 023610 001126                MOVB    DISPLY,$BDDAT   ;LOAD ACTUAL
1385 013564 023737 001124 001126                CMP     $GDDAT,$BDDAT   ;CHECK FOR TMO
1386 013572 001403                              BEQ     4$              ;SKIP IF SET
1387 013574 104044                              ERROR+  44              ;TMO DID NOT SET ON TIMER UNDERFLOW
1388 013576 004737 022234                       JSR     PC,DMPREG
1389 013602 012777 000001 007762 4$:    MOV     #PGCSE,@ACR     ;SELECT PGCS
1390 013610 017737 007762 023610                MOV     @PGCS,DISPLY    ;READ PGCS
1391 013616 032737 000010 023610                BIT     #IP,DISPLY      ;TEST FOR IP
1392 013624 001403                              BEQ     5$              ;SKIP IF CLR
1393 013626 104044                              ERROR+  44              ;SBF SET IP
1394 013630 004737 022234                       JSR     PC,DMPREG
1395 013634                             5$:
```

```
1397                                    ;;*****************************************************************
 (3)                                    ;*TEST 45       CHECK THE COUNTER FOR PROPER DECREMENTATION
 (3)                                    ;;*****************************************************************
 (2) 013634  000004                     TST45:  SCOPE
1398 013636  012737  000001  001204              MOV     #1,$TMP3        ;INIT THE COUNT VALUE
1399 013644  005037  001126                       CLR     $BDDAT          ;ZERO THE HIGH BYTE OF $BDDAT
1400 013650  012737  013656  001110              MOV     #1$,$LPERR      ;SET SCOPE POINT
1401 013656  004737  022204              1$:     JSR     PC,SSTMNT       ;ENTER MAINT. LOOP MODE
1402 013662  012777  000002  007702              MOV     #STTEE,@ACR     ;SELECT STTE
1403 013670  016477  020526  007700              MOV     SIFTAB(4),@STTE ;SET LINE ENABLE
1404 013676  112777  000001  007672              MOVB    #ENB,@STCS      ;ENABLE COUNTING
1405 013704  005377  007662                       DEC     @ACR            ;SELECT STCS
1406 013710  113777  001204  007662              MOVB    $TMP3,@STCSH    ;LOAD COUNT AND START
1407 013716  013737  001204  001124              MOV     $TMP3,$GDDAT    ;GET STARTING COUNT VALUE
1408 013724  005337  001124                       DEC     $GDDAT          ;MINUS 1
1409 013730  013700  023612                       MOV     TIMPER,R0       ;GET BASIC TIME CONSTANT
1410 013734  006300                              ASL     R0              ;*2
1411 013736  005377  007630              2$:     DEC     @ACR            ;SELECT STCS
1412 013742  117737  007632  001126              MOVB    @STCSH,$BDDAT   ;READ COUNT VALUE
1413 013750  123737  001204  001126              CMPB    $TMP3,$BDDAT    ;CHECK FOR CHANGE
1414 013756  001006                              BNE     3$              ;SKIP IF CHANGED
1415 013760  005300                              DEC     R0              ;DEFLATE COUNTER
1416 013762  001365                              BNE     2$              ;CHECK AGAIN
1417 013764  104044                              ERROR+  44              ;TIMER DID NOT DECREMENT
1418 013766  004737  022234                       JSR     PC,DMPREG
1419 013772  000414                              BR      4$              ;SKIP NEXT CHECK
1420 013774  005377  007572              3$:     DEC     @ACR            ;SELECT STCS
1421 014000  117737  007574  001126              MOVB    @STCSH,$BDDAT   ;REREAD TIMER
1422 014006  023737  001124  001126              CMP     $GDDAT,$BDDAT   ;CHECK FOR CORRECT DECREMENTATION
1423 014014  001403                              BEQ     4$              ;SKIP IF OK
1424 014016  104044                              ERROR+  44              ;COUNT REGISTER DID NOT DECREMENT CORRECTLY
1425 014020  004737  022234                       JSR     PC,DMPREG
1426 014024  105237  001204              4$:     INCB    $TMP3           ;BUMP TEST VALUE
1427 014030  001312                              BNE     1$              ;DO 1 - 377
```

```
1429                                    ;ISSUE A PROGRAM-GENERATED REQUEST WITH INTERRUPT MASKS CLEAR WITH ALL COMBINATIONS OF
1430                                    ;TRANSMIT ENABLES, MAINTENANCE TYPE, AND MAINTENANCE FRAMING
1431                                    ;AND CHECK FOR:
1432                                    ;       CORRECT CONTENTS OF PGCS (RDY, IP, SELF-ID)
1433                                    ;       RTE BITS IN THE EXC REGISTER TO BE SET
1434                                    ;       ALL OTHER FLAGS TO BE CLEAR (PGF, STF, DCF, UIF)
1435                                    ;       SELECTED RTE FLAG CAN BE CLEARED
1436
1437            000005                          W=5
1438
1439                                    ;;*****************************************************************
 (3)                                    ;*TEST 46        CHECK OPERATION OF PARITY & FRAMING ERRORS, IM(3:0)=0
 (3)                                    ;;*****************************************************************
 (2)  014032  000004                    TST46:  SCOPE
1440  014034  004737  022042                    JSR     PC,SETMNT       ;ENABLE MAINTENANCE LOOPBACK
1441  014040  012737  014064  001110            MOV     #1$,$LPERR      ;FIX START OF LOOP
1442  014046  012777  000007  007516            MOV     #DCFE,@ACR
1443  014054  017737  007516  023616            MOV     @DCF,SNAP0      ;SAVE IMAGE OF DCF REGISTER
1444  014062  005000                            CLR     R0              ;INIT FRAME, TYPE, PGTE DATA SETUP
1445
1446  014064  010037  001200            1$:     MOV     R0,$TMP1
1447  014070  004737  022214                    JSR     PC,PARCAL       ;CALCULATE PARITY OF DATA
1448  014074  113737  001200  001201            MOVB    $TMP1,$TMP1+1
1449  014102  106037  001201                    RORB    $TMP1+1         ;POSITION DATA
1450  014106  106037  001201                    RORB    $TMP1+1
1451  014112  106037  001201                    RORB    $TMP1+1
1452  014116  106037  001201                    RORB    $TMP1+1
1453  014122  012777  000000  007442            MOV     #PGTEE,@ACR
1454  014130  013777  001200  007440            MOV     $TMP1,@PGTE     ;LOAD TRANSMIT ENABLES
1455  014136  006301                            ASL     R1              ;GET PARITY
1456  014140  042701  177775            2$:     BIC     #177775,R1
1457  014144  016437  020516  001124            MOV     SIDTAB(4),$GDDAT        ;GET SELF-ID
1458  014152  050137  001124                    BIS     R1,$GDDAT
1459  014156  052701  000001                    BIS     #1,R1
1460  014162  012777  000001  007402            MOV     #PGCSE,@ACR
1461  014170  010177  007402                    MOV     R1,@PGCS        ;LOAD PARITY, SET GO
1462  014174  005001                            CLR     R1
1463  014176  052737  004010  001124            BIS     #PRDY+IP,$GDDAT
1464  014204  105377  007362            3$:     DECB    @ACR
1465  014210  017737  007362  001126            MOV     @PGCS,$BDDAT
1466  014216  032737  004000  001126            BIT     #PRDY,$BDDAT    ;WAIT FOR READY
1467  014224  001005                            BNE     4$
1468  014226  005301                            DEC     R1              ;BUT DON'T WAIT TOO LONG
1469  014230  001365                            BNE     3$
1470  014232  104044                            ERROR+  44
1471  014234  004737  022234                    JSR     PC,DMPREG
1472  014240  023737  001126  001124  4$:       CMP     $BDDAT,$GDDAT   ;IS PGCS OK?
1473  014246  001403                            BEQ     5$
1474  014250  104044                            ERROR+  44
1475  014252  004737  022234                    JSR     PC,DMPREG
```

```
1477  014256  012737  000017  001124  5$:     MOV     #17,$GDDAT      ;GET EXPECTED RTE FLAGS
1478  014264  012777  000010  007300          MOV     #EXCE,@ACR
1479  014272  017737  007300  001126          MOV     @EXC,$BDDAT     ;GET EXC REGISTER
1480  014300  023737  001124  001126          CMP     $GDDAT,$BDDAT
1481  014306  001403                          BEQ     6$
1482  014310  104044                          ERROR+  44
1483  014312  004737  022234                  JSR     PC,DMPREG
1484  014316  105377  007250          6$:     DECB    @ACR
1485  014322  016437  020526  001124          MOV     PIFTAB(4),$GDDAT
1486  014330  013777  001124  007240          MOV     $GDDAT,@EXC     ;CLEAR FLAG
1487  014336  005137  001124                  COM     $GDDAT          ;MAKE FLAG DATA
1488  014342  042737  177760  001124          BIC     #177760,$GDDAT
1489  014350  105377  007216                  DECB    @ACR
1490  014354  017737  007216  001126          MOV     @EXC,$BDDAT
1491  014362  023737  001126  001124          CMP     $BDDAT,$GDDAT
1492  014370  001403                          BEQ     7$
1493  014372  104044                          ERROR+  44
1494  014374  004737  022234                  JSR     PC,DMPREG
1495  014400  005037  001124          7$:     CLR     $GDDAT
1496  014404  012777  000005  007160          MOV     #PGFE,@ACR
1497  014412  017737  007160  001126          MOV     @PGF,$BDDAT     ;PGF SHOULD BE 0
1498  014420  001403                          BEQ     8$
1499  014422  104044                          ERROR+  44
1500  014424  004737  022234                  JSR     PC,DMPREG
1501  014430  017737  007142  001126  8$:     MOV     @STF,$BDDAT     ;GET STF
1502  014436  001403                          BEQ     9$              ;STF SHOULD BE 0
1503  014440  104044                          ERROR+  44
1504  014442  004737  022234                  JSR     PC,DMPREG
1505  014446  013737  023616  001124  9$:     MOV     SNAP0,$GDDAT    ;GET EXPECTED DATA FOR DCF
1506  014454  017737  007116  001126          MOV     @DCF,$BDDAT     ;GET DCF REGISTER
1507  014462  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;CHECK THAT THERE IS NO CHANGE
1508  014470  001403                          BEQ     10$
1509  014472  104044                          ERROR+  44
1510  014474  004737  022234                  JSR     PC,DMPREG
1511  014500  005200          10$:    INC     R0              ;INCREMENT TEST DATA
1512  014502  012777  000015  007062          MOV     #MTCE,@ACR      ;POINT TO MTC
1513  014510  042777  006000  007060          BIC     #MTYP+MFRM,@MTC ;CLEAR TYPE AND FRAME ERROR
1514  014516  032700  000400                  BIT     #400,R0
1515  014522  001403                          BEQ     11$             ;IF R0(8)=1, SET MTYPE
1516  014524  052777  004000  007044          BIS     #MTYP,@MTC
1517  014532  032700  001000          11$:    BIT     #1000,R0
1518  014536  001403                          BEQ     12$
1519  014540  052777  002000  007030          BIS     #MFRM,@MTC      ;IF R0(9)=1, SET FRAMING ERROR
1520  014546  020027  002000          12$:    CMP     R0,#2000
1521  014552  103002                          BHIS    13$             ;LOOP UNTIL ALL COMBOS OF DATA, TYPE &
1522                                                                  ;FRAMING HAVE BEEN TESTED.
1523  014554  000137  014064                  JMP     1$
1524  014560                          13$:
```

```
                                        ;ISSUE A PROGRAM-GENERATED REQUEST WITH INTERRUPT MASKS SET WITH ALL COMBINATIONS OF
                                        ;TRANSMIT ENABLES, MAINTENANCE TYPE, AND MAINTENANCE FRAMING
                                        ;AND CHECK FOR:
                                        ;       CORRECT CONTENTS OF PGCS (RDY, SELF-ID)
                                        ;       RTE BITS IN THE EXC REGISTER TO BE CLEAR
                                        ;       ALL OTHER FLAGS TO BE CLEAR (PGF, STF, DCF, UIF)

                                        ;;*****************************************************************
                                        ;*TEST 47       CHECK OPERATION OF PARITY & FRAMING ERRORS, IM(3:0)=1
                                        ;;*****************************************************************
 (2) 014560  000004                     TST47:  SCOPE
1534 014562  004737  022042                     JSR     PC,SETMNT       ;ENABLE MAINTENANCE LOOPBACK
1535 014566  012737  014626  001110             MOV     #1$,$LPERR      ;FIX START OF LOOP
1536 014574  012777  000007  006770             MOV     #DCFE,@ACR
1537 014602  017737  006770  023616             MOV     @DCF,SNAP0      ;SAVE IMAGE OF DCF REGISTER
1538 014610  012777  000004  006754             MOV     #IMSKE,@ACR     ;POINT TO IMASK
1539 014616  012777  000017  006752             MOV     #17,@IMSK       ;SET ALL INTERRUPT MASKS
1540 014624  005000                             CLR     R0              ;INIT FRAME, TYPE, PGTE DATA SETUP
1541
1542 014626  010037  001200             1$:     MOV     R0,$TMP1
1543 014632  004737  022214                     JSR     PC,PARCAL       ;CALCULATE PARITY OF DATA
1544 014636  113737  001200  001201             MOVB    $TMP1,$TMP1+1
1545 014644  106037  001201                     RORB    $TMP1+1         ;POSITION DATA
1546 014650  106037  001201                     RORB    $TMP1+1
1547 014654  106037  001201                     RORB    $TMP1+1
1548 014660  106037  001201                     RORB    $TMP1+1
1549 014664  012777  000000  006700             MOV     #PGTEE,@ACR
1550 014672  013777  001200  006676             MOV     $TMP1,@PGTE     ;LOAD TRANSMIT ENABLES
1551 014700  006301                             ASL     R1              ;GET PARITY
1552 014702  042701  177775             2$:     BIC     #177775,R1
1553 014706  016437  020516  001124             MOV     SIDTAB(4),$GDDAT        ;GET SELF-ID '
1554 014714  050137  001124                     BIS     R1,$GDDAT
1555 014720  052701  000001                     BIS     #1,R1
1556 014724  012777  000001  006640             MOV     #PGCSE,@ACR
1557 014732  010177  006640                     MOV     R1,@PGCS        ;LOAD PARITY, SET GO
1558 014736  005001                             CLR     R1
1559 014740  052737  004000  001124             BIS     #PRDY,$GDDAT
1560 014746  105377  006620             3$:     DECB    @ACR
1561 014752  017737  006620  001126             MOV     @PGCS,$BDDAT
1562 014760  032737  004000  001126             BIT     #PRDY,$BDDAT    ;WAIT FOR READY
1563 014766  001005                             BNE     4$
1564 014770  005301                             DEC     R1              ;BUT DON'T WAIT TOO LONG
1565 014772  001365                             BNE     3$
1566 014774  104044                             ERROR+  44
1567 014776  004737  022234                     JSR     PC,DMPREG
1568 015002  023737  001126  001124     4$:     CMP     $BDDAT,$GDDAT   ;IS PGCS OK?
1569 015010  001403                             BEQ     5$
1570 015012  104044                             ERROR+  44
1571 015014  004737  022234                     JSR     PC,DMPREG
1572 015020  012737  000000  001124     5$:     MOV     #0,$GDDAT       ;GET EXPECTED RTE FLAGS
1573 015026  012777  000010  006536             MOV     #EXCE,@ACR
1574 015034  017737  006536  001126             MOV     @EXC,$BDDAT     ;GET EXC REGISTER
1575 015042  001403                             BEQ     6$
1576 015044  104044                             ERROR+  44
1577 015046  004737  022234                     JSR     PC,DMPREG
1578 015052                             6$:
```

```
1579  015052  005037  001124          7$:     CLR     $GDDAT
1580  015056  012777  000005  006506          MOV     #PGFE,@ACR
1581  015064  017737  006506  001126          MOV     @PGF,$BDDAT     ;PGF SHOULD BE 0
1582  015072  001403                          BEQ     8$
1583  015074  104044                          ERROR+  44
1584  015076  004737  022234                  JSR     PC,DMPREG
1585  015102  017737  006470  001126  8$:     MOV     @STF,$BDDAT     ;GET STF
1586  015110  001403                          BEQ     9$              ;STF SHOULD BE 0
1587  015112  104044                          ERROR+  44
1588  015114  004737  022234                  JSR     PC,DMPREG
1589  015120  013737  023616  001124  9$:     MOV     SNAP0,$GDDAT    ;GET EXPECTED DATA FOR DCF
1590  015126  017737  006444  001126          MOV     @DCF,$BDDAT     ;GET DCF REGISTER
1591  015134  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;CHECK THAT THERE IS NO CHANGE
1592  015142  001403                          BEQ     10$
1593  015144  104044                          ERROR+  44
1594  015146  004737  022234                  JSR     PC,DMPREG
1595  015152  005200                  10$:    INC     R0              ;INCREMENT TEST DATA
1596  015154  012777  000015  006410          MOV     #MTCE,@ACR      ;POINT TO MTC
1597  015162  042777  006000  006406          BIC     #MTYP+MFRM,@MTC ;CLEAR TYPE AND FRAME ERROR
1598  015170  032700  000400                  BIT     #400,R0
1599  015174  001403                          BEQ     11$             ;IF R0(8)=1, SET MTYPE
1600  015176  052777  004000  006372          BIS     #MTYP,@MTC
1601  015204  032700  001000          11$:    BIT     #1000,R0
1602  015210  001403                          BEQ     12$
1603  015212  052777  002000  006356          BIS     #MFRM,@MTC      ;IF R0(9)=1, SET FRAMING ERROR
1604  015220  020027  002000          12$:    CMP     R0,#2000
1605  015224  103600                          BLO     1$              ;LOOP UNTIL ALL COMBOS OF DATA, TYPE &
1606                                                                  ;FRAMING HAVE BEEN TESTED.
```

```
                                        ;TEST ALL THE PGCS ERROR BITS FOR PROPER OPERATION
                                        ;
                                        ;;*****************************************************************
                                        ;*TEST 50       TEST GRJ FOR PROPER OPERATION
                                        ;;*****************************************************************
                                        TST50:  SCOPE
                                                JSR     PC,SETMNT       ;CLR IIST AND ENTER MAINT. MODE
                                                MOV     ACR,R1          ;R1=ACR
                                                MOV     ADR,R2          ;R2=ADR
                                                MOV     #PGTEE,@R1      ;SELECT PGTE
                                                MOV     PGITAB(4),@R2   ;LOAD PGTE WITH ENABLE
                                                MOV     #GO,@R2         ;TRANSMIT
                                                DEC     @R1             ;SELECT PGCS
                                                INC     @R2             ;SET GO AGAIN
                                                MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED
                                                ADD     #ERR+GRJ+PRDY+IP,$GDDAT
                                        1$:     MOV     #PGCSE,@R1      ;SELECT PGCS
                                                MOV     @R2,$BDDAT      ;READ PGCS
                                                BIT     #PRDY,$BDDAT    ;TEST FOR RDY
                                                BEQ     1$              ;LOOP UNTIL SET
                                                CMP     $GDDAT,$BDDAT   ;CHECK STATUS
                                                BEQ     2$              ;SKIP IF OK
                                                ERROR+  44              ;ERR,GRJ, OR IP DID NOT SET ON
                                                JSR     PC,DMPREG
                                                                        ;A GRJ ERROR
                                        2$:     MOV     #PGFE,@R1       ;SELECT PGF
                                                MOV     #17,@R2         ;CLEAR PGF
                                                MOV     @IIIP,-(SP)     ;SAVE OLD
                                                MOV     @IIIV,-(SP)     ;PI DATA
                                                MOV     #3$,@IIIV       ;LOAD NEW
                                                MOV     #340,@IIIP      ;PI DATA
                                                MOV     #PGCSE,@R1      ;SELECT PGCS
                                                MOV     #IE,@R2         ;ENABLE PI
                                                CLR     PS              ;ALLOW PI
                                                NOP                     ;STALL
                                                MOV     #340,PS         ;LOCKOUT PI
                                                ERROR+  44              ;NO PI ON ERR WITH IE=1
                                                JSR     PC,DMPREG
                                                BR      4$              ;SKIP
                                        3$:     POPPOP                  ;FIX STACK
                                        4$:     MOV     (SP)+,@IIIV     ;RESTORE OLD
                                                MOV     (SP)+,@IIIP     ;PI INFO
                                                DEC     @R1             ;SELECT PGCS
                                                MOV     #ERR,@R2        ;CLEAR ERR BY WRITING 1 TO IT
                                                BIC     #ERR+GRJ+IP,$GDDAT
                                                DEC     @R1             ;SELECT PGCS
                                                MOV     @R2,$BDDAT      ;READ PGCS
                                                CMP     $GDDAT,$BDDAT   ;CHECK STATUS
                                                BEQ     5$              ;SKIP IF OK
                                                ERROR+  44              ;WRITING A ONE TO ERR DID NOT
                                                JSR     PC,DMPREG
                                                                        ;CLEAR ERR,GRJ, AND IP
                                        5$:
```

```
                                ;;*****************************************************************
                                ;*TEST 51        TEST PG RMR FOR PROPER OPERATION
                                ;;*****************************************************************
                                TST51:  SCOPE
                                        JSR     PC,SETMNT       ;CLR IIST AND ENTER MAINT. MODE
                                        MOV     ACR,R1          ;R1=ACR
                                        MOV     ADR,R2          ;R2=ADR
                                        MOV     #PGTEE,@R1      ;SELECT PGTE
                                        MOV     PGITAB(4),@R2   ;LOAD PGTE WITH ENABLE
                                        MOV     #GO,@R2         ;TRANSMIT
                                        MOV     #PGTEE,@R1      ;SELECT PGTE
                                        INC     @R2             ;ACCESS PGTE AGAIN
                                        MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED
                                        ADD     #ERR+PGMR+PRDY+IP,$GDDAT
                                1$:     MOV     #PGCSE,@R1      ;SELECT PGCS
                                        MOV     @R2,$BDDAT      ;READ PGCS
                                        BIT     #PRDY,$BDDAT    ;TEST FOR RDY
                                        BEQ     1$              ;LOOP UNTIL SET
                                        CMP     $GDDAT,$BDDAT   ;CHECK STATUS
                                        BEQ     2$              ;SKIP IF OK
                                        ERROR+  44              ;ERR,PGMR, OR IP DID NOT SET ON
                                        JSR     PC,DMPREG
                                                                ;A PGMR ERROR
                                2$:     MOV     #PGFE,@R1       ;SELECT PGF
                                        MOV     #17,@R2         ;CLEAR PGF
                                        MOV     #PGCSE,@R1      ;SELECT PGCS
                                        MOV     #ERR,@R2        ;CLEAR ERR BY WRITING 1 TO IT
                                        BIC     #ERR+PGMR+IP,$GDDAT
                                        MOV     #PGCSE,@R1      ;SELECT PGCS
                                        MOV     @R2,$BDDAT      ;READ PGCS
                                        CMP     $GDDAT,$BDDAT   ;CHECK STATUS
                                        BEQ     5$              ;SKIP IF OK
                                        ERROR+  44              ;WRITING A ONE TO ERR DID NOT
                                        JSR     PC,DMPREG
                                                                ;CLEAR ERR,PGMR, AND IP
                                5$:
```

```
 1693                                   ;;*****************************************************************
  (3)                                   ;*TEST 52       TEST ST RMR FOR PROPER OPERATION
  (3)                                   ;;*****************************************************************
  (2)  015666  000004                   TST52:  SCOPE
 1694  015670  004737  022204                    JSR     PC,SSTMNT       ;CLR IIST AND ENTER MAINT. MODE
 1695  015674  013701  023572                    MOV     ACR,R1          ;R1=ACR
 1696  015700  013702  023576                    MOV     ADR,R2          ;R2=ADR
 1697  015704  012711  000002                    MOV     #STTEE,@R1      ;SELECT STTE
 1698  015710  016412  020526                    MOV     STITAB(4),@R2   ;LOAD STTE WITH ENABLE
 1699  015714  012712  000001                    MOV     #ENB,@R2        ;START TIMER
 1700  015720  016437  020516  001124            MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED
 1701  015726  062737  114010  001124            ADD     #ERR+STMR+PRDY+IP,$GDDAT
 1702  015734  012711  000003              1$:   MOV     #STCSE,@R1      ;SELECT STCS
 1703  015740  032712  000010                    BIT     #TMO,@R2        ;TEST FOR TMO
 1704  015744  001773                            BEQ     1$              ;LOOP UNTIL SET
 1705  015746  012711  000002                    MOV     #STTEE,@R1      ;SELECT STTE
 1706  015752  005212                            INC     @R2             ;ACCESS STTE AGAIN
 1707  015754  012711  000001                    MOV     #PGCSE,@R1      ;SELECT PGCS
 1708  015760  011237  001126                    MOV     @R2,$BDDAT      ;READ PGCS
 1709  015764  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CHECK STATUS
 1710  015772  001403                            BEQ     2$              ;SKIP IF OK
 1711  015774  104044                            ERROR+  44              ;ERR,STMR, OR IP DID NOT SET ON
 1712  015776  004737  022234                    JSR     PC,DMPREG
 1713                                                                    ;A STMR ERROR
 1714  016002  012746  100000              2$:   MOV     #100000,-(SP)   ;PUSH VALUE
 1715  016006  005216                      4$:   INC     (SP)            ;STALL
 1716  016010  001376                            BNE     4$
 1717  016012  005026                            CLR     (SP)+
 1718  016014  012711  000006                    MOV     #STFE,@R1       ;SELECT STF
 1719  016020  012712  000017                    MOV     #17,@R2         ;CLEAR STF
 1720  016024  012711  000001                    MOV     #PGCSE,@R1      ;SELECT PGCS
 1721  016030  012712  100000                    MOV     #ERR,@R2        ;CLEAR ERR BY WRITING 1 TO IT
 1722  016034  042737  110010  001124            BIC     #ERR+STMR+IP,$GDDAT
 1723  016042  012711  000001                    MOV     #PGCSE,@R1      ;SELECT PGCS
 1724  016046  011237  001126                    MOV     @R2,$BDDAT      ;READ PGCS
 1725  016052  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CHECK STATUS
 1726  016060  001403                            BEQ     5$              ;SKIP IF OK
 1727  016062  104044                            ERROR+  44              ;WRITING A ONE TO ERR DID NOT
 1728  016064  004737  022234                    JSR     PC,DMPREG
 1729                                                                    ;CLEAR ERR,STMR, AND IP
 1730  016070                          5$:
 1731  016070  000207                            RTS     PC              ;;EXIT
```

```
                                        ;ISSUE A PROGRAM-GENERATED REQUEST WITH INTERRUPT MASKS CLEAR WITH ALL COMBINATIONS OF
                                        ;TRANSMIT ENABLES AND CHECK FOR:
                                        ;       CORRECT CONTENTS OF PGCS (RDY, IP, SELF-ID)
                                        ;       RTE BITS IN THE EXC REGISTER TO BE CLEAR
                                        ;       ALL OTHER FLAGS TO BE CLEAR ( STF, DCF, UIF)

016072                                  PGONL:
                                        ;;*****************************************************************
                                        ;*TEST 53       CHECK OPERATION OF PG REQUESTS IN NORMAL MODE
                                        ;;*****************************************************************
016072  000004                          TST53:  SCOPE
016074  012777  100000  005470                  MOV     #CLR,@ACR       ;CLEAR THE IIST
016102  005737  023660                          TST     CONFIG          ;ARE WE PREVENTED FROM ONLINE? (CONFIG=0)
016106  001002                          BNE     .+6                     ;SKIP IF NOT
016110  000137  016716                  JMP     10$                     ;IF YES, GO TO NEXT TEST.
016114  005000                          CLR     R0                      ;STALL
016116  012777  100000  005446  55$:    MOV     #CLR,@ACR
016124  105200                          INCB    R0
016126  001373                          BNE     55$
016130  012737  016154  001110          MOV     #1$,$LPERR              ;FIX START OF LOOP
016136  012777  000007  005426          MOV     #DCFE,@ACR
016144  017737  005426  023616          MOV     @DCF,SNAP0              ;SAVE IMAGE OF DCF REGISTER
016152  005000                          CLR     R0                      ;INIT PGTE DATA SETUP
016154  012777  100000  005410  1$:     MOV     #CLR,@ACR               ;CLEAR THE IIST
016162  012777  000005  005402          MOV     #PGFE,@ACR              ;SELECT PGF
016170  012777  007400  005400          MOV     #7400,@PGF              ;CLEAR ALL PGF FLAGS
016176  012777  007400  005372          MOV     #7400,@STF              ;CLEAR ALL STF FLAGS
016204  012777  000017  005364          MOV     #17,@DCF                ;CLR BRK
016212  012777  000015  005352          MOV     #MTCE,@ACR              ;SELECT MTC
016220  012777  000010  005350          MOV     #DSBT,@MTC              ;DISABLE BOOTS
016226  010037  001200                  MOV     R0,$TMP1
016232  023727  023660  000003          CMP     CONFIG,#3               ;SHOULD WE PREVENT XMIT OF ALL CODES?
016240  001013                          BNE     11$                     ;NO IF NOT CONFIG #3.
016242  016401  020526                  MOV     PGITAB(4),R1            ;YES -- GET MASK FOR SELF.
016246  010146                          MOV     R1,-(SP)                ;SAVE INTR ENB
016250  006301                          ASL     R1                      ;SHIFT 4X TO BOOT ENB
016252  006301                          ASL     R1
016254  006301                          ASL     R1
016256  006301                          ASL     R1
016260  052601                          BIS     (SP)+,R1                ;GET INTR ENB MASK
016262  005101                          COM     R1
016264  040137  001200                  BIC     R1,$TMP1                ;CLEAR BITS FOR DESTINATION OTHER THAN SELF.
016270                          11$:
016270  010046                          MOV     R0,-(SP)                ;SAVE COUNT VALUE
016272  013700  001200                  MOV     $TMP1,R0                ;GET ENABLE FOR PARITY GENERATOR
016276  004737  022214                  JSR     PC,PARCAL               ;CALCULATE PARITY OF DATA
016302  012600                          MOV     (SP)+,R0                ;RESTORE COUNT VALUE.
016304  113737  001200  001201          MOVB    $TMP1,$TMP1+1
016312  106037  001201                  RORB    $TMP1+1                 ;POSITION DATA
016316  106037  001201                  RORB    $TMP1+1
016322  106037  001201                  RORB    $TMP1+1
016326  106037  001201                  RORB    $TMP1+1
016332  012777  000000  005232          MOV     #PGTEE,@ACR
016340  013777  001200  005230          MOV     $TMP1,@PGTE             ;LOAD TRANSMIT ENABLES
016346  006301                          ASL     R1                      ;GET PARITY
016350  005101                          COM     R1                      ;FIX PAR BIT
```

```
1786  016352  042701  177775          2$:     BIC     #177775,R1
1787  016356  016437  020516  001124          MOV     SIDTAB(4),$GDDAT        ;GET SELF-ID
1788  016364  050137  001124                  BIS     R1,$GDDAT
1789  016370  052701  000001                  BIS     #1,R1
1790  016374  012777  000001  005170          MOV     #PGCSE,@ACR
1791  016402  010177  005170                  MOV     R1,@PGCS        ;LOAD PARITY, SET GO
1792  016406  005001                          CLR     R1
1793  016410  052737  004000  001124          BIS     #PRDY,$GDDAT
1794  016416  036437  020526  001200          BIT     PGITAB(4),$TMP1 ;TEST FOR PGI ENABLE
1795  016424  001403                          BEQ     3$              ;SKIP IF CLEAR
1796  016426  052737  000010  001124          BIS     #IP,$GDDAT      ;SET IP IF RELEVANT
1797  016434  105377  005132          3$:     DECB    @ACR
1798  016440  017737  005132  001126          MOV     @PGCS,$BDDAT
1799  016446  032737  004000  001126          BIT     #PRDY,$BDDAT    ;WAIT FOR READY
1800  016454  001005                          BNE     4$
1801  016456  005301                          DEC     R1              ;BUT DON'T WAIT TOO LONG
1802  016460  001365                          BNE     3$
1803  016462  104044                          ERROR+  44              ;PRDY DID NOT SET ON A PG XFER
1804  016464  004737  022234                  JSR     PC,DMPREG
1805  016470  023737  001126  001124  4$:     CMP     $BDDAT,$GDDAT   ;IS PGCS OK?
1806  016476  001403                          BEQ     5$
1807  016500  104044                          ERROR+  44              ;PGCS STATUS ERROR ON PG XFER
1808  016502  004737  022234                  JSR     PC,DMPREG
1809  016506  005037  001124          5$:     CLR     $GDDAT          ;SET EXPECTED RTE FLAGS
1810  016512  012777  000010  005052          MOV     #EXCE,@ACR
1811  016520  017737  005052  001126          MOV     @EXC,$BDDAT     ;GET EXC REGISTER
1812  016526  023737  001124  001126          CMP     $GDDAT,$BDDAT
1813  016534  001403                          BEQ     6$
1814  016536  104044                          ERROR+  44              ;EXC NOT 0 ON A PG XFER
1815  016540  004737  022234                  JSR     PC,DMPREG
1816  016544  036437  020526  001200  6$:     BIT     PGITAB(4),$TMP1 ;CHECK FOR LEGAL PGF
1817  016552  001403                          BEQ     61$             ;SKIP IF NO
1818  016554  056437  020526  001124          BIS     PGITAB(4),$GDDAT;LOAD PGF
1819  016562  036437  020536  001200  61$:    BIT     PGBTAB(4),$TMP1 ;CHECK FOR LEGAL PGB
1820  016570  001403                          BEQ     62$             ;SKIP IF NO
1821  016572  056437  020536  001124          BIS     PGBTAB(4),$GDDAT;LOAD PGB
1822  016600  012777  000005  004764  62$:    MOV     #PGFE,@ACR
1823  016606  017737  004764  001126          MOV     @PGF,$BDDAT     ;READ PGF
1824  016614  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK PGF
1825  016622  001403                          BEQ     7$
1826  016624  104044                          ERROR+  44              ;PGF ERROR ON PG XFER
1827  016626  004737  022234                  JSR     PC,DMPREG
1828  016632  005037  001124          7$:     CLR     $GDDAT          ;EXPECTED = 0
1829  016636  017737  004734  001126          MOV     @STF,$BDDAT     ;GET STF
1830  016644  001403                          BEQ     8$              ;STF SHOULD BE 0
1831  016646  104044                          ERROR+  44              ;STF NO 0 ON A PG XFER
1832  016650  004737  022234                  JSR     PC,DMPREG
1833  016654  013737  023616  001124  8$:     MOV     SNAP0,$GDDAT    ;GET EXPECTED DATA FOR DCF
1834  016662  017737  004710  001126          MOV     @DCF,$BDDAT     ;GET DCF REGISTER
1835  016670  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;CHECK THAT THERE IS NO CHANGE
1836  016676  001403                          BEQ     9$
1837  016700  104044                          ERROR+  44              ;DCF ERROR ON A PG XFER
1838  016702  004737  022234                  JSR     PC,DMPREG
1839  016706  105200                  9$:     INCB    R0              ;INCREMENT TEST DATA
1840  016710  001402                          BEQ     10$             ;SKIP WHEN DONE
1841  016712  000137  016154                  JMP     1$
```

```
1842  016716                    10$:
```

```
                                        ;FORCE A BREAK USING THE MAINTENENCE REGISTER AND TEST
                                        ;THE DCF REGISTER
                                        ;
                                        ;;*******************************************************************
 (3)                                    ;*TEST 54       TEST THE DCF REGISTER
 (3)                                    ;;*******************************************************************
 (2) 016716  000004                     TST54:  SCOPE
1848 016720  012777  100000  004644             MOV     #CLR,@ACR       ;INIT THE IIST
1849 016726  012737  000010  001200             MOV     #DSBT,$TMP1     ;SET VALUE FOR MTC TO CLEAR BREAK.
1850 016734  012737  000000  001202             MOV     #0,$TMP2        ;SET VALUE FOR CLEARING UNWANTED BITS.
1851 016742  005737  023660                     TST     CONFIG          ;SEE WHAT CONFIGURATION.
1852 016746  001005                             BNE     11$             ;IF NOT MAINT., SKIP ON.
1853 016750  004737  022042                     JSR     PC,SETMNT       ;IF MAINT. ONLY, SET MAINT. LOOPBACK.
1854 016754  012737  000012  001200             MOV     #MLEN+DSBT,$TMP1        ;NEW VALUE FOR CLEARING BRK.
1855 016762  016401  020536             11$:    MOV     BRKTAB(4),R1    ;GET BRK BIT FOR OWN LINE.
1856 016766  056401  020526                     BIS     DCFTAB(4),R1    ;AND DCF BIT.
1857 016772  005101                             COM     R1              ;MAKE 'BIC'' MASK.
1858 016774  042701  170360                     BIC     #170360,R1
1859 017000  010137  001202                     MOV     R1,$TMP2        ;STORE THE VALUE.
1860 017004  012777  000007  004560             MOV     #DCFE,@ACR      ;SELECT DCF
1861 017012  017737  004560  023616             MOV     @DCF,SNAP0      ;SAVE THE DCF CONTENTS
1862 017020  013737  023616  001124             MOV     SNAP0,$GDDAT    ;COPY DCF
1863 017026  023727  023660  000001             CMP     CONFIG,#1       ;ANY OTHER INTERFACES?
1864 017034  001003                             BNE     12$
1865 017036  012737  007400  001124             MOV     #7400,$GDDAT    ;SAY EXPECTED HAS OTHER BRK BITS SET.
1866 017044  005737  023660             12$:    TST     CONFIG          ;IN MAINT. MODE?
1867 017050  001002                             BNE     13$
1868 017052  005037  001124                     CLR     $GDDAT          ;YES -- ALL DCF BITS CLEAR.
1869 017056  046437  020536  001124     13$:    BIC     BRKTAB(4),$GDDAT;REMOVE SID BRK
1870 017064  013737  023616  001126             MOV     SNAP0,$BDDAT    ;READ ACTUAL
1871 017072  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK FOR LEGAL DCF
1872 017100  001403                             BEQ     1$              ;SKIP IF OK
1873 017102  104044                             ERROR+  44              ;INCORRECT DCF FOR SELECTED IIST
1874 017104  004737  022234                     JSR     PC,DMPREG
1875 017110  012777  000015  004454     1$:     MOV     #MTCE,@ACR      ;SELECT MTC
1876 017116  012777  000011  004452             MOV     #DSBT+MDSD,@MTC ;FORCE A BREAK
1877 017124  012777  000007  004440             MOV     #DCFE,@ACR      ;SELECT DCF
1878 017132  017737  004440  001126             MOV     @DCF,$BDDAT     ;READ DCF
1879 017140  043737  001202  001126             BIC     $TMP2,$BDDAT    ;CLEAR BITS NOT OUR OWN.
1880 017146  016437  020536  001124             MOV     BRKTAB(4),$GDDAT;SET RELEVANT BRK BIT
1881 017154  056437  020526  001124             BIS     DCFTAB(4),$GDDAT;SET RELEVANT DCF BIT
1882 017162  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK DCF
1883 017170  001403                             BEQ     2$              ;SKIP IF OK
1884 017172  104044                             ERROR+  44              ;INCORRECT DCF AFTER FORCING A BREAK
1885 017174  004737  022234                     JSR     PC,DMPREG
1886 017200  012777  000015  004364     2$:     MOV     #MTCE,@ACR      ;SELECT MTC
1887 017206  013777  001200  004362             MOV     $TMP1,@MTC              ;CLEAR BRK BIT
1888 017214  012777  000007  004350             MOV     #DCFE,@ACR      ;SELECT DCF
1889 017222  017737  004350  001126             MOV     @DCF,$BDDAT     ;READ DCF
1890 017230  043737  001202  001126             BIC     $TMP2,$BDDAT
1891 017236  046437  020536  001124             BIC     BRKTAB(4),$GDDAT;CLEAR OUT BRK BIT
1892 017244  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK DCF
1893 017252  001403                             BEQ     3$              ;SKIP IF BRK=0
1894 017254  104044                             ERROR+  44              ;INCORRECT DCF AFTER REMOVING THE BRK CONDITION
1895 017256  004737  022234                     JSR     PC,DMPREG
1896 017262  016437  020516  001124     3$:     MOV     SIDTAB(4),$GDDAT;LOAD SELF ID #
```

```
1897  017270  052737  004010  001124          BIS     #IP+PRDY,$GDDAT ;ADD OTHER EXPECTED DATA
1898  017276  012777  000007  004266          MOV     #DCFE,@ACR      ;SELECT DCF
1899  017304  013777  001202  004264          MOV     $TMP2,@DCF      ;CLEAR UNWANTED FLAGS.
1900  017312  012777  000001  004252          MOV     #PGCSE,@ACR     ;SELECT PGCS
1901  017320  017737  004252  001126          MOV     @PGCS,$BDDAT    ;READ PGCS
1902  017326  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK PGCS FOR IP
1903  017334  001403                          BEQ     4$              ;SKIP IF OK
1904  017336  104044                          ERROR+  44              ;DCF FLAG DID NOT SET IP
1905  017340  004737  022234                  JSR     PC,DMPREG
1906  017344  017746  004232          4$:     MOV     @IIIV,-(SP)     ;SAVE OLD PI
1907  017350  017746  004230                  MOV     @IIIP,-(SP)     ;DATA
1908  017354  012777  017430  004220          MOV     #5$,@IIIV       ;LOAD NEW
1909  017362  012777  000340  004214          MOV     #340,@IIIP      ;PI DATA
1910  017370  012777  000001  004174          MOV     #PGCSE,@ACR     ;SELECT PGCS
1911  017376  012777  000004  004172          MOV     #IE,@PGCS       ;ENABLE PI
1912  017404  005037  177776                  CLR     PS              ;ALLOW PI
1913  017410  000240                          NOP                     ;STALL
1914  017412  012737  000340  177776          MOV     #340,PS         ;LOCKOUT PI
1915  017420  104044                          ERROR+  44              ;NO PI ON A DCF FLAG
1916  017422  004737  022234                  JSR     PC,DMPREG
1917  017426  000401                          BR      6$              ;SKIP
1918  017430  022626                  5$:     POPPOP                  ;FIX STACK
1919  017432  012677  004146          6$:     MOV     (SP)+,@IIIP     ;RESTORE OLD
1920  017436  012677  004140                  MOV     (SP)+,@IIIV     ;PI DATA
1921  017442  012737  000000  001124          MOV     #0,$GDDAT       ;RELOAD EXPECTED DCF VALUE
1922  017450  012777  000007  004114          MOV     #DCFE,@ACR      ;SELECT DCF
1923  017456  016477  020526  004112          MOV     DCFTAB(4),@DCF  ;CLEAR DCF FLAG
1924  017464  005377  004102                  DEC     @ACR            ;RESELECT DCF
1925  017470  017737  004102  001126          MOV     @DCF,$BDDAT     ;READ DCF
1926  017476  043737  001202  001126          BIC     $TMP2,$BDDAT    ;MASK OUT JUNK.
1927  017504  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK DCF
1928  017512  001403                          BEQ     7$              ;SKIP IF DCF-0
1929  017514  104044                          ERROR+  44              ;WRITING A 1 TO DCF FLAG DID NOT CLEAR IT
1930  017516  004737  022234                  JSR     PC,DMPREG
1931  017522  012777  100000  004042  7$:     MOV     #CLR,@ACR       ;INIT. THE INTERFACE.
1932  017530  012777  000004  004034          MOV     #IMSKE,@ACR     ;SELECT THE IMSK REGISTER.
1933  017536  012777  000017  004032          MOV     #17,@IMSK       ;SET ALL INTERRUPT MASKS.
1934  017544  012777  000015  004020          MOV     #MTCE,@ACR      ;SELECT MTC.
1935  017552  012777  000012  004016          MOV     #DSBT+MLEN,@MTC ;DO MAINT. LOOP TO CLEAR BREAKS.
1936  017560  027777  004006  004004          CMP     @ACR,@ACR       ;DELAY
1937  017566  027777  004000  003776          CMP     @ACR,@ACR       ;  ''
1938  017574  027777  003772  003770          CMP     @ACR,@ACR
1939  017602  012777  000011  003766          MOV     #DSBT+MDSD,@MTC ;CAUSE BREAK
1940  017610  027777  003756  003754          CMP     @ACR,@ACR
1941  017616  027777  003750  003746          CMP     @ACR,@ACR
1942  017624  012737  000000  001124          MOV     #0,$GDDAT       ;SAY GOOD IS 0
1943  017632  012777  000007  003732          MOV     #DCFE,@ACR      ;SELECT DCF
1944  017640  017737  003732  001126          MOV     @DCF,$BDDAT     ;GET DCF
1945  017646  042737  007400  001126          BIC     #7400,$BDDAT    ;CLEAR BREAKS
1946  017654  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;VERIFY
1947  017662  001403                          BEQ     8$
1948  017664  104044                          ERROR+  44
1949  017666  004737  022234                  JSR     PC,DMPREG
1950  017672  000207                  8$:     RTS     PC              ;EXIT
```

```
                                ;ISSUE A SANITY TIMER REQUEST WITH INTERRUPT MASKS CLEAR WITH ALL COMBINATIONS OF
                                ;TRANSMIT ENABLES AND CHECK FOR:
                                ;       CORRECT CONTENTS OF PGCS (RDY, IP, SELF-ID)
                                ;       RTE BITS IN THE EXC REGISTER TO BE CLEAR
                                ;       ALL OTHER FLAGS TO BE CLEAR (PGF, DCF, UIF)

017674                          STONL:
                                ;;*****************************************************************
                                ;*TEST 55        CHECK OPERATION OF ST REQUESTS IN NORMAL MODE
                                ;;*****************************************************************
017674  000004                  TST55:  SCOPE
017676  012777  100000  003666          MOV     #CLR,@ACR       ;CLEAR THE IIST
017704  005737  023660                  TST     CONFIG
017710  001001                          BNE     .+4
017712  000207                          RTS     PC              ;EXIT IF WE CAN'T XMIT
017714  012737  017740  001110          MOV     #1$,$LPERR      ;FIX START OF LOOP
017722  012777  000007  003642          MOV     #DCFE,@ACR
017730  017737  003642  023616          MOV     @DCF,SNAP0      ;SAVE IMAGE OF DCF REGISTER
017736  005000                          CLR     R0              ;INIT STTE DATA SETUP
017740  012777  100000  003624  1$:     MOV     #CLR,@ACR       ;CLEAR THE IIST
017746  012777  000005  003616          MOV     #PGFE,@ACR      ;SELECT PGF
017754  012777  007400  003614          MOV     #7400,@PGF      ;CLEAR ALL
017762  012777  007400  003606          MOV     #7400,@STF      ;FLAGS
017770  012777  000017  003600          MOV     #17,@DCF        ;CLEAR BRK
017776  012777  000015  003566          MOV     #MTCE,@ACR      ;SELECT MTC
020004  012777  000010  003564          MOV     #DSBT,@MTC      ;DISABLE BOOTS
020012  010037  001200                  MOV     R0,$TMP1
020016  023727  023660  000003          CMP     CONFIG,#3       ;SHOULD WE PREVENT XMIT OF ALL CODES?
020024  001013                          BNE     11$             ;NO IF NOT CONFIG #3.
020026  016401  020526                  MOV     STITAB(4),R1    ;YES -- GET MASK FOR SELF.
020032  010146                          MOV     R1,-(SP)        ;SAVE INTR ENB
020034  006301                          ASL     R1              ;SHIFT 4X TO BOOT ENB
020036  006301                          ASL     R1
020040  006301                          ASL     R1
020042  006301                          ASL     R1
020044  052601                          BIS     (SP)+,R1        ;GET INTR ENB MASK
020046  005101                          COM     R1
020050  040137  001200                  BIC     R1,$TMP1        ;CLEAR BITS FOR DESTINATION OTHER THAN SELF.
020054                          11$:
020054  010046                          MOV     R0,-(SP)        ;SAVE COUNT VALUE
020056  013700  001200                  MOV     $TMP1,R0        ;GET ENABLE FOR PARITY GENERATOR
020062  004737  022214                  JSR     PC,PARCAL       ;CALCULATE PARITY OF DATA
020066  012600                          MOV     (SP)+,R0        ;RESTORE COUNT VALUE.
020070  113737  001200  001201          MOVB    $TMP1,$TMP1+1
020076  106037  001201                  RORB    $TMP1+1         ;POSITION DATA
020102  106037  001201                  RORB    $TMP1+1
020106  106037  001201                  RORB    $TMP1+1
020112  106037  001201                  RORB    $TMP1+1
020116  012777  000002  003446          MOV     #STTEE,@ACR
020124  013777  001200  003444          MOV     $TMP1,@STTE     ;LOAD TRANSMIT ENABLES
020132  006301                          ASL     R1              ;GET PARITY
020134  005101                          COM     R1              ;FIX PAR BIT
020136  042701  177775          2$:     BIC     #177775,R1
020142  010137  001124                  MOV     R1,$GDDAT       ;LOAD PARITY
020146  052701  000001                  BIS     #1,R1
020152  012777  000003  003412          MOV     #STCSE,@ACR
```

```
2005  020160  010177  003412                    MOV     R1,@STCS        ;LOAD PARITY, ENB, AND COUNT
2006  020164  005001                            CLR     R1
2007  020166  052737  177411  001124            BIS     #TMO+177401,$GDDAT
2008  020174  012777  000004  003370            MOV     #IMSKE,@ACR     ;SELECT IMSK
2009  020202  105377  003364          3$:       DECB    @ACR
2010  020206  017737  003364  001126            MOV     @STCS,$BDDAT
2011  020214  032737  000010  001126            BIT     #TMO,$BDDAT     ;WAIT FOR TMO
2012  020222  001005                            BNE     4$
2013  020224  005301                            DEC     R1              ;BUT DON'T WAIT TOO LONG
2014  020226  001365                            BNE     3$
2015  020230  104044                            ERROR+  44              ;TMO DID NOT SET ON A ST XFER
2016  020232  004737  022234                    JSR     PC,DMPREG
2017  020236  012777  000003  003326  4$:       MOV     #STCSE,@ACR     ;SELECT STCS
2018  020244  017737  003326  001126            MOV     @STCS,$BDDAT    ;READ STCS
2019  020252  023737  001126  001124            CMP     $BDDAT,$GDDAT   ;IS STCS OK?
2020  020260  001403                            BEQ     5$
2021  020262  104044                            ERROR+  44              ;STCS STATUS ERROR ON ST XFER
2022  020264  004737  022234                    JSR     PC,DMPREG
2023  020270  005037  001124          5$:       CLR     $GDDAT          ;SET EXPECTED RTE FLAGS
2024  020274  012777  000010  003270            MOV     #EXCE,@ACR
2025  020302  017737  003270  001126            MOV     @EXC,$BDDAT     ;GET EXC REGISTER
2026  020310  023737  001124  001126            CMP     $GDDAT,$BDDAT
2027  020316  001403                            BEQ     6$
2028  020320  104044                            ERROR+  44              ;EXC NOT 0 ON A ST XFER
2029  020322  004737  022234                    JSR     PC,DMPREG
2030  020326  036437  020526  001200  6$:       BIT     STITAB(4),$TMP1 ;CHECK FOR STF
2031  020334  001403                            BEQ     61$             ;SKIP IF 0
2032  020336  056437  020526  001124            BIS     STITAB(4),$GDDAT;SET STF
2033  020344  036437  020536  001200  61$:      BIT     STBTAB(4),$TMP1 ;CHECK FOR STB
2034  020352  001403                            BEQ     62$             ;SKIP IF 0
2035  020354  056437  020536  001124            BIS     STBTAB(4),$GDDAT;SET STB
2036  020362  012777  000006  003202  62$:      MOV     #STFE,@ACR
2037  020370  017737  003202  001126            MOV     @STF,$BDDAT     ;READ STF
2038  020376  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CHECK DATA
2039  020404  001403                            BEQ     7$
2040  020406  104044                            ERROR+  44              ;STF ERROR ON ST XFER
2041  020410  004737  022234                    JSR     PC,DMPREG
2042  020414  005037  001124          7$:       CLR     $GDDAT          ;EXPECTED = 0
2043  020420  012777  000005  003144            MOV     #PGFE,@ACR      ;SELECT PGF
2044  020426  017737  003144  001126            MOV     @PGF,$BDDAT     ;GET PGF
2045  020434  001403                            BEQ     8$              ;PGF SHOULD BE 0
2046  020436  104044                            ERROR+  44              ;PGF NO 0 ON A ST XFER
2047  020440  004737  022234                    JSR     PC,DMPREG
2048  020444  013737  023616  001124  8$:       MOV     SNAP0,$GDDAT    ;GET EXPECTED DATA FOR DCF
2049  020452  012777  000007  003112            MOV     #DCFE,@ACR      ;SELECT DCF
2050  020460  017737  003112  001126            MOV     @DCF,$BDDAT     ;GET DCF REGISTER
2051  020466  023737  001126  001124            CMP     $BDDAT,$GDDAT   ;CHECK THAT THERE IS NO CHANGE
2052  020474  001403                            BEQ     9$
2053  020476  104044                            ERROR+  44              ;DCF ERROR ON A ST XFER
2054  020500  004737  022234                    JSR     PC,DMPREG
2055  020504  105200                  9$:       INCB    R0              ;INCREMENT TEST DATA
2056  020506  001402                            BEQ     10$             ;SKIP WHEN DONE
2057  020510  000137  017740                    JMP     1$
2058  020514  000207                 10$:       RTS     PC              ;EXIT
```

```
                                        .SBTTL  LINE CONTROL BIT TABLES

020516  000000                  SIDTAB: SID0                    ;SID TABLE
020520  000400                          SID1
020522  001000                          SID2
020524  001400                          SID3

020526                          PGITAB:                         ;PROGRAM INTERRUPT ENABLE TABLE
020526                          STITAB:                         ;SANITY INTERRUPT ENABLE TABLE
020526                          IMITAB:                         ;INTERRUPT MASK INHIBIT TABLE
020526                          PIFTAB:                         ;PGM INTERRUPT FLAG TABLE
020526                          SIFTAB:                         ;SANITY INTERRUPT FLAG TABLE
020526                          DCFTAB:                         ;DCLO INTERRUPT FLAG TABLE
020526  000001                  RTETAB: RTE0                    ;INVALID INPUT FLAG TABLE
020530  000002                          RTE1
020532  000004                          RTE2
020534  000010                          RTE3

020536                          PGBTAB:                         ;PROGRAM BOOT ENABLE TABLE
020536                          STBTAB:                         ;SANITY BOOT ENABLE TABLE
020536                          BMITAB:                         ;BOOT MASK INHIBIT TABLE
020536                          PBFTAB:                         ;PGM BOOT FLAG TABLE
020536                          SBFTAB:                         ;SANITY BOOT FLAG TABLE
020536                          BRKTAB:                         ;BREAK STATE TABLE
020536  000400                  UVITAB: UI0                     ;UNEXPECTED INPUT FLAG TABLE
020540  001000                          UI1
020542  002000                          UI2
020544  004000                          UI3
```

```
                                ENDTST:

                                ;ON FIRST PASS, DETERMINE SANITY-TIMER COUNT RATE (256 US, 8.192 MS,
                                ;OR 16.384 MS) AND PRINT IT OUT.
                                ;ON SUBSEQUENT PASSES, CHECK FOR CONSISTENCY.
                                ;
                                ;MAINTENANCE LOOPBACK IS USED, WITH HARDWARE SELF-ID.

                                ;;*****************************************************************
                                ;*TEST 56       DETERMINE SANITY TIMER COUNT RATE
                                ;;*****************************************************************
                                TST56:  SCOPE
020546  000004                    
020550  012737  000340  177776          MOV     #340,PSW        ;RAISE PROCESSOR PRIORITY
020556  004737  022042                  JSR     PC,SETMNT       ;INIT THE INTERFACE
020562  012777  000015  003002          MOV     #MTCE,@ACR      ;SELECT MTC REGISTER
020570  012777  000013  003000          MOV     #DSBT+MLEN+MDSD,@MTC    ;SET LOOPBACK, DISABLE BOOT & DRIVERS
020576  117704  002772                  MOVB    @ACRH,R4        ;GET HARDWARE SELF-ID
020602  010437  023614                  MOV     R4,LINMBR       ;PUT IT INTO LINE NUMBER WORD
020606  006304                          ASL     R4              ;MAKE A WORD INDEX
020610  012777  000002  002754          MOV     #STTEE,@ACR     ;SELECT STTE (S.T. TRANSMIT ENABLE)
020616  016477  020526  002752          MOV     STITAB(R4),@STTE        ;SET XMIT ENABLE FOR SELF
020624  017746  002754                  MOV     @IIIP,-(SP)     ;SAVE OLD INTR VECTOR INFO
020630  017746  002746                  MOV     @IIIV,-(SP)
020634  012777  020770  002740          MOV     #4$,@IIIV       ;SETUP NEW INTR VECTOR
020642  012777  000340  002734          MOV     #340,@IIIP
020650  005000                          CLR     R0              ;CLEAR CHARACTER COUNT
020652  013701  023572                  MOV     ACR,R1          ;R1 = ACR ADDRESS
020656  013702  023576                  MOV     ADR,R2          ;R2 = ADR ADDRESS
020662  012703  000001                  MOV     #PGCSE,R3       ;R3 = PGCS ACCESS CODE
020666  012704  004000                  MOV     #PRDY,R4        ;R4 = PG RDY BIT MASK
020672  012705  000007                  MOV     #PTP+IE+GO,R5   ;R5 = VALUE FOR LOADING PGCS
020676  012711  000003                  MOV     #STCSE,(R1)     ;SELECT STCS
020702  012712  000401                  MOV     #401,(R2)       ;START SANITY TIMER FOR DOING 2 COUNTS
020706  005037  177776                  CLR     PSW             ;ALLOW INTERRUPTS

020712  010311                  1$:     MOV     R3,(R1)         ;SELECT PGCS
020714  010512                          MOV     R5,(R2)         ;XMIT NULL PG COMMAND, SET INTR ENABLE
020716  005200                          INC     R0              ;COUNT # OF CHARS TRANSMITTED
020720  001404                          BEQ     3$              ;BUT DON'T DO FOREVER.
020722  010311                  2$:     MOV     R3,(R1)         ;SELECT PGCS
020724  030412                          BIT     R4,(R2)         ;LOOK FOR PG RDY
020726  001371                          BNE     1$              ;XMIT AGAIN IF SET
020730  000774                          BR      2$              ;LOOP IF NOT

                                ;COME HERE IF NO TIMEOUT INTR SEEN AFTER 64K TRANSMITS.
020732  012737  000340  177776  3$:     MOV     #340,PSW        ;RAISE PROCESSOR TO LOCKOUT INTERRUPTS
020740  012737  177410  001124          MOV     #177400+TMO,$GDDAT ;SETUP EXPECTED DATA
020746  012711  000003                  MOV     #STCSE,(R1)     ;SELECT STCS
020752  011237  001126                  MOV     (R2),$BDDAT     ;GET CONTENTS OF STCS FOR PRINTOUT
020756  104044                          ERROR+  44              ;NO TIMEOUT SEEN AFTER 64K TRANSMITS.
020760  004737  022234                  JSR     PC,DMPREG
020764  000137  022014                  JMP     99$             ;THEN JUST EXIT

                                ;COME HERE ON INTERRUPT, CHECK THAT IT'S THE CORRECT ONE.
020770  062706  000004          4$:     ADD     #4,SP           ;POP STACK
020774  010046                          MOV     R0,-(SP)        ;SAVE COUNT
```

```
2142  020776  010311                          MOV     R3,(R1)          ;SELECT PGCS
2143  021000  012705  000062                  MOV     #50.,R5          ;WAIT FOR FINAL PG RDY
2144  021004  005305                          DEC     R5
2145  021006  001376                          BNE     .-2
2146  021010  011237  001126                  MOV     (R2),$BDDAT      ;GET PGCS CONTENTS
2147  021014  011137  001124                  MOV     (R1),$GDDAT      ;GET SELF-ID
2148  021020  042737  000017  001124          BIC     #17,$GDDAT       ;CLEAR CODE
2149  021026  052737  004016  001124          BIS     #PRDY+IE+IP+PTP,$GDDAT  ;SET UP EXPECTED VALUE
2150  021034  023737  001126  001124          CMP     $BDDAT,$GDDAT    ;VERIFY PGCS CONTENTS
2151  021042  001403                          BEQ     5$
2152  021044  104044                          ERROR+  44               ;PGCS INCORRECT AFTER INTERRUPT
2153  021046  004737  022234                  JSR     PC,DMPREG
2154  021052  005037  001124          5$:     CLR     $GDDAT           ;SETUP EXPECTED VALUE FOR PGF, DCF, AND EXC
2155  021056  012777  000005  002506          MOV     #PGFE,@ACR       ;SELECT PGF
2156  021064  017737  002506  001126          MOV     @PGF,$BDDAT      ;GET PG RECEIVE FLAGS
2157  021072  001403                          BEQ     6$               ;SKIP IF 0
2158  021074  104044                          ERROR+  44               ;PGF NOT 0 AFTER INTERRUPT
2159  021076  004737  022234                  JSR     PC,DMPREG
2160  021102  012777  000007  002462  6$:     MOV     #DCFE,@ACR       ;SELECT DCF
2161  021110  017737  002462  001126          MOV     @DCF,$BDDAT      ;GET FLAGS & TEST THEM
2162  021116  001403                          BEQ     7$               ;  -- SHOULD BE 0
2163  021120  104044                          ERROR+  44               ;DCF NOT 0 AFTER INTERRUPT
2164  021122  004737  022234                  JSR     PC,DMPREG
2165  021126  012777  000010  002436  7$:     MOV     #EXCE,@ACR       ;SELECT EXC
2166  021134  017737  002436  001126          MOV     @EXC,$BDDAT      ;GET FLAGS & TEST THEM
2167  021142  001403                          BEQ     8$               ; -- SHOULD BE 0
2168  021144  104044                          ERROR+  44               ;EXC NOT CLEAR AFTER INTERRUPT
2169  021146  004737  022234                  JSR     PC,DMPREG
2170  021152  012737  000017  001124  8$:     MOV     #17,$GDDAT       ;SETUP EXPECTED VALUE FOR STF
2171  021160  012777  000006  002404          MOV     #STFE,@ACR       ;SELECT STF
2172  021166  017737  002404  001126          MOV     @STF,$BDDAT      ;GET STF
2173  021174  023737  001126  001124          CMP     $BDDAT,$GDDAT    ;CHECK VALUE
2174  021202  001403                          BEQ     9$
2175  021204  104044                          ERROR+  44               ;STF INCORRECT AFTER INTERRUPT
2176  021206  004737  022234                  JSR     PC,DMPREG
2177
2178  021212  012637  001126          9$:     MOV     (SP)+,$BDDAT     ;GET COUNT VALUE
2179  021216  005737  023662                  TST     STCCNT           ;IS THERE A VALID COUNT?
2180  021222  001402                          BEQ     .+6
2181  021224  000137  021734                  JMP     20$              ;IF YES, GO VERIFY CONSISTENCY.
2182  021230  012737  000014  001124          MOV     #12.,$GDDAT      ;NO -- SO START CHECKING VALUE.
2183  021236  023737  001126  001124          CMP     $BDDAT,$GDDAT    ;IS COUNT .LT 14.?
2184  021244  002002                          BGE     .+6              ;GO ON IF NO
2185  021246  000137  021724                  JMP     12$              ;ERROR IF YES
2186  021252  012737  000024  001124          MOV     #20.,$GDDAT
2187  021260  023737  001126  001124          CMP     $BDDAT,$GDDAT    ;
2188  021266  003047                          BGT     10$              ;GREATER THAN 20.?
2189  021270  013737  001126  023662          MOV     $BDDAT,STCCNT    ;NO, SO BETWEEN 12 & 21 -- VALID
2190  021276  012737  000024  023666          MOV     #20.,MAXCNT      ;SAVE MAX LIMIT
2191  021304  012737  000014  023664          MOV     #12.,MINCNT      ;AND MIN LIMIT, FOR FUTURE TESTING.
2192  021312  005737  000042                  TST     @#42             ;ARE WE IN ACT?
2193  021316  001402                          BEQ     .+6              ;BEQ IF NO.
2194  021320  000137  022014                  JMP     99$              ;IF YES, DON'T PRINT.
2195  021324  104401  021332                  TYPE    ,65$             ;;TYPE ASCIZ STRING
  (1) 021330  000424                          BR      64$              ;;GET OVER THE ASCIZ
  (1)                             ;;65$:      .ASCIZ  <200>/SANITY TIMER SET FOR 256US PER COUNT./<200>
```

```
   (1)  021402                              64$:
2196    021402  000137  022014                      JMP     99$
2197
2198    021406  012737  000620  001124  10$:        MOV     #400.,$GDDAT
2199    021414  023737  001126  001124              CMP     $BDDAT,$GDDAT   ;CHECK FOR COUNT BETWEEN 400.
2200    021422  002540                              BLT     12$
2201    021424  012737  001200  001124              MOV     #640.,$GDDAT
2202    021432  023727  001126  001200              CMP     $BDDAT,#640.    ;AND 640.
2203    021440  003045                              BGT     11$
2204    021442  013737  001126  023662              MOV     $BDDAT,STCCNT   ;SAVE COUNT
2205    021450  012737  000620  023664              MOV     #400.,MINCNT    ;AND LIMITS
2206    021456  012737  001200  023666              MOV     #640.,MAXCNT
2207    021464  005737  000042                      TST     @#42            ;ARE WE IN ACT?
2208    021470  001151                              BNE     99$             ;IF YES, DON'T PRINT.
2209    021472  104401  021500                      TYPE    ,67$            ;;TYPE ASCIZ STRING
   (1)  021476  000425                              BR      66$             ;;GET OVER THE ASCIZ
   (1)                              ;;67$:  .ASCIZ  <200>/SANITY TIMER SET FOR 8.192MS PER COUNT./<200>
   (1)  021552                      66$:
2210    021552  000520                              BR      99$
2211
2212    021554  012737  001440  001124  11$:        MOV     #800.,$GDDAT
2213    021562  023737  001126  001124              CMP     $BDDAT,$GDDAT   ;COUNT .LE 800.?
2214    021570  002455                              BLT     12$             ;ERROR IF YES.
2215    021572  012737  002400  001124              MOV     #1280.,$GDDAT
2216    021600  023727  001126  002400              CMP     $BDDAT,#1280.   ;GREATER THAN 1280.?
2217    021606  003046                              BGT     12$             ;ERROR IF YES.
2218    021610  013737  001126  023662              MOV     $BDDAT,STCCNT   ;SAVE COUNT
2219    021616  012737  001440  023664              MOV     #800.,MINCNT    ;AND LIMITS
2220    021624  012737  002400  023666              MOV     #1280.,MAXCNT
2221    021632  005737  000042                      TST     @#42            ;ARE WE IN ACT?
2222    021636  001066                              BNE     99$
2223    021640  104401  021646                      TYPE    ,69$            ;;TYPE ASCIZ STRING
   (1)  021644  000426                              BR      68$             ;;GET OVER THE ASCIZ
   (1)                              ;;69$:  .ASCIZ  <200>/SANITY TIMER SET FOR 16.384MS PER COUNT./<200>
   (1)  021722                      68$:
2224    021722  000434                              BR      99$
2225
2226                                ;HERE ON ERROR
2227    021724  104044              12$:        ERROR+  44              ;CAN'T RESOLVE SANITY TIMER VALUE
2228    021726  004737  022234                      JSR     PC,DMPREG       ;(COUNTS NOT WITHIN LIMITS
2229    021732  000430                              BR      99$
2230
2231                                ;HERE ON SUBSEQUENT PASSES TO CHECK CONSISTENCY
2232    021734  013737  023666  001124  20$:        MOV     MAXCNT,$GDDAT   ;GET EXPECTED MAX LIMIT FOR TYPEOUT.
2233    021742  013737  023662  023610              MOV     STCCNT,DISPLY   ;DISPLAY COUNT AS FOUND ON FIRST PASS.
2234    021750  023737  001126  001124              CMP     $BDDAT,$GDDAT   ;SEE THAT COUNT NOT EXCEEDED.
2235    021756  003404                              BLE     21$             ;OK IF .LE MAX LIMIT.
2236    021760  104044                              ERROR+  44              ;COUNT GREATER THAN MAX OF RANGE.
2237    021762  004737  022234                      JSR     PC,DMPREG
2238    021766  000412                              BR      99$
2239    021770  013737  023664  001124  21$:        MOV     MINCNT,$GDDAT   ;GET EXPECTED MIN LIMIT.
2240    021776  023737  001126  001124              CMP     $BDDAT,$GDDAT   ;CHECK
2241    022004  003003                              BGT     99$             ;OK IF ACTUAL .GT MIN LIMIT.
2242    022006  104044                              ERROR+  44              ;COUNT LESS THAN MIN OF RANGE.
2243    022010  004737  022234                      JSR     PC,DMPREG
2244
```

```
 2245  022014  012677  001562          99$:    MOV     (SP)+,@IIIV     ;RESTORE VECTOR
 2246  022020  012677  001560                  MOV     (SP)+,@IIIP     ;
 2247  022024  012777  100000  001540          MOV     #CLR,@ACR       ;INIT THE INTERFACE
 2248
 2249
 2250  022032  000005                          RESET
 2251  022034  000240                          NOP
 2252  022036  000137  023670                  JMP     $EOP
```

```
                                        .SBTTL  TEST AND UTILITY SUBROUTINES

                                ;SET THE IIST TO INTERNAL LOOPBACK MAINTENENCE MODE USING
                                ;THE SID VALUE POINTED TO BY R4

022042  005037  022202          SETMNT: CLR     MNTBTS          ;CLEAR EXTRA MNT BITS
022046  012777  100000  001516  SETMN1: MOV     #CLR,@ACR       ;RESET THE INTERFACE
022054  016400  020516                  MOV     SIDTAB(4),R0    ;GET SID VALUE
022060  062700  000016                  ADD     #TMT,R0         ;ADD LOOPBACK + BOOT DISABLE TO MNT. SID
022064  063700  022202                  ADD     MNTBTS,R0       ;ADD ANY EXTRA MNT BITS
022070  027777  001476  001474          CMP     @ACR,@ACR       ;SHORT DELAY TO ALLOW RESET TO WORK
022076  027777  001470  001466          CMP     @ACR,@ACR
022104  012777  000015  001460          MOV     #MTCE,@ACR      ;CHOOSE MTC
022112  010077  001460                  MOV     R0,@MTC ;LOAD MTC
022116  012777  000001  001446          MOV     #PGCSE,@ACR     ;SELECT PGCS
022124  017737  001446  023610          MOV     @PGCS,DISPLY    ;READ PGCS
022132  032737  004000  023610          BIT     #PRDY,DISPLY    ;CHECK FOR RDY
022140  001003                          BNE     1$              ;SKIP IF SET
022142  104044                          ERROR+  44              ;DEVICE CLR DID NOT SET PGCS RDY
022144  004737  022234                  JSR     PC,DMPREG
022150  012777  000005  001414  1$:     MOV     #PGFE,@ACR      ;SELECT PGF
022156  012777  007400  001412          MOV     #7400,@PGF      ;CLEAR BOOT FLAGS
022164  012777  007400  001404          MOV     #7400,@STF      ;CLEAR BOOT FLAGS
022172  012777  000017  001376          MOV     #17,@DCF        ;CLEAR DCF FLAGS
022200  000207                          RTS     PC              ;EXIT

022202  000000          MNTBTS: 0

                                ;SET MAINTENENCE MODE SO THAT SANITY TIMER OPERATIONS ACT
                                ;LIKE PROGRAM GENERATED OPERATIONS

022204  012737  004000  022202  SSTMNT: MOV     #MTYP,MNTBTS    ;SET TYP TO A 1
022212  000715                  BR      SETMN1          ;EXIT THROUGH SETMNT

                                ;PARITY CALCULATION SUBROUTINE
                                ;COUNTS THE NUMBER OF 1 BITS IN R0 IN R1

022214  010046          PARCAL: MOV     R0,-(SP)
022216  005001                  CLR     R1
022220  006300          1$:     ASL     R0
022222  005501                  ADC     R1
022224  005700                  TST     R0
022226  001374                  BNE     1$
022230  012600                  MOV     (SP)+,R0
022232  000207                  RTS     PC
```

```
                                        ;DUMP ALL REGISTERS ROUTINE
                                        ;
022234  032777  000001  156676  DMPREG: BIT     #BIT0,@SWR      ;TEST BIT0
022242  001001                          BNE     1$              ;SKIP IF SET
022244  000207                          RTS     PC              ;EXIT IF 0
022246  104401  033561                1$:     TYPE    ,HDRMES
022252  012737  000011  022334            MOV     #9.,TCR         ;DO 9 TIMES
022260  017746  001306                      MOV     @ACR,-(SP)      ;;PUSH @ACR ON STACK
022264  005077  001302                      CLR     @ACR            ;INIT ADDRESS
022270                                  2$:
(1) 022270  017746  001302                  MOV     @ADR,-(SP)      ;;SAVE @ADR FOR TYPEOUT
(1) 022274  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
022276  104401  033671                      TYPE    ,TWOSP
022302  005337  022334                      DEC     TCR             ;DONE 9?
022306  001370                          BNE     2$              ;DO 9
022310  104401  033710                      TYPE    ,CARET
022314  012677  001252                      MOV     (SP)+,@ACR      ;;POP STACK INTO @ACR
022320  005777  156614                      TST     @SWR            ;CHECK FOR ERR HLT
022324  100002                          BPL     3$              ;SKIP IF 0
022326  000000                          HALT
022330  104407                          CKSWR
022332  000207                  3$:     RTS     PC              ;EXIT
                                        ;
022334  000000                  TCR:    0
                                        ;
                                        ;ROUTINE TO GET DEVICE PARAMETERS
                                        ;
022336                          GETPAR:
(1) 022336  104401  022344              TYPE    ,65$            ;;TYPE ASCIZ STRING
(1) 022342  000444                      BR      64$             ;;GET OVER THE ASCIZ
(1)                             ;;65$:  .ASCIZ  <200>/IN THE FOLLOWING THREE PRINTOUTS, TYPE THE NEW DATA OR 0 IF NO CHA
(1) 022454                      64$:
022454  104401  033465          10$:    TYPE    ,ASKDVA
022460  013746  023572                  MOV     ACR,-(SP)       ;;SAVE ACR FOR TYPEOUT
(1) 022464  104402                      TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
022466  104401  033667                  TYPE    ,SPACE
022472  104412                          RDOCT
022474  012600                          MOV     (SP)+,R0        ;GET VALUE TYPED IN.
022476  001402                          BEQ     11$             ;IF ZERO, DON'T CHANGE.
022500  010037  023572                  MOV     R0,ACR          ;STORE NEW VALUE
022504  023727  023572  160000  11$:    CMP     ACR,#160000     ;CHECK FOR LEGALITY
022512  103760                          BLO     10$             ;LOOP IF BAD
022514  032737  000003  023572          BIT     #3,ACR          ;CHECK FOR MOD. 4
022522  001354                          BNE     10$             ;LOOP IF BAD
022524  013737  023572  023574          MOV     ACR,ACRH        ;LOAD OTHER ADDRESS SLOTS
022532  005237  023574                  INC     ACRH
022536  013737  023574  023576          MOV     ACRH,ADR
022544  005237  023576                  INC     ADR
022550  013737  023576  023600          MOV     ADR,ADRH
022556  005237  023600                  INC     ADRH
022562  104401  033505          1$:     TYPE    ,ASKPIV
022566  013746  023602                  MOV     IIIV,-(SP)      ;;SAVE IIIV FOR TYPEOUT
(1)                                     ;;1
(1) 022572  104403                      TYPOS                   ;;GO TYPE--OCTAL ASCII
(1) 022574     006                      .BYTE   6               ;;TYPE 6 DIGITS
(1) 022575     000                      .BYTE   0               ;;SUPPRESS LEADING ZEROS
```

```
 2345  022576  104401  033667                    TYPE    ,SPACE
 2346  022602  104412                            RDOCT
 2347  022604  012600                            MOV     (SP)+,R0        ;GET VALUE TYPED IN.
 2348  022606  001402                            BEQ     12$             ;IF 0, DON'T CHANGE.
 2349  022610  010037  023602                    MOV     R0,IIIV         ;LOAD NEW VALUE
 2350  022614  023727  023602  002000  12$:     CMP     IIIV,#2000      ;CHECK FOR LEGALITY
 2351  022622  103357                            BHIS    1$              ;LOOP IF BAD
 2352  022624  032737  000003  023602            BIT     #3,IIIV         ;CHECK FOR MOD. 4
 2353  022632  001353                            BNE     1$              ;LOOP IF BAD
 2354  022634  013737  023602  023604            MOV     IIIV,IIIP       ;LOAD PIL
 2355  022642  062737  000002  023604            ADD     #2,IIIP         ;WITH IIIV+2
 2356  022650  104401  033540          2$:       TYPE    ,ASKBRL
 2357  022654  013746  023606                    MOV     IIIL,-(SP)      ;;SAVE IIIL FOR TYPEOUT
  (1)                                                                    ;;1
  (1)  022660  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
  (1)  022662     006                            .BYTE   6               ;;TYPE 6 DIGITS
  (1)  022663     000                            .BYTE   0               ;;SUPPRESS LEADING ZEROS
 2358  022664  104401  033667                    TYPE    ,SPACE
 2359  022670  104412                            RDOCT
 2360  022672  012600                            MOV     (SP)+,R0        ;GET VALUE TYPED IN.
 2361  022674  001402                            BEQ     21$             ;IF ZERO, DON'T CHANGE.
 2362  022676  010037  023606                    MOV     R0,IIIL         ;GET BR LEVEL
 2363  022702  023727  023606  000004  21$:     CMP     IIIL,#4         ;CHECK FOR LEGALITY
 2364  022710  103757                            BLO     2$              ;LOOP IF BAD
 2365  022712  023727  023606  000007            CMP     IIIL,#7         ;CHECK
 2366  022720  101353                            BHI     2$              ;LOOP IF BAD
 2367  022722  000137  023346                    JMP     3$              ;JUMP OVER HELP MSG.
 2368  022726                          30$:
  (1)  022726  104401  022734                    TYPE    ,67$            ;;TYPE ASCIZ STRING
  (1)  022732  000422                            BR      66$             ;;GET OVER THE ASCIZ
  (1)                                    ;;67$:  .ASCIZ  <200>/HARDWARE CONFIGURATION CODES ARE:/
  (1)  023000                            66$:
 2369  023000  104401  023006                    TYPE    ,69$            ;;TYPE ASCIZ STRING
  (1)  023004  000432                            BR      68$             ;;GET OVER THE ASCIZ
  (1)                                    ;;69$:  .ASCIZ  <CRLF>/   0: MAINT. MODE ONLY;EITHER NO TERM, OR INHIBITS/
  (1)  023072                            68$:
 2370  023072  104401  023100                    TYPE    ,71$            ;;TYPE ASCIZ STRING
  (1)  023076  000436                            BR      70$             ;;GET OVER THE ASCIZ
  (1)                                    ;;71$:  .ASCIZ  <CRLF>/   1: IIST BUS TERMINATED, WITH NO INTERFERING INTERFACES/
  (1)  023174                            70$:
 2371  023174  104401  023202                    TYPE    ,73$            ;;TYPE ASCIZ STRING
  (1)  023200  000427                            BR      72$             ;;GET OVER THE ASCIZ
  (1)                                    ;;73$:  .ASCIZ  <CRLF>/   2: OTHER INTERFACES ONLINE WITH IDLE XMIT/
  (1)  023260                            72$:
 2372  023260  104401  023266                    TYPE    ,75$            ;;TYPE ASCIZ STRING
  (1)  023264  000430                            BR      74$             ;;GET OVER THE ASCIZ
  (1)                                    ;;75$:  .ASCIZ  <CRLF>/   3: OTHER ACTIVE NON-INHIBITED UNITS ONLINE/
  (1)  023346                            74$:
 2373  023346                          3$:
  (1)  023346  104401  023354                    TYPE    ,77$            ;;TYPE ASCIZ STRING
  (1)  023352  000436                            BR      76$             ;;GET OVER THE ASCIZ
  (1)                                    ;;77$:  .ASCIZ  <200>/ENTER HARDWARE CONFIGURATION TYPE (0-3) OR 111 FOR HELP: /
  (1)  023450                            76$:
 2374  023450  104412                            RDOCT
 2375  023452  012637  023660                    MOV     (SP)+,CONFIG    ;GET VALUE
 2376  023456  023727  023660  000003            CMP     CONFIG,#3       ;SEE IF VALID
```

```
2377  023464  101402                        BLOS    .+6
2378  023466  000137  022726                JMP     30$             ;IF NOT, DO HELP MSG.
2379  023472  000207                        RTS     PC              ;EXIT
2380                                ;ROUTINE TO REPORT THE SELF ID OF THE IIST
2381
2382  023474  005737  001236        GETSID: TST     $PASS           ;PASS 1?
2383  023500  001401                        BEQ     1$              ;SKIP IF YES
2384  023502  000207                        RTS     PC              ;EXIT IF NOT
2385  023504  005737  000042        1$:     TST     @#42            ;ARE WE IN ACT?
2386  023510  001002                        BNE     2$              ;IF YES, DON'T PRINT
2387  023512  104401  033674                TYPE    ,DMPSID
2388  023516  012777  100000  000046  2$:   MOV     #CLR,@ACR       ;CLEAR INTERFACE
2389  023524  017700  000042                MOV     @ACR,R0         ;GET SID
2390  023530  000300                        SWAB    R0              ;INTO B2-0
2391  023532  042700  177770                BIC     #177770,R0      ;CLEAR B15-3
2392  023536  010037  023570                MOV     R0,SIDNUM       ;SAVE SELF ID NUM.
2393  023542  005737  000042                TST     @#42            ;IN ACT
2394  023546  001007                        BNE     3$              ;DON'T PRINT IF YES.
2395  023550  010046                        MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
  (1)                                                             ;;1
  (1)   023552  104403                      TYPOS                   ;;GO TYPE--OCTAL ASCII
  (1)   023554     006                      .BYTE   6               ;;TYPE 6 DIGITS
  (1)   023555     000                      .BYTE   0               ;;SUPPRESS LEADING ZEROS
2396  023556  104401  033710                TYPE    ,CARET
2397  023562  104401  033710                TYPE    ,CARET
2398  023566  000207                3$:     RTS     PC              ;EXIT
2399
2400  023570  000000                SIDNUM: 0
```

```
                                        .SBTTL  DEVICE ADDRESS PARAMETER BLOCK

023572  177500          ACR:    177500          ;ACCESS CONTROL REGISTER
023574  177501          ACRH:   177501          ;HIGH BYTE OF ACR

023576                  ADR:                    ;ACCESS DATA REGISTER
023576                  PGTE:                   ;PGM-GEN. XMIT ENABLES REGISTER
023576                  PGCS:                   ;PGM-GEN. CSR
023576                  STTE:                   ;SANITY TIMER XMIT ENABLES REGISTER
023576                  STCS:                   ;SANITY TIMER CSR
023576                  IMSK:                   ;INPUT MASKS REGISTER
023576                  PGF:                    ;PGM-GEN. I/B FALAGS REGISTER
023576                  STF:                    ;SANITY TIMER FLAGS REGISTER
023576                  DCF:                    ;DCLO/DISCONNECT FLAGS REGISTER
023576                  EXC:                    ;EXCEPTIONS REGISTER
023576  177502          MTC:    177502          ;MAINTENANCE REGISTER
023600                  ADRH:                   ;ADR HIGH BYTE
023600                  PGCSH:                  ;PGCS HIGH BYTE
023600                  STCSH:                  ;STCS HIGH BYTE
023600                  IMSKH:                  ;IMSK HIGH BYTE
023600                  PGFH:                   ;PGF HIGH BYTE
023600                  DCFH:                   ;DCF HIGH BYTE
023600                  EXCH:                   ;EXC HIGH BYTE
023600  177503          MTCH:   177503          ;MTC HIGH BYTE
023602  000260          IIIV:   260             ;IIST INTERRUPT VECTOR ADDRESS
023604  000262          IIIP:   262             ;IIST INTERRUPT PRIORITY ADDRESS
023606  000006          IIIL:   6               ;IIST INTERRUPT LEVEL


                        ;CONSTANTS AND VARIABLES

023610  000000          DISPLY: 0
023612  000000          TIMPER: 0
023614  000000          LINMBR: 0
023616  000020          SNAPO:  .BLKW   16.

023656  000000          PARFLG: .WORD   0

023660  000000          CONFIG: .WORD   0

023662  000000          STCCNT: .WORD   0       ;COUNT OF # PG CHARACTERS TRANSMITTED DURING
                                                ;TWO COUNTS OF THE SANITY TIMER.
023664  000000          MINCNT: .WORD   0       ;MINIMUM ALLOWED VALUE OF STCCNT.
023666  000000          MAXCNT  :.WORD  0       ;MAXIMUM ALLOWED VALUE OF STCCNT.
```

```
2447                                    .SBTTL  END OF PASS ROUTINE
 (1)
 (2)                                    ;;**********************************************************************
 (1)                                    ;*INCREMENT THE PASS NUMBER ($PASS)
 (1)                                    ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
 (1)                                    ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
 (1)                                    ;*IF THERES A MONITOR GO TO IT
 (1)                                    ;*IF THERE ISN'T JUMP TO TST1
 (1)
 (1)  023670                            $EOP:
 (1)  023670  000004                            SCOPE
 (1)  023672  005037  001102                    CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 (1)  023676  005037  001214                    CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 (1)  023702  005237  001236                    INC     $PASS           ;;INCREMENT THE PASS NUMBER
 (1)  023706  042737  100000  001236            BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 (1)  023714  005327                            DEC     (PC)+           ;;LOOP?
 (1)  023716  000001            $EOPCT: .WORD   1
 (1)  023720  003022                            BGT     $DOAGN          ;;YES
 (1)  023722  012737                            MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 (1)  023724  000001            $ENDCT: .WORD   1
 (1)  023726  023716                            $EOPCT
 (1)  023730  104401  023775                    TYPE    ,$ENDMG         ;;TYPE 'END PASS #'
 (2)  023734  013746  001236                    MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 (2)  023740  104405                            TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 (1)  023742  104401  023772                    TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 (1)  023746  013700  000042            $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 (1)  023752  001405                            BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 (1)  023754  000005                            RESET                   ;;CLEAR THE WORLD
 (1)  023756  004710            $ENDAD: JSR     PC,(R0)                 ;;GO TO MONITOR
 (1)  023760  000240                            NOP                     ;;SAVE ROOM
 (1)  023762  000240                            NOP                     ;;FOR
 (1)  023764  000240                            NOP                     ;;ACT11
 (1)  023766                            $DOAGN:
 (1)  023766  000137                            JMP     @(PC)+          ;;RETURN
 (1)  023770  002524            $RTNAD: .WORD   TST1
 (1)  023772     377     377     000    $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 (1)  023775     015  042412  042116    $ENDMG: .ASCIZ  <15><12>/END PASS #/
 (1)  024002  050040  051501  020123
 (1)  024010  000043
2448
2449                                    .SBTTL  SCOPE HANDLER ROUTINE
 (1)
 (2)                                    ;;**********************************************************************
 (1)                                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
 (1)                                    ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
 (1)                                    ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
 (1)                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 (1)                                    ;*SW14=1        LOOP ON TEST
 (1)                                    ;*SW11=1        INHIBIT ITERATIONS
 (1)                                    ;*SW09=1        LOOP ON ERROR
 (1)                                    ;*CALL
 (1)                                    ;*      SCOPE           ;;SCOPE=IOT
 (1)
 (1)  024012                            $SCOPE:
 (1)  024012  104407                            CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
 (1)                                    ;;GO TO ERROR ROUTINE IF RETURN PC LESS THAN 1002
```

```
(1)
(1)  024014  021627  001002              ;;OTHERWISE CONTINUE
(1)  024014  021627  001002                    CMP     (SP),#1002      ;;UNEXPECTED TRAP OR INTERRRUPT
(1)  024020  101002                            BHI     1$              ;;ARE TRAPPED HERE VIA IOT
(1)  024022  000137  024264                    JMP     $ERROR          ;;GO PROCESS UNEXPECTED TRAP
(1)  024026  032777  040000  155104  1$:      BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(1)  024034  001104                            BNE     $OVER           ;;YES IF SW14=1
(1)
(1)                                     ;#####START OF CODE FOR THE XOR TESTER#####
(1)  024036  000416                     $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                                                     ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(1)  024040  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  024044  012737  024064  000004            MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  024052  005737  177060                    TST     @#177060        ;;TIME OUT ON XOR?
(1)  024056  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  024062  000453                            BR      $SVLAD          ;;GO TO THE NEXT TEST
(1)  024064  022626                     5$:    CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(1)  024066  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  024072  000413                            BR      7$              ;;LOOP ON THE PRESENT TEST
(1)  024074                             6$:;#####END OF CODE FOR THE XOR TESTER#####
(1)  024074  105737  001103              2$:   TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(1)  024100  001421                            BEQ     3$              ;;BR IF NO
(1)  024102  123737  001115  001103            CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
(1)  024110  101015                            BHI     3$              ;;BR IF NO
(1)  024112  032777  001000  155020            BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
(1)  024120  001404                            BEQ     4$              ;;BR IF NO
(1)  024122  013737  001110  001106  7$:       MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
(1)  024130  000446                            BR      $OVER
(1)  024132  105037  001103              4$:   CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  024136  005037  001214                    CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  024142  000415                            BR      1$              ;;ESCAPE TO THE NEXT TEST
(1)  024144  032777  004000  154766  3$:       BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(1)  024152  001011                            BNE     1$              ;;BR IF YES
(1)  024154  005737  001236                    TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  024160  001406                            BEQ     1$              ;;      INHIBIT ITERATIONS
(1)  024162  005237  001104                    INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  024166  023737  001214  001104            CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  024174  002024                            BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(1)  024176  012737  000001  001104  1$:       MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  024204  013737  024262  001214            MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  024212  105237  001102              $SVLAD: INCB  $TSTNM          ;;COUNT TEST NUMBERS
(1)  024216  113737  001102  001234            MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
(1)  024224  011637  001106                    MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  024230  011637  001110                    MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(1)  024234  005037  001216                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  024240  112737  000001  001115            MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)  024246  013777  001102  154666  $OVER:    MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1)  024254  013716  001106                    MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  024260  000002                            RTI                     ;;FIXES PS
(1)  024262  000001                     $MXCNT: 1                      ;;MAX. NUMBER OF ITERATIONS
2450                                    .SBTTL  ERROR HANDLER ROUTINE
(1)
(2)                                     ;;**********************************************************************
(1)                                     ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
(1)                                     ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1)                                     ;*AND GO TO $ERRTYP ON ERROR
(1)                                     ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                     ;*SW15=1          HALT ON ERROR
```

```
(1)                                     ;*SW13=1        INHIBIT ERROR TYPEOUTS
(1)                                     ;*SW10=1        BELL ON ERROR
(1)                                     ;*SW09=1        LOOP ON ERROR
(1)                                     ;*CALL
(1)                                     ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(1)
(1)  024264                             $ERROR:
(1)  024264  104407                             CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
(1)  024266  105237  001103                7$:  INCB    $ERFLG          ;;SET THE ERROR FLAG
(1)  024272  001775                             BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(1)  024274  013777  001102  154640             MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  024302  032777  002000  154630             BIT     #BIT10,@SWR     ;;BELL ON ERROR?
(1)  024310  001402                             BEQ     1$              ;;NO - SKIP
(1)  024312  104401  001220                     TYPE    ,$BELL          ;;RING BELL
(1)  024316  005237  001112                1$:  INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
(1)  024322  011637  001116                     MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  024326  162737  000002  001116             SUB     #2,$ERRPC
(1)  024334  117737  154556  001114             MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  024342  032777  020000  154570             BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
(1)  024350  001055                             BNE     20$             ;;SKIP TYPEOUTS
(1)  024352  021627  001002                     CMP     (SP),#1002      ;;IF RETURN PC LESS THAN 1002
(1)  024356  101046                             BHI     12$             ;;ERROR IS ILLEGAL TRAP
(1)                                     ;;PROCESS UNEXPECTED TRAP OR INTERRUPT
(1)  024360  016637  000004  001116             MOV     4(SP),$ERRPC    ;;GET PC AT TIME OF FALSE TRAP
(1)  024366  162737  000002  001116             SUB     #2,$ERRPC       ;;ADJUST PC
(1)  024374  104401  024440                     TYPE    ,10$            ;;TYPE HEADER
(2)  024400  013746  001116                     MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)  024404  104402                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  024406  104401  024446                     TYPE    ,11$
(1)  024412  162716  000004                     SUB     #4,(SP)         ;;GET FALSE TRAP VECTOR ADDR
(1)  024416  011637  001116                     MOV     (SP),$ERRPC
(2)  024422  013746  001116                     MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)  024426  104402                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  024430  104401  001225                     TYPE    ,$CRLF
(1)  024434  022626                             CMP     (SP)+,(SP)+     ;;POP FALSE TRAP VECTOR PC&ADDR
(1)  024436  000422                             BR      20$
(1)  024440  050200  036503  000040  10$:  .ASCIZ  <200>'PC= '
(1)  024446  020040  047125  054105  11$:  .ASCIZ  '  UNEXPECTED TRAP TO '
(1)  024454  042520  052103  042105
(1)  024462  052040  040522  020120
(1)  024470  047524  000040
(1)
(1)                                             .EVEN
(1)  024474                             12$:
(1)  024474  004737  024606                     JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  024500  104401  001225                     TYPE    ,$CRLF
(1)  024504                             20$:
(1)  024504  122737  000001  001250             CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  024512  001007                             BNE     2$              ;;NO,SKIP APT ERROR REPORT
(1)  024514  113737  001114  024526             MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  024522  004737  027550                     JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
(1)  024526     000                     21$:    .BYTE   0
(1)  024527     000                             .BYTE   0
(1)  024530  000777                     22$:    BR      22$             ;;APT ERROR LOOP
(1)  024532  005777  154402                2$:  TST     @SWR            ;;HALT ON ERROR
(1)  024536  100002                             BPL     3$              ;;SKIP IF CONTINUE
(1)  024540  000000                             HALT                    ;;HALT ON ERROR!
```

```
(1)  024542  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
(1)  024544  032777  001000  154366  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
(1)  024552  001402                          BEQ     4$              ;;BR IF NO
(1)  024554  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(1)  024560  005737  001216          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(1)  024564  001402                          BEQ     5$              ;;BR IF NONE
(1)  024566  013716  001216                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  024572                          5$:
(1)  024572  022737  023756  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
(1)  024600  001001                          BNE     6$              ;;BRANCH IF NO
(1)  024602  000000                          HALT                    ;;YES
(1)  024604                          6$:
(1)  024604  000002                          RTI                     ;;RETURN
2451                                 .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
 (1)
 (2)                                 ;;**************************************************************************
 (1)                                 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 (1)                                 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
 (1)                                 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
 (1)
(1)  024606                          $ERRTYP:
(1)  024606  104401  001225                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  024612  010046                          MOV     R0,-(SP)        ;;SAVE R0
(1)  024614  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  024616  153700  001114                  BISB    @#$ITEMB,R0
(1)  024622  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                  ;;TYPE THE PC OF THE ERROR
(2)  024624  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                  ;;ERROR ADDRESS
(2)  024630  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  024632  000445                          BR      10$             ;;GET OUT
(1)  024634  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(1)  024636  006300                          ASL     R0              ;;       WORK FOR THE ERROR TABLE
(1)  024640  006300                          ASL     R0
(1)  024642  006300                          ASL     R0
(1)  024644  062700  001354                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(1)  024650  012037  024660                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(1)  024654  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(1)  024656  104401                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
(1)  024660  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  024662  104401  001225                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  024666  012037  024676          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
(1)  024672  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
(1)  024674  104401                          TYPE                    ;;TYPE THE "DATA HEADER"
(1)  024676  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
(1)  024700  104401  001225                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  024704  010146                  5$:     MOV     R1,-(SP)        ;;SAVE R1
(1)  024706  012001                          MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
(1)  024710  001415                          BEQ     9$              ;;BR IF NO DATA TO BE TYPED
(1)  024712  012000                          MOV     (R0)+,R0        ;;PICKUP "DATA FORMAT" POINTER
(1)  024714  105720                  6$:     TSTB    (R0)+           ;;"OCTAL" OR "DECIMAL"
(1)  024716  001003                          BNE     7$              ;;BR IF DECIMAL
(2)  024720  013146                          MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
(2)  024722  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  024724  000402                          BR      8$
(1)  024726                          7$:
```

```
(2)  024726  013146                    MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
(2)  024730  104405                    TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  024732  005711              8$:   TST     (R1)            ;;IS THERE ANOTHER NUMBER?
(1)  024734  001403                    BEQ     9$              ;;BR IF NO
(1)  024736  104401  024756            TYPE    ,11$            ;;TYPE TWO(2) SPACES
(1)  024742  000764                    BR      6$              ;;LOOP
(1)
(1)  024744  012601              9$:   MOV     (SP)+,R1        ;;RESTORE R1
(1)  024746  012600              10$:  MOV     (SP)+,R0        ;;RESTORE R0
(1)  024750  104401  001225            TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & 'LINE FEED''
(1)  024754  000207                    RTS     PC              ;;RETURN
(1)  024756  020040      000   11$:   .ASCIZ  /  /            ;;TWO(2) SPACES
(1)          024762                    .EVEN
2452                             .SBTTL  ROUTINE TO SIZE MEMORY
(1)
(2)                              ;;******************************************************************
(1)                              ;*CALL:
(1)                              ;*      JSR     PC,$SIZE
(1)                              ;*      RETURN
(1)                              ;*$LSTAD WILL CONTAIN:
(1)                              ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
(1)                              ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
(1)                              ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
(1)                              ;*$KT11 IS THE MEMORY MANAGEMENT KEY
(1)                              ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
(1)                              ;*      MUST BE SETUP BEFORE THE CALL
(1)                              ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
(1)                              ;*      DETERMINED BY ROUTINE
(1)
(1)  024762  010046              $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
(1)  024764  010146                      MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
(1)  024766  010246                      MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
(1)  024770  010346                      MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
(1)  024772  013746  000004              MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
(1)  024776  013746  000006              MOV     @#ERRVEC+2,-(SP)
(1)  025002  010600                      MOV     SP,R0           ;;SAVE THE STACK POINTER
(1)                              ;;SET THE ERRVEC PS TO THE PRESENT PS
(2)  025004  104400                      TRAP                    ;;PUSH OLD PSW AND PC ON STACK
(2)  025006  012637  000006              MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
(1)  025012  012701  003776              MOV     #3776,R1        ;;SETUP ADDRESS
(1)  025016  105727                      TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
(1)  025020  000200              $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
(1)  025022  100062                      BPL     $CORE           ;;BR IF NO
(1)  025024  012737  025162  000004      MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
(1)  025032  005737  177572              TST     @#SR0           ;;KT11 ARE YOU THERE?
(1)  025036  052737  100000  025020      BIS     #100000,$KT11   ;;YES--SET KT11 KEY
(1)  025044  005046                      CLR     -(SP)           ;;INITIALIZE FOR 'PAR'' LOADING
(1)  025046  012702  172340              MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST 'PAR'
(1)  025052  012703  000010              MOV     #^D8,R3         ;;LOAD EIGHT 'PAR.'S'' AND EIGHT 'PDR.'S''
(1)  025056  012762  077406  177740  1$: MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
(1)  025064  011622                      MOV     (SP),(R2)+      ;;LOAD 'PAR''
(1)  025066  062716  000200              ADD     #200,(SP)       ;;UPDATE FOR NEXT 'PAR''
(1)  025072  077307                      SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
(1)  025074  012742  177600              MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
(1)  025100  005042                      CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
(1)  025102  012737  025120  000004      MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
```

```
(1)  025110  012737  000020  172516          MOV     #20,@#SR3          ;;ENABLE 22 BIT MODE
(1)  025116  000401                          BR      3$                 ;;THIS PDP-11 HAS A SR3 REGISTER
(1)  025120  022626                  2$:     CMP     (SP)+,(SP)+        ;;CLEAN OFF THE STACK--NO SR3
(1)  025122  005237  177572          3$:     INC     @#SR0              ;;TURN ON MEMORY MANAGEMENT
(1)  025126  012737  025152  000004          MOV     #$KTOUT,@#ERRVEC   ;;SET FOR TIME OUT
(1)  025134  005737  143776          4$:     TST     @#143776           ;;TRAP ON NON-EX-MEM
(1)  025140  062712  000040                  ADD     #40,(R2)           ;;MAKE A 1K STEP
(1)  025144  023712  172356                  CMP     @#KIPAR7,(R2)      ;;LAST ONE?
(1)  025150  101371                          BHI     4$                 ;;NO--TRY IT
(1)  025152  011202                  $KTOUT: MOV     (R2),R2            ;;GET LAST BANK+1
(1)  025154  005037  177572                  CLR     @#SR0              ;;TURN OFF MEMORY MANAGEMENT
(1)  025160  000421                          BR      $SIZEX
(1)  025162  042737  100000  025020  $KTNEX: BIC     #100000,$KT11      ;;KT11 NON-EXISTENT
(1)  025170  012737  025220  000004  $CORE:  MOV     #$CROUT,@#ERRVEC   ;;SET FOR TIMEOUT
(1)  025176  005002                          CLR     R2                 ;;SET UP BANK
(1)  025200  062701  004000          1$:     ADD     #4000,R1           ;;INCREMENT BY 1K
(1)  025204  062702  000040                  ADD     #40,R2             ;;1K STEP
(1)  025210  005711                          TST     (R1)               ;;TRAP ON TIME OUT
(1)  025212  022701  177776                  CMP     #177776,R1         ;;LAST ONE
(1)  025216  001370                          BNE     1$                 ;;NO--TRY AGAIN
(1)  025220  162701  004000  $CROUT: SUB     #4000,R1
(1)  025224  162702  000040  $SIZEX: SUB     #40,R2             ;;DROP BACK
(1)  025230  010006                          MOV     R0,SP              ;;RESTORE THE STACK
(1)  025232  012637  000006                  MOV     (SP)+,@#ERRVEC+2   ;;RESTORE ERROR VECTOR
(1)  025236  012637  000004                  MOV     (SP)+,@#ERRVEC
(1)  025242  010137  025264                  MOV     R1,$LSTAD          ;;LAST ADDRESS
(1)  025246  010237  025266                  MOV     R2,$LSTBK          ;;LAST BANK
(1)  025252  012603                          MOV     (SP)+,R3           ;;RESTORE R3
(1)  025254  012602                          MOV     (SP)+,R2           ;;RESTORE R2
(1)  025256  012601                          MOV     (SP)+,R1           ;;RESTORE R1
(1)  025260  012600                          MOV     (SP)+,R0           ;;RESTORE R0
(1)  025262  000207                          RTS     PC
(1)  025264  000000                  $LSTAD: .WORD   0                  ;;CONTAINS THE LAST ADDRESS
(1)  025266  000000                  $LSTBK: .WORD   0                  ;;CONTAINS THE LAST BANK
2453                                 .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
(1)
(2)                                  ;;**********************************************************************
(1)                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
(1)                                  ;*CHANGE IT TO BINARY.
(1)                                  ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
(1)                                  ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
(1)                                  ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
(1)                                  ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
(1)                                  ;*CALL:
(1)                                  ;*      RDOCT                      ;;READ AN OCTAL NUMBER
(1)                                  ;*      RETURN HERE                ;;LOW ORDER BITS ARE ON TOP OF THE STACK
(1)                                  ;*                                 ;;HIGH ORDER BITS ARE IN $HIOCT
(1)
(1)  025270  011646                  $RDOCT: MOV     (SP),-(SP)         ;;PROVIDE SPACE FOR THE
(1)  025272  016666  000004  000002          MOV     4(SP),2(SP)        ;;INPUT NUMBER
(3)  025300  010046                          MOV     R0,-(SP)           ;;PUSH R0 ON STACK
(3)  025302  010146                          MOV     R1,-(SP)           ;;PUSH R1 ON STACK
(3)  025304  010246                          MOV     R2,-(SP)           ;;PUSH R2 ON STACK
(1)  025306  104411                  1$:     RDLIN                      ;;READ AN ASCIZ LINE
(1)  025310  012600                          MOV     (SP)+,R0           ;;GET ADDRESS OF 1ST CHARACTER
(1)  025312  010037  025416                  MOV     R0,5$              ;;AND SAVE IT
```

```
(1)  025316  005001                        CLR     R1              ;;CLEAR DATA WORD
(1)  025320  005002                        CLR     R2
(1)  025322  112046                2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
(1)  025324  001420                        BEQ     3$              ;;IF ZERO GET OUT
(1)  025326  122716  000060                CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
(1)  025332  003026                        BGT     4$              ;;IS AN OCTAL DIGIT
(1)  025334  122716  000067                CMPB    #'7,(SP)
(1)  025340  002423                        BLT     4$
(1)  025342  006301                        ASL     R1              ;;*2
(1)  025344  006102                        ROL     R2
(1)  025346  006301                        ASL     R1              ;;*4
(1)  025350  006102                        ROL     R2
(1)  025352  006301                        ASL     R1              ;;*8
(1)  025354  006102                        ROL     R2
(1)  025356  042716  177770                BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK.
(1)  025362  062601                        ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
(1)  025364  000756                        BR      2$              ;;LOOP
(1)  025366  005726                3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
(1)  025370  010166  000012                MOV     R1,12(SP)       ;;SAVE THE RESULT
(1)  025374  010237  025426                MOV     R2,$HIOCT
(3)  025400  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  025402  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  025404  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  025406  000002                        RTI                     ;;RETURN
(1)  025410  005726                4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
(1)  025412  105010                        CLRB    (R0)            ;;SET A TERMINATOR
(1)  025414  104401                        TYPE                    ;;TYPE UP THRU THE BAD CHAR.
(1)  025416  000000                5$:     .WORD   0
(1)  025420  104401  001224                TYPE    ,$QUES          ;;''?'' ''CR'' & ''LF''
(1)  025424  000730                        BR      1$              ;;TRY AGAIN
(1)  025426  000000                $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
2454                               .SBTTL  TTY INPUT ROUTINE
 (1)
 (2)                               ;;**********************************************************************
 (1)                               .ENABL  LSB
 (1)
 (2)                               ;;**********************************************************************
 (1)                               ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 (1)                               ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
 (1)                               ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
 (1)                               ;*WHEN OPERATING IN TTY FLAG MODE.
(1)  025430  022737  000176  001140 $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
(1)  025436  001074                        BNE     15$             ;;BRANCH IF NO
(1)  025440  105777  153500                TSTB    @$TKS           ;;CHAR THERE?
(1)  025444  100071                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
(1)  025446  117746  153474                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
(1)  025452  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
(1)  025456  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
(1)  025462  001062                        BNE     15$             ;;NO, RETURN TO USER
(1)  025464  123727  001134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
(1)  025472  001456                        BEQ     15$             ;;BRANCH IF YES
(1)
(1)  025474  104401  026307                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
(1)  025500  104401  026314        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
(2)  025504  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
(2)  025510  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
(1)  025512  104401  026325          TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
(1)  025516  005046          19$:   CLR     -(SP)           ;;CLEAR COUNTER
(1)  025520  005046                 CLR     -(SP)           ;;THE NEW SWR
(1)  025522  105777  153416    7$:   TSTB    @$TKS           ;;CHAR THERE?
(1)  025526  100375                 BPL     7$              ;;IF NOT TRY AGAIN
(1)
(1)  025530  117746  153412         MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
(1)  025534  042716  177600         BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)
(1)
(1)
(1)  025540  021627  000025    9$:   CMP     (SP),#25        ;;IS IT A CONTROL-U?
(1)  025544  001005                 BNE     10$             ;;BRANCH IF NOT
(1)  025546  104401  026302         TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
(1)  025552  062706  000006    20$:  ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
(1)  025556  000757                 BR      19$             ;;LET'S TRY IT AGAIN
(1)
(1)
(1)  025560  021627  000015    10$:  CMP     (SP),#15        ;;IS IT A <CR>?
(1)  025564  001022                 BNE     16$             ;;BRANCH IF NO
(1)  025566  005766  000004         TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
(1)  025572  001403                 BEQ     11$             ;;BRANCH IF YES
(1)  025574  016677  000002  153336 MOV     2(SP),@SWR      ;;SAVE NEW SWR
(1)  025602  062706  000006    11$:  ADD     #6,SP           ;;CLEAR UP STACK
(1)  025606  104401  001225    14$:  TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
(1)  025612  123727  001135  000001 CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
(1)  025620  001003                 BNE     15$             ;;BRANCH IF NOT
(1)  025622  012777  000100  153314 MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  025630  000002          15$:   RTI                     ;;RETURN
(1)  025632  004737  026550    16$:  JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  025636  021627  000060         CMP     (SP),#60        ;;CHAR < 0?
(1)  025642  002420                 BLT     18$             ;;BRANCH IF YES
(1)  025644  021627  000067         CMP     (SP),#67        ;;CHAR > 7?
(1)  025650  003015                 BGT     18$             ;;BRANCH IF YES
(1)  025652  042726  000060         BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  025656  005766  000002         TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  025662  001403                 BEQ     17$             ;;BRANCH IF YES
(1)  025664  006316                 ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  025666  006316                 ASL     (SP)            ;;    CHAR OVER TO MAKE
(1)  025670  006316                 ASL     (SP)            ;;    ROOM FOR NEW ONE.
(1)  025672  005266  000002    17$:  INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  025676  056616  177776         BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  025702  000707                 BR      7$              ;;GET THE NEXT ONE
(1)  025704  104401  001224    18$:  TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  025710  000720                 BR      20$             ;;SIMULATE CONTROL-U
(1)                         .DSABL  LSB
(1)
(1)
(2)                         ;;******************************************************************
(1)                         ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1)                         ;*CALL:
(1)                         ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1)                         ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
(1)                         ;*                              ;;WITH PARITY BIT STRIPPED OFF
(1)                         ;
(1)
```

```
(1)  025712  011646                        $RDCHR: MOV     (SP),-(SP)          ;;PUSH DOWN THE PC
(1)  025714  016666  000004  000002                MOV     4(SP),2(SP)         ;;SAVE THE PS
(1)  025722  105777  153216                1$:     TSTB    @$TKS               ;;WAIT FOR
(1)  025726  100375                                BPL     1$                  ;;A CHARACTER
(1)  025730  117766  153212  000004                MOVB    @$TKB,4(SP)         ;;READ THE TTY
(1)  025736  042766  177600  000004                BIC     #^C<177>,4(SP)      ;;GET RID OF JUNK IF ANY
(1)  025744  026627  000004  000023                CMP     4(SP),#23           ;;IS IT A CONTROL-S?
(1)  025752  001013                                BNE     3$                  ;;BRANCH IF NO
(1)  025754  105777  153164                2$:     TSTB    @$TKS               ;;WAIT FOR A CHARACTER
(1)  025760  100375                                BPL     2$                  ;;LOOP UNTIL ITS THERE
(1)  025762  117746  153160                        MOVB    @$TKB,-(SP)         ;;GET CHARACTER
(1)  025766  042716  177600                        BIC     #^C177,(SP)         ;;MAKE IT 7-BIT ASCII
(1)  025772  022627  000021                        CMP     (SP)+,#21           ;;IS IT A CONTROL-Q?
(1)  025776  001366                                BNE     2$                  ;;IF NOT DISCARD IT
(1)  026000  000750                                BR      1$                  ;;YES, RESUME
(1)  026002  026627  000004  000140  3$:     CMP     4(SP),#140          ;;IS IT UPPER CASE?
(1)  026010  002407                                BLT     4$                  ;;BRANCH IF YES
(1)  026012  026627  000004  000175                CMP     4(SP),#175          ;;IS IT A SPECIAL CHAR?
(1)  026020  003003                                BGT     4$                  ;;BRANCH IF YES
(1)  026022  042766  000040  000004                BIC     #40,4(SP)           ;;MAKE IT UPPER CASE
(1)  026030  000002                        4$:     RTI                         ;;GO BACK TO USER
(2)                                        ;;******************************************************************
(1)                                        ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1)                                        ;*CALL:
(1)                                        ;*      RDLIN                       ;;INPUT A STRING FROM THE TTY
(1)                                        ;*      RETURN HERE                 ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)                                        ;*                                  ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
(1)
(1)  026032  010346                        $RDLIN: MOV     R3,-(SP)            ;;SAVE R3
(1)  026034  005046                                CLR     -(SP)               ;;CLEAR THE RUBOUT KEY
(1)  026036  012703  026266                1$:     MOV     #$TTYIN,R3          ;;GET ADDRESS
(1)  026042  022703  026302                2$:     CMP     #$TTYIN+12.,R3      ;;BUFFER FULL?
(1)  026046  101456                                BLOS    4$                  ;;BR IF YES
(1)  026050  104410                                RDCHR                       ;;GO READ ONE CHARACTER FROM THE TTY
(1)  026052  112613                                MOVB    (SP)+,(R3)          ;;GET CHARACTER
(1)  026054  122713  000177                10$:    CMPB    #177,(R3)           ;;IS IT A RUBOUT
(1)  026060  001022                                BNE     5$                  ;;BR IF NO
(1)  026062  005716                                TST     (SP)                ;;IS THIS THE FIRST RUBOUT?
(1)  026064  001007                                BNE     6$                  ;;BR IF NO
(1)  026066  112737  000134  026264                MOVB    #'\,9$              ;;TYPE A BACK SLASH
(1)  026074  104401  026264                        TYPE    ,9$
(1)  026100  012716  177777                        MOV     #-1,(SP)            ;;SET THE RUBOUT KEY
(1)  026104  005303                        6$:     DEC     R3                  ;;BACKUP BY ONE
(1)  026106  020327  026266                        CMP     R3,#$TTYIN          ;;STACK EMPTY?
(1)  026112  103434                                BLO     4$                  ;;BR IF YES
(1)  026114  111337  026264                        MOVB    (R3),9$             ;;SETUP TO TYPEOUT THE DELETED CHAR.
(1)  026120  104401  026264                        TYPE    ,9$                 ;;GO TYPE
(1)  026124  000746                                BR      2$                  ;;GO READ ANOTHER CHAR.
(1)  026126  005716                        5$:     TST     (SP)                ;;RUBOUT KEY SET?
(1)  026130  001406                                BEQ     7$                  ;;BR IF NO
(1)  026132  112737  000134  026264                MOVB    #'\,9$              ;;TYPE A BACK SLASH
(1)  026140  104401  026264                        TYPE    ,9$
(1)  026144  005016                                CLR     (SP)                ;;CLEAR THE RUBOUT KEY
(1)  026146  122713  000025                7$:     CMPB    #25,(R3)            ;;IS CHARACTER A CTRL U?
(1)  026152  001003                                BNE     8$                  ;;BR IF NO
(1)  026154  104401  026302                        TYPE    ,$CNTLU             ;;TYPE A CONTROL 'U'
```

```
  (1)  026160  000726                          BR      1$              ;;GO START OVER
  (1)  026162  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A '^R'?
  (1)  026166  001011                          BNE     3$              ;;BRANCH IF NO
  (1)  026170  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
  (1)  026172  104401  001225                  TYPE    ,$CRLF          ;;TYPE A 'CR' & 'LF'
  (1)  026176  104401  026266                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
  (1)  026202  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
  (1)  026204  104401  001224          4$:     TYPE    ,$QUES          ;;TYPE A '?'
  (1)  026210  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
  (1)  026212  111337  026264          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
  (1)  026216  104401  026264                  TYPE    ,9$
  (1)  026222  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
  (1)  026226  001305                          BNE     2$              ;;LOOP IF NOT RETURN
  (1)  026230  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
  (1)  026234  104401  001226                  TYPE    ,$LF            ;;TYPE A LINE FEED
  (1)  026240  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
  (1)  026242  012603                          MOV     (SP)+,R3        ;;RESTORE R3
  (1)  026244  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
  (1)  026246  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
  (1)  026254  012766  026266  000004          MOV     #$TTYIN,4(SP)
  (1)  026262  000002                          RTI                     ;;RETURN
  (1)  026264     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
  (1)  026265     000                  .BYTE   0                       ;;TERMINATOR
  (1)  026266  000014          $TTYIN: .BLKB   12.                     ;;RESERVE 12. BYTES FOR TTY INPUT
  (1)  026302  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>     ;;CONTROL 'U'
  (1)  026307     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>     ;;CONTROL 'G'
  (1)  026314  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
  (1)  026322  020075     000
  (1)  026325     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
  (1)  026332  036440  000040
 2455                                  .SBTTL  TYPE ROUTINE
  (1)
  (2)                                  ;;**********************************************************************
  (1)                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
  (1)                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
  (1)                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
  (1)                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
  (1)                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
  (1)                                  ;*
  (1)                                  ;*CALL:
  (1)                                  ;*1) USING A TRAP INSTRUCTION
  (1)                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
  (1)                                  ;*OR
  (1)                                  ;*      TYPE
  (1)                                  ;*      MESADR
  (1)                                  ;*
  (1)
  (1)  026336  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
  (1)  026342  100002                          BPL     1$              ;;BR IF YES
  (1)  026344  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
  (1)  026346  000430                          BR      3$              ;;LEAVE
  (1)  026350  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
  (1)  026352  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
  (1)  026356  122737  000001  001250          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
  (1)  026364  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
  (1)  026366  132737  000100  001251          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
```

```
(1)  026374  001405                        BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(1)  026376  010037  026406                MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(1)  026402  004737  027540                JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(1)  026406  000000                61$:    .WORD   0               ;;MESSAGE ADDRESS
(1)  026410  132737  000040  001251  62$:  BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(1)  026416  001003                        BNE     60$             ;;YES,SKIP TYPE OUT
(1)  026420  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  026422  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  026424  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  026426  012600                60$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  026430  062716  000002                3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  026434  000002                        RTI                     ;;RETURN
(1)  026436  122716  000011                4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  026442  001430                        BEQ     8$
(1)  026444  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  026450  001006                        BNE     5$
(1)  026452  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  026454  104401                        TYPE                    ;;TYPE A CR AND LF
(1)  026456  001225                        $CRLF
(1)  026460  105037  026614                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  026464  000755                        BR      2$              ;;GET NEXT CHARACTER
(1)  026466  004737  026550                5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  026472  123726  001156                6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  026476  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  026500  013746  001154                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                                                ;;AND THE NULL CHAR.
(1)  026504  105366  000001                7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  026510  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  026512  004737  026550                JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  026516  105337  026614                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  026522  000770                        BR      7$              ;;LOOP
(1)
(1)                                        ;HORIZONTAL TAB PROCESSOR
(1)
(1)  026524  112716  000040                8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  026530  004737  026550                9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  026534  132737  000007  026614                BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  026542  001372                        BNE     9$              ;;TAB STOP
(1)  026544  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
(1)  026546  000724                        BR      2$              ;;GET NEXT CHARACTER
(1)  026550  105777  152374                $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  026554  100375                        BPL     $TYPEC
(1)  026556  116677  000002  152366        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  026564  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  026572  001003                        BNE     1$              ;;BRANCH IF NO
(1)  026574  105037  026614                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  026600  000406                        BR      $TYPEX          ;;EXIT
(1)  026602  122766  000012  000002        1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  026610  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
(1)  026612  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  026614  000000                $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(1)  026616  000207                $TYPEX: RTS     PC
(1)
2456                                       .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(2)                                        ;;**********************************************************************
```

```
(1)                                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1)                                     ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(1)                                     ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPOS                   ;;CALL FOR TYPEOUT
(1)                                     ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1)                                     ;*      .BYTE   M               ;;M=1 OR 0
(1)                                     ;*                                      ;;1=TYPE LEADING ZEROS
(1)                                     ;*                                      ;;0=SUPPRESS LEADING ZEROS
(1)                                     ;*
(1)                                     ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1)                                     ;*$TYPOS OR $TYPOC
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPON                   ;;CALL FOR TYPEOUT
(1)                                     ;*
(1)                                     ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPOC                   ;;CALL FOR TYPEOUT
(1)
(1)  026620  017646  000000             $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  026624  116637  000001  027043             MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  026632  112637  027045             $TYPOC: MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  026636  062716  000002                     ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  026642  000406                             BR      $TYPON
(1)  026644  112737  000001  027043             MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
(1)  026652  112737  000006  027045             MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  026660  112737  000005  027042     $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(1)  026666  010346                             MOV     R3,-(SP)        ;;SAVE R3
(1)  026670  010446                             MOV     R4,-(SP)        ;;SAVE R4
(1)  026672  010546                             MOV     R5,-(SP)        ;;SAVE R5
(1)  026674  113704  027045                     MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  026700  005404                             NEG     R4
(1)  026702  062704  000006                     ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  026706  110437  027044                     MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  026712  113704  027043                     MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  026716  016605  000012                     MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  026722  005003                             CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  026724  006105                     1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  026726  000404                             BR      3$              ;;GO DO MSB
(1)  026730  006105                     2$:     ROL     R5              ;;FORM THIS DIGIT
(1)  026732  006105                             ROL     R5
(1)  026734  006105                             ROL     R5
(1)  026736  010503                             MOV     R5,R3
(1)  026740  006103                     3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  026742  105337  027044                     DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  026746  100016                             BPL     7$              ;;BR IF NO
(1)  026750  042703  177770                     BIC     #177770,R3      ;;GET RID OF JUNK
(1)  026754  001002                             BNE     4$              ;;TEST FOR 0
(1)  026756  005704                             TST     R4              ;;SUPPRESS THIS 0?
(1)  026760  001403                             BEQ     5$              ;;BR IF YES
(1)  026762  005204                     4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  026764  052703  000060                     BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  026770  052703  000040             5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
```

```
(1)  026774  110337  027040            MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  027000  104401  027040            TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  027004  105337  027042    7$:     DECB    $OCNT           ;;COUNT BY 1
(1)  027010  003347                    BGT     2$              ;;BR IF MORE TO DO
(1)  027012  002402                    BLT     6$              ;;BR IF DONE
(1)  027014  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  027016  000744                    BR      2$              ;;GO DO THE LAST DIGIT
(1)  027020  012605            6$:     MOV     (SP)+,R5        ;;RESTORE R5
(1)  027022  012604                    MOV     (SP)+,R4        ;;RESTORE R4
(1)  027024  012603                    MOV     (SP)+,R3        ;;RESTORE R3
(1)  027026  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  027034  012616                    MOV     (SP)+,(SP)
(1)  027036  000002                    RTI                     ;;RETURN
(1)  027040     000    8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  027041     000            .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  027042     000    $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(1)  027043     000    $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(1)  027044  000000    $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2457                   .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
(1)
(2)                    ;;**************************************************************
(1)                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
(1)                    ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
(1)                    ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
(1)                    ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
(1)                    ;*REPLACED WITH SPACES.
(1)                    ;*CALL:
(1)                    ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
(1)                    ;*      TYPDS                   ;;GO TO THE ROUTINE
(1)
(1)  027046            $TYPDS:
(3)  027046  010046            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  027050  010146            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  027052  010246            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  027054  010346            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  027056  010546            MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  027060  012746  020200    MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)  027064  016605  000020    MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)  027070  100004            BPL     1$              ;;BR IF INPUT IS POS.
(1)  027072  005405            NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)  027074  112766  000055  000001    MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  027102  005000    1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  027104  012703  027262            MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  027110  112723  000040            MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  027114  005002    2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  027116  016001  027252            MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  027122  160105    3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  027124  002402            BLT     4$              ;;BR IF DONE
(1)  027126  005202            INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  027130  000774            BR      3$
(1)  027132  060105    4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  027134  005702            TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  027136  001002            BNE     5$              ;;FALL THROUGH IF 0
(1)  027140  105716            TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)  027142  100407            BMI     7$              ;;BR IF YES
(1)  027144  106316    5$:     ASLB    (SP)            ;;MSD?
```

```
(1)  027146  103003                        BCC     6$                  ;;BR IF NO
(1)  027150  116663  000001  177777        MOVB    1(SP),-1(R3)        ;;YES--SET THE SIGN
(1)  027156  052702  000060          6$:   BIS     #'0,R2              ;;MAKE THE BCD DIGIT ASCII
(1)  027162  052702  000040          7$:   BIS     #' ,R2              ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  027166  110223                        MOVB    R2,(R3)+            ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  027170  005720                        TST     (R0)+               ;;JUST INCREMENTING
(1)  027172  020027  000010                CMP     R0,#10              ;;CHECK THE TABLE INDEX
(1)  027176  002746                        BLT     2$                  ;;GO DO THE NEXT DIGIT
(1)  027200  003002                        BGT     8$                  ;;GO TO EXIT
(1)  027202  010502                        MOV     R5,R2               ;;GET THE LSD
(1)  027204  000764                        BR      6$                  ;;GO CHANGE TO ASCII
(1)  027206  105726                  8$:   TSTB    (SP)+               ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  027210  100003                        BPL     9$                  ;;BR IF NO
(1)  027212  116663  177777  177776        MOVB    -1(SP),-2(R3)       ;;YES--SET THE SIGN FOR TYPING
(1)  027220  105013                  9$:   CLRB    (R3)                ;;SET THE TERMINATOR
(3)  027222  012605                        MOV     (SP)+,R5            ;;POP STACK INTO R5
(3)  027224  012603                        MOV     (SP)+,R3            ;;POP STACK INTO R3
(3)  027226  012602                        MOV     (SP)+,R2            ;;POP STACK INTO R2
(3)  027230  012601                        MOV     (SP)+,R1            ;;POP STACK INTO R1
(3)  027232  012600                        MOV     (SP)+,R0            ;;POP STACK INTO R0
(1)  027234  104401  027262                TYPE    ,$DBLK              ;;NOW TYPE THE NUMBER
(1)  027240  016666  000002  000004        MOV     2(SP),4(SP)         ;;ADJUST THE STACK
(1)  027246  012616                        MOV     (SP)+,(SP)
(1)  027250  000002                        RTI                         ;;RETURN TO USER
(1)  027252  023420                $DTBL:  10000.
(1)  027254  001750                        1000.
(1)  027256  000144                        100.
(1)  027260  000012                        10.
(1)  027262  000004                $DBLK:  .BLKW   4
2458                               .SBTTL  TRAP DECODER
(1)
(2)                                ;;**********************************************************************
(1)                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
(1)                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1)                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1)                                ;*GO TO THAT ROUTINE.
(1)
(1)  027272  010046                $TRAP:  MOV     R0,-(SP)            ;;SAVE R0
(1)  027274  016600  000002                MOV     2(SP),R0            ;;GET TRAP ADDRESS
(1)  027300  005740                        TST     -(R0)               ;;BACKUP BY 2
(1)  027302  111000                        MOVB    (R0),R0             ;;GET RIGHT BYTE OF TRAP
(1)  027304  006300                        ASL     R0                  ;;POSITION FOR INDEXING
(1)  027306  016000  027326                MOV     $TRPAD(R0),R0       ;;INDEX TO TABLE
(1)  027312  000200                        RTS     R0                  ;;GO TO ROUTINE
(1)
(1)
(1)                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
(1)
(1)  027314  011646                $TRAP2: MOV     (SP),-(SP)          ;;MOVE THE PC DOWN
(1)  027316  016666  000004  000002        MOV     4(SP),2(SP)         ;;MOVE THE PSW DOWN
(1)  027324  000002                        RTI                         ;;RESTORE THE PSW
(1)
(3)                                .SBTTL  TRAP TABLE
(3)
(3)                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(3)                                ;*BY THE "TRAP" INSTRUCTION.
```

```
 (3)
 (3)                                      ;       ROUTINE
 (3)                                      ;       -------
 (3)   027326  027314                     $TRPAD: .WORD   $TRAP2
 (3)   027330  026336                             $TYPE   ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
 (3)   027332  026644                             $TYPOC  ;;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 (3)   027334  026620                             $TYPOS  ;;CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
 (3)   027336  026660                             $TYPON  ;;CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
 (3)   027340  027046                             $TYPDS  ;;CALL=TYPDS     TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
 (1)
 (3)   027342  025500                             $GTSWR  ;;CALL=GTSWR     TRAP+6(104406)  GET SOFT-SWR SETTING
 (1)
 (3)   027344  025430                             $CKSWR  ;;CALL=CKSWR     TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
 (3)   027346  025712                             $RDCHR  ;;CALL=RDCHR     TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
 (3)   027350  026032                             $RDLIN  ;;CALL=RDLIN     TRAP+11(104411) TTY TYPEIN STRING ROUTINE
 (3)   027352  025270                             $RDOCT  ;;CALL=RDOCT     TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
2459                                      .SBTTL  POWER DOWN AND UP ROUTINES
 (1)
 (2)                                      ;;*****************************************************************
 (1)                                      ;POWER DOWN ROUTINE
 (1)   027354  012737  027514  000024     $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
 (1)   027362  012737  000340  000026             MOV     #340,@#PWRVEC+2  ;;PRIO:7
 (3)   027370  010046                             MOV     R0,-(SP)         ;;PUSH R0 ON STACK
 (3)   027372  010146                             MOV     R1,-(SP)         ;;PUSH R1 ON STACK
 (3)   027374  010246                             MOV     R2,-(SP)         ;;PUSH R2 ON STACK
 (3)   027376  010346                             MOV     R3,-(SP)         ;;PUSH R3 ON STACK
 (3)   027400  010446                             MOV     R4,-(SP)         ;;PUSH R4 ON STACK
 (3)   027402  010546                             MOV     R5,-(SP)         ;;PUSH R5 ON STACK
 (3)   027404  017746  151530                     MOV     @SWR,-(SP)       ;;PUSH @SWR ON STACK
 (1)   027410  010637  027520                     MOV     SP,$SAVR6        ;;SAVE SP
 (1)   027414  012737  027426  000024             MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
 (1)   027422  000000                             HALT
 (1)   027424  000776                             BR      .-2              ;;HANG UP
 (1)
 (2)                                      ;;*****************************************************************
 (1)                                      ;POWER UP ROUTINE
 (1)   027426  012737  027514  000024     $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
 (1)   027434  013706  027520                     MOV     $SAVR6,SP        ;;GET SP
 (1)   027440  005037  027520                     CLR     $SAVR6           ;;WAIT LOOP FOR THE TTY
 (1)   027444  005237  027520             1$:     INC     $SAVR6           ;;WAIT FOR THE INC
 (1)   027450  001375                             BNE     1$               ;;OF  WORD
 (3)   027452  012677  151462                     MOV     (SP)+,@SWR       ;;POP STACK INTO @SWR
 (3)   027456  012605                             MOV     (SP)+,R5         ;;POP STACK INTO R5
 (3)   027460  012604                             MOV     (SP)+,R4         ;;POP STACK INTO R4
 (3)   027462  012603                             MOV     (SP)+,R3         ;;POP STACK INTO R3
 (3)   027464  012602                             MOV     (SP)+,R2         ;;POP STACK INTO R2
 (3)   027466  012601                             MOV     (SP)+,R1         ;;POP STACK INTO R1
 (3)   027470  012600                             MOV     (SP)+,R0         ;;POP STACK INTO R0
 (1)   027472  012737  027354  000024             MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
 (1)   027500  012737  000340  000026             MOV     #340,@#PWRVEC+2  ;;PRIO:7
 (1)   027506  104401                             TYPE                     ;REPORT THE POWER FAILURE
 (1)   027510  027522                     $PWRMG: .WORD   $POWER           ;;POWER FAIL MESSAGE POINTER
 (1)   027512  000002                             RTI
 (1)   027514  000000                     $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
 (1)   027516  000776                             BR      .-2              ;; BEFORE THE POWER DOWN WAS COMPLETE
 (1)   027520  000000                     $SAVR6: 0                        ;;PUT THE SP HERE
```

```
   (1)  027522  005015  047520  042527  $POWER: .ASCIZ  <15><12>'POWER'
   (1)  027530  000122
   (1)                                          .EVEN
  2460                                  .SBTTL  APT COMMUNICATIONS ROUTINE
   (1)
   (2)                                  ;;*****************************************************************
   (1)  027532  112737  000001  027776  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
   (1)  027540  112737  000001  027774  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
   (1)  027546  000403                          BR      $ATYC
   (1)  027550  112737  000001  027776  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
   (1)  027556                          $ATYC:
   (3)  027556  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
   (3)  027560  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
   (1)  027562  105737  027774                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
   (1)  027566  001450                          BEQ     5$              ;;IF NOT:  BR
   (1)  027570  122737  000001  001250          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
   (1)  027576  001031                          BNE     3$              ;;IF NOT:  BR
   (1)  027600  132737  000100  001251          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
   (1)  027606  001425                          BEQ     3$              ;;IF NOT:  BR
   (1)  027610  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
   (1)  027614  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
   (1)  027622  005737  001230          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
   (1)  027626  001375                          BNE     1$              ;;IF NOT:  WAIT
   (1)  027630  010037  001244                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
   (1)  027634  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
   (1)  027636  001376                          BNE     2$
   (1)  027640  163700  001244                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
   (1)  027644  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
   (1)  027646  010037  001246                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
   (1)  027652  012737  000004  001230          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
   (1)  027660  000413                          BR      5$
   (1)  027662  017637  000004  027706  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
   (1)  027670  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
   (3)  027676  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
   (1)  027702  004737  026336                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
   (1)  027706  000000                  4$:     .WORD   0
   (1)  027710                          5$:
   (1)  027710  105737  027776          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
   (1)  027714  001416                          BEQ     12$             ;;IF NOT:  BR
   (1)  027716  005737  001250                  TST     $ENV            ;;RUNNING UNDER APT?
   (1)  027722  001413                          BEQ     12$             ;;IF NOT:  BR
   (1)  027724  005737  001230          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
   (1)  027730  001375                          BNE     11$             ;;IF NOT:  WAIT
   (1)  027732  017637  000004  001232          MOV     @4(SP),$FATAL   ;;GET ERROR #
   (1)  027740  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
   (1)  027746  005237  001230                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
   (1)  027752  105037  027776          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
   (1)  027756  105037  027775                  CLRB    $LFLG           ;;CLEAR LOG FLAG
   (1)  027762  105037  027774                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
   (3)  027766  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
   (3)  027770  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
   (1)  027772  000207                          RTS     PC              ;;RETURN
   (1)  027774     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
   (1)  027775     000          $LFLG:  .BYTE   0               ;;LOG FLAG
   (1)  027776     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
   (1)          030000                  .EVEN
```

```
(1)  000200  APTSIZE=200
(1)  000001  APTENV=001
(1)  000100  APTSPOOL=100
(1)  000040  APTCSUP=040
```

```
                                                .SBTTL  MESSAGES

030000 041501 020122 044502  EM1:    .ASCIZ          /ACR BIT3-0 LOAD-READ ERROR/
030006 031524 030055 046040
030014 040517 026504 042522
030022 042101 042440 051122
030030 051117    000
030033    101 051103 041040  EM2:    .ASCIZ          /ACR BIT3-0 WERE NOT CLEARED BY RESET (ACR-CLR)/
030040 052111 026463 020060
030046 042527 042522 047040
030054 052117 041440 042514
030062 051101 042105 041040
030070 020131 042522 042523
030076 020124 040450 051103
030104 041455 051114 000051
030112 042511 024040 043520  EM7:    .ASCIZ          /IE (PGTE BIT2) CANNOT BE SET/
030120 042524 041040 052111
030126 024462 041440 047101
030134 047516 020124 042502
030142 051440 052105    000
030147    111 020105 050050  EM10:   .ASCIZ          /IE (PGTE BIT2) CANNOT BE WRITTEN TO 0/
030154 052107 020105 044502
030162 031124 020051 040503
030170 047116 052117 041040
030176 020105 051127 052111
030204 042524 020116 047524
030212 030040    000
030215    111 020105 050050  EM11:   .ASCIZ          /IE (PGTE BIT2) CANNOT BE CLEARED BY RESET (ACR CLR)/
030222 052107 020105 044502
030230 031124 020051 040503
030236 047116 052117 041040
030244 020105 046103 040505
030252 042522 020104 054502
030260 051040 051505 052105
030266 024040 041501 020122
030274 046103 024522    000
030301    120 050124 024040  EM12:   .ASCIZ          /PTP (PGTE BIT1) CANNOT BE SET/
030306 043520 042524 041040
030314 052111 024461 041440
030322 047101 047516 020124
030330 042502 051440 052105
030336    000
030337    120 050124 024040  EM13:   .ASCIZ          /PTP (PGTE BIT1) CANNOT BE WRITTEN TO 0/
030344 043520 042524 041040
030352 052111 024461 041440
030360 047101 047516 020124
030366 042502 053440 044522
030374 052124 047105 052040
030402 020117 000060
030406 052120 020120 050050  EM14:   .ASCIZ          /PTP (PGTE BIT1) CANNOT BE CLEARED BY RESET (ACR CLR)/
030414 052107 020105 044502
030422 030524 020051 040503
030430 047116 052117 041040
030436 020105 046103 040505
030444 042522 020104 054502
030452 051040 051505 052105
```

```
                030460  024040  041501  020122
                030466  046103  024522     000
2472            030473     120  052107  020105  EM3:    .ASCIZ          /PGTE BIT11-8 LOAD-READ ERROR/
                030500  044502  030524  026461
                030506  020070  047514  042101
                030514  051055  040505  020104
                030522  051105  047522  000122
2473            030530  043520  042524  041040  EM4:    .ASCIZ          /PGTE BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)/
                030536  052111  030461  034055
                030544  053440  051105  020105
                030552  047516  020124  046103
                030560  040505  042522  020104
                030566  054502  051040  051505
                030574  052105  024040  041501
                030602  020122  046103  024522
                030610     000
2474            030611     120  052107  020105  EM5:    .ASCIZ          /PGTE BIT3-0 LOAD-READ ERROR/
                030616  044502  031524  030055
                030624  046040  040517  026504
                030632  042522  042101  042440
                030640  051122  051117     000
2475            030645     120  052107  020105  EM6:    .ASCIZ          /PGTE BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)/
                030652  044502  031524  030055
                030660  053440  051105  020105
                030666  047516  020124  046103
                030674  040505  042522  020104
                030702  054502  051040  051505
                030710  052105  024040  041501
                030716  020122  046103  024522
                030724     000
2476            030725     123  052124  020105  EM15:   .ASCIZ          /STTE BIT11-8 LOAD-READ ERROR/
                030732  044502  030524  026461
                030740  020070  047514  042101
                030746  051055  040505  020104
                030754  051105  047522  000122
2477            030762  052123  042524  041040  EM16:   .ASCIZ          /STTE BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)/
                030770  052111  030461  034055
                030776  053440  051105  020105
                031004  047516  020124  046103
                031012  040505  042522  020104
                031020  054502  051040  051505
                031026  052105  024040  041501
                031034  020122  046103  024522
                031042     000
2478            031043     123  052124  020105  EM17:   .ASCIZ          /STTE BIT3-0 LOAD-READ ERROR/
                031050  044502  031524  030055
                031056  046040  040517  026504
                031064  042522  042101  042440
                031072  051122  051117     000
2479            031077     123  052124  020105  EM20:   .ASCIZ          /STTE BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)/
                031104  044502  031524  030055
                031112  053440  051105  020105
                031120  047516  020124  046103
                031126  040505  042522  020104
                031134  054502  051040  051505
                031142  052105  024040  041501
```

```
                031150  020122  046103  024522
                031156     000
2480            031157     114  042513  024040  EM21:   .ASCIZ       /LKE (STCS BIT2) CANNOT BE SET/
                031164  052123  051503  041040
                031172  052111  024462  041440
                031200  047101  047516  020124
                031206  042502  051440  052105
                031214     000
2481            031215     114  042513  024040  EM22:   .ASCIZ       /LKE (STCS BIT2) CANNOT BE WRITTEN TO 0/
                031222  052123  051503  041040
                031230  052111  024462  041440
                031236  047101  047516  020124
                031244  042502  053440  044522
                031252  052124  047105  052040
                031260  020117  000060
2482            031264  045514  020105  051450  EM23:   .ASCIZ       /LKE (STCS BIT2) CANNOT BE CLEARED BY RESET (ACR CLR)/
                031272  041524  020123  044502
                031300  031124  020051  040503
                031306  047116  052117  041040
                031314  020105  046103  040505
                031322  042522  020104  054502
                031330  051040  051505  052105
                031336  024040  041501  020122
                031344  046103  024522     000
2483            031351     123  050124  024040  EM24:   .ASCIZ       /STP (STCS BIT1) CANNOT BE SET/
                031356  052123  051503  041040
                031364  052111  024461  041440
                031372  047101  047516  020124
                031400  042502  051440  052105
                031406     000
2484            031407     123  050124  024040  EM25:   .ASCIZ       /STP (STCS BIT1) CANNOT BE WRITTEN TO 0/
                031414  052123  051503  041040
                031422  052111  024461  041440
                031430  047101  047516  020124
                031436  042502  053440  044522
                031444  052124  047105  052040
                031452  020117  000060
2485            031456  052123  020120  051450  EM26:   .ASCIZ       /STP (STCS BIT1) CANNOT BE CLEARED BY RESET (ACR CLR)/
                031464  041524  020123  044502
                031472  030524  020051  040503
                031500  047116  052117  041040
                031506  020105  046103  040505
                031514  042522  020104  054502
                031522  051040  051505  052105
                031530  024040  041501  020122
                031536  046103  024522     000
2486            031543     105  041116  024040  EM27:   .ASCIZ       /ENB (STCS BIT0) CANNOT BE SET/
                031550  052123  051503  041040
                031556  052111  024460  041440
                031564  047101  047516  020124
                031572  042502  051440  052105
                031600     000
2487            031601     105  041116  024040  EM30:   .ASCIZ       /ENB (STCS BIT0) CANNOT BE WRITTEN TO 0/
                031606  052123  051503  041040
                031614  052111  024460  041440
                031622  047101  047516  020124
```

```
                031630  042502  053440  044522
                031636  052124  047105  052040
                031644  020117  000060
2488            031650  047105  020102  051450  EM31:   .ASCIZ       /ENB (STCS BIT0) CANNOT BE CLEARED BY RESET (ACR CLR)/
                031656  041524  020123  044502
                031664  030124  020051  040503
                031672  047116  052117  041040
                031700  020105  046103  040505
                031706  042522  020104  054502
                031714  051040  051505  052105
                031722  024040  041501  020122
                031730  046103  024522     000
2489            031735     123  041524  020123  EM32:   .ASCIZ       /STCS BIT15-8 LOAD-READ ERROR/
                031742  044502  030524  026465
                031750  020070  047514  042101
                031756  051055  040505  020104
                031764  051105  047522  000122
2490            031772  052123  051503  041040  EM33:   .ASCIZ       /STCS BIT15-8 WERE NOT CLEARED BY RESET (ACR CLR)/
                032000  052111  032461  034055
                032006  053440  051105  020105
                032014  047516  020124  046103
                032022  040505  042522  020104
                032030  054502  051040  051505
                032036  052105  024040  041501
                032044  020122  046103  024522
                032052     000
2491            032053     111  051515  020113  EM34:   .ASCIZ       /IMSK BIT11-8 LOAD-READ ERROR/
                032060  044502  030524  026461
                032066  020070  047514  042101
                032074  051055  040505  020104
                032102  051105  047522  000122
2492            032110  046511  045523  041040  EM35:   .ASCIZ       /IMSK BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)/
                032116  052111  030461  034055
                032124  053440  051105  020105
                032132  047516  020124  046103
                032140  040505  042522  020104
                032146  054502  051040  051505
                032154  052105  024040  041501
                032162  020122  046103  024522
                032170     000
2493            032171     111  051515  020113  EM36:   .ASCIZ       /IMSK BIT3-0 LOAD-READ ERROR/
                032176  044502  031524  030055
                032204  046040  040517  026504
                032212  042522  042101  042440
                032220  051122  051117     000
2494            032225     111  051515  020113  EM37:   .ASCIZ       /IMSK BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)/
                032232  044502  031524  030055
                032240  053440  051105  020105
                032246  047516  020124  046103
                032254  040505  042522  020104
                032262  054502  051040  051505
                032270  052105  024040  041501
                032276  020122  046103  024522
                032304     000
2495            032305     115  041524  041040  EM40:   .ASCIZ       /MTC BIT11-8 LOAD-READ ERROR/
                032312  052111  030461  034055
```

```
                032320  046040  040517  026504
                032326  042522  042101  042440
                032334  051122  051117     000
 2496   032341     115  041524  041040  EM41:   .ASCIZ       /MTC BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)/
                032346  052111  030461  034055
                032354  053440  051105  020105
                032362  047516  020124  046103
                032370  040505  042522  020104
                032376  054502  051040  051505
                032404  052105  024040  041501
                032412  020122  046103  024522
                032420     000
 2497   032421     115  041524  041040  EM42:   .ASCIZ       /MTC BIT3-0 LOAD-READ ERROR/
                032426  052111  026463  020060
                032434  047514  042101  051055
                032442  040505  020104  051105
                032450  047522  000122
 2498   032454  052115  020103  044502  EM43:   .ASCIZ       /MTC BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)/
                032462  031524  030055  053440
                032470  051105  020105  047516
                032476  020124  046103  040505
                032504  042522  020104  054502
                032512  051040  051505  052105
                032520  024040  041501  020122
                032526  046103  024522     000
 2499   032533     111  051511  020124  EM44:   .ASCII       /IIST ERROR - REFER TO THE LISTING AT 'PC'/
                032540  051105  047522  020122
                032546  020055  042522  042506
                032554  020122  047524  052040
                032562  042510  046040  051511
                032570  044524  043516  040440
                032576  020124  050042  021103
 2500   032604  052200  020117  042504          .ASCIZ       <CRLF>/TO DETERMINE THE ERROR./
                032612  042524  046522  047111
                032620  020105  044124  020105
                032626  051105  047522  027122
                032634     000
```

```
2502  032635    120   051501  020123  DH1:   .ASCIZ    /PASS   PC   EXPCTD  ACTUAL  ACR/
      032642  020040  050040  020103
      032650  020040  020040  042440
      032656  050130  052103  020104
      032664  040440  052103  040525
      032672  020114  040440  051103
      032700     000
2503  032701    120   051501  020123  DH2:   .ASCIZ    /PASS   PC   ACR/
      032706  020040  050040  020103
      032714  020040  020040  040440
      032722  051103     000
2504  032725    120   051501  020123  DH3:   .ASCIZ    /PASS   PC   PGTE/
      032732  020040  050040  020103
      032740  020040  020040  050040
      032746  052107  000105
2505  032752  040520  051523  020040  DH4:   .ASCIZ    /PASS   PC   EXPCTD  ACTUAL  PGTE/
      032760  020040  041520  020040
      032766  020040  020040  054105
      032774  041520  042124  020040
      033002  041501  052524  046101
      033010  020040  043520  042524
      033016     000
2506  033017    120   051501  020123  DH5:   .ASCIZ    /PASS   PC   EXPCTD  ACTUAL  STTE/
      033024  020040  050040  020103
      033032  020040  020040  042440
      033040  050130  052103  020104
      033046  040440  052103  040525
      033054  020114  051440  052124
      033062  000105
2507  033064  040520  051523  020040  DH6:   .ASCIZ    /PASS   PC   STTE/
      033072  020040  041520  020040
      033100  020040  020040  052123
      033106  042524     000
2508  033111    120   051501  020123  DH7:   .ASCIZ    /PASS   PC   STCS/
      033116  020040  050040  020103
      033124  020040  020040  051440
      033132  041524  000123
2509  033136  040520  051523  020040  DH8:   .ASCIZ    /PASS   PC   EXPCTD  ACTUAL  STCS/
      033144  020040  041520  020040
      033152  020040  020040  054105
      033160  041520  042124  020040
      033166  041501  052524  046101
      033174  020040  052123  051503
      033202     000
2510  033203    120   051501  020123  DH9:   .ASCIZ    /PASS   PC   EXPCTD  ACTUAL  IMSK/
      033210  020040  050040  020103
      033216  020040  020040  042440
      033224  050130  052103  020104
      033232  040440  052103  040525
      033240  020114  044440  051515
      033246  000113
2511  033250  040520  051523  020040  DH10:  .ASCIZ    /PASS   PC   IMSK/
      033256  020040  041520  020040
      033264  020040  020040  046511
      033272  045523     000
2512  033275    120   051501  020123  DH11:  .ASCIZ    /PASS   PC   EXPCTD  ACTUAL  MTCE/
```

```
        033302  020040  050040  020103
        033310  020040  020040  042440
        033316  050130  052103  020104
        033324  040440  052103  040525
        033332  020114  046440  041524
        033340  000105
2513    033342  040520  051523  020040  DH12:    .ASCIZ          /PASS     PC       MTCE/
        033350  020040  041520  020040
        033356  020040  020040  052115
        033364  042503     000
2514    033367     120  051501  020123  DH13:    .ASCIZ          /PASS     PC       LINE #  EXPCTD  ACTUAL  DISPLY  $TMP0   $TMP1/
        033374  020040  050040  020103
        033402  020040  020040  046040
        033410  047111  020105  020043
        033416  042440  050130  052103
        033424  020104  040440  052103
        033432  040525  020114  042040
        033440  051511  046120  020131
        033446  022040  046524  030120
        033454  020040  022040  046524
        033462  030520     000
2515    033465     200  040600  051103  ASKDVA:  .ASCIZ  <CRLF><CRLF>/ACR ADDRESS: /
        033472  040440  042104  042522
        033500  051523  020072     000
2516    033505     111  052116  051105  ASKPIV:  .ASCIZ  /INTERRUPT VECTOR ADDRESS: /
        033512  052522  052120  053040
        033520  041505  047524  020122
        033526  042101  051104  051505
        033534  035123  000040
2517    033540  051102  046040  053105  ASKBRL:  .ASCIZ  /BR LEVEL (4-7): /
        033546  046105  024040  026464
        033554  024467  020072     000
2518    033561     200  043520  042524  HDRMES:  .ASCII  <CRLF>/PGTE    PGCS    STTE    STCS    IMSK    PGF     STF     DCF     E
        033566  020040  020040  043520
        033574  051503  020040  020040
        033602  052123  042524  020040
        033610  020040  052123  051503
        033616  020040  020040  046511
        033624  045523  020040  020040
        033632  043520  020106  020040
        033640  020040  052123  020106
        033646  020040  020040  041504
        033654  020106  020040  020040
        033662  054105     103
2519    033665     200     000                    .ASCIZ  <CRLF>
2520    033667     040     000          SPACE:    .ASCIZ  / /
2521    033671     040  000040          TWOSP:    .ASCIZ  /  /
2522    033674  051600  046105  020106  DMPSID:   .ASCIZ  <CRLF>/SELF ID = /
        033702  042111  036440  000040
2523    033710  000200                  CARET:    .ASCIZ  <CRLF>
2524                                              .EVEN
2525
2526    033712  001236  001116  001124  DT1:      .WORD   $PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY,0
        033720  001126  023610  000000
2527    033726  001236  001116  023610  DT2:      .WORD   $PASS,$ERRPC,DISPLY,0
        033734  000000
```

```
2528  033736  001236  001116  023614  DT3:    .WORD   $PASS,$ERRPC,LINMBR,$GDDAT,$BDDAT,DISPLY,$TMP0,$TMP1,0
      033744  001124  001126  023610
      033752  001176  001200  000000
2529  033760     001     000     000  DAF1:   .BYTE   1,0,0,0,0,0,0,0
      033763     000     000     000
      033766     000     000
2530                                   .EVEN
2531
2532          000001                   .END            ;THAT'S ALL FOLKS!
```

```
ABASE = 000000
ACDW1 = 000000
ACDW2 = 000000
ACPUOP= 000000
ACR     023572
ACRH    023574
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
ADR     023576
ADRH    023600
AENV  = 000000
AENVM = 000000
AFATAL= 000000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
APASS = 000000
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPO= 000100
ASKBRL  033540
ASKDVA  033465
ASKPIV  033505

ASWREG= 000000
ATESTN= 000000
AUNIT = 000000
AUSWR = 000000
AVECT1= 000000
AVECT2= 000000
BEGIN   002524
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BMITAB  020536
BM0   = 000400
BM1   = 001000
BM2   = 002000
BM3   = 004000
BPTVEC= 000014
BRKTAB  020536
BRK0  = 000400
BRK1  = 001000
BRK2  = 002000
BRK3  = 004000
CARET   033710
CKSWR = 104407
CLR   = 100000
CONFIG  023660
CR    = 000015
CRLF  = 000200
DAF1    033760
DCF     023576
DCFE  = 000007

DCFH    023600
DCFTAB  020526
DCF0  = 000001
DCF1  = 000002
DCF2  = 000004
DCF3  = 000010
DDISP = 177570
DH1     032635
DH10    033250
DH11    033275
DH12    033342
DH13    033367
DH2     032701
DH3     032725
DH4     032752
DH5     033017
DH6     033064
DH7     033111
DH8     033136
DH9     033203
DISPLA  001142
DISPLY  023610
DISPRE  000174
DMPREG  022234
DMPSID  033674
DSBT  = 000010
DSWR  = 177570
DT1     033712
DT2     033726
DT3     033736
EMTVEC= 000030
EM1     030000
EM10    030147
EM11    030215
EM12    030301
EM13    030337
EM14    030406
EM15    030725
EM16    030762
EM17    031043
EM2     030033
EM20    031077
EM21    031157
EM22    031215
EM23    031264
EM24    031351
EM25    031407
EM26    031456
EM27    031543
EM3     030473
EM30    031601
EM31    031650
EM32    031735

EM33    031772
EM34    032053
EM35    032110
EM36    032171
EM37    032225
EM4     030530
EM40    032305
EM41    032341
EM42    032421
EM43    032454
EM44    032533
EM5     030611
EM6     030645
EM7     030112
ENB   = 000001
ENDTST  020546
ERR   = 100000
ERRVEC= 000004
EXC     023576
EXCE  = 000010
EXCH    023600
GETPAR  022336
GETSID  023474
GO    = 000001
GRJ   = 040000
GTSWR = 104406
HDRMES  033561
HT    = 000011
IE    = 000004
IIIL    023606
IIIP    023604
IIIV    023602
IMITAB  020526
IMSK    023576
IMSKE = 000004
IMSKH   023600
IM0   = 000001
IM1   = 000002
IM2   = 000004
IM3   = 000010
IOTVEC= 000020
IP    = 000010
KIPAR0= 172340
KIPAR1= 172342
KIPAR2= 172344
KIPAR3= 172346
KIPAR4= 172350
KIPAR5= 172352
KIPAR6= 172354
KIPAR7= 172356
KIPDR0= 172300
KIPDR1= 172302
KIPDR2= 172304

KIPDR3= 172306
KIPDR4= 172310
KIPDR5= 172312
KIPDR6= 172314
KIPDR7= 172316
LF    = 000012
LINE0   007234
LINE1   007274
LINE2   007306
LINMBR  023614
LINTST  007324
LKE   = 000004
MAXCNT  023666
MDSD  = 000001
MENB  = 000004
MFRM  = 002000
MID0  = 000000
MID1  = 000400
MID2  = 001000
MID3  = 001400
MINCNT  023664
MLEN  = 000002
MMVEC = 000250
MNTBTS  022202
MTC     023576
MTCE  = 000015
MTCH    023600
MTYP  = 004000
N     = 000044
NOPAR   002502
PARCAL  022214
PARFLG  023656
PBFTAB  020536
PBF0  = 000400
PBF1  = 001000
PBF2  = 002000
PBF3  = 004000
PB0   = 000400
PB1   = 001000
PB2   = 002000
PB3   = 004000
PGBTAB  020536
PGCS    023576
PGCSE = 000001
PGCSH   023600
PGF     023576
PGFE  = 000005
PGFH    023600
PGITAB  020526
PGMR  = 020000
PGONL   016072
PGTE    023576
PGTEE = 000000
```

```
PIFTAB  020526
PIF0  = 000001
PIF1  = 000002
PIF2  = 000004
PIF3  = 000010
PIRQ  = 177772
PIRQVE= 000240
PI0   = 000001
PI1   = 000002
PI2   = 000004
PI3   = 000010
POP   = 005726
POPPOP= 022626
PRDY  = 004000
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSW   = 177776
PTP   = 000002
PWRVEC= 000024
RDCHR = 104410
RDLIN = 104411
RDOCT = 104412
RESVEC= 000010
RTETAB  020526
RTE0  = 000001
RTE1  = 000002
RTE2  = 000004
RTE3  = 000010
R6    =%000006
R7    =%000007
SBFTAB  020536
SBF0  = 000001
SBF1  = 000002
SBF2  = 000004
SBF3  = 000010
SB0   = 000400
SB1   = 001000
SB2   = 002000
SB3   = 004000
SELPAR  002512
SETMNT  022042
SETMN1  022046
SIDNUM  023570
SIDTAB  020516
SID0  = 000000
SID1  = 000400
SID2  = 001000
SID3  = 001400
SIFTAB  020526
SIF0  = 000001
SIF1  = 000002
SIF2  = 000004
SIF3  = 000010
SI0   = 000001
SI2   = 000002
SI3   = 000010
SNAP0   023616
SPACE   033667
SR0   = 177572
SR1   = 177574
SR2   = 177576
SR3   = 172516
SSTMNT  022204
STACK = 001100
START   002014
STBTAB  020536
STCCNT  023662
STCS    023576
STCSE = 000003
STCSH   023600
STF     023576
STFE  = 000006
STITAB  020526
STKLMT= 177774
STMR  = 010000
STONL   017674
STP   = 000002
STTE    023576
STTEE = 000002
SWR     001140
SWREG   000176
SW0   = 000001
SW00  = 000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
SW15  = 100000
SW2   = 000004
SW3   = 000010
SW4   = 000020
SW5   = 000040
SW6   = 000100
SW7   = 000200
SW8   = 000400
SW9   = 001000
TBITVE= 000014
TCR     022334
TIMPER  023612
TKVEC = 000060
TMO   = 000010
TMT   = 000016
TPVEC = 000064
TRAPVE= 000034
TRTVEC= 000014
TST1    002524
TST10   003634
TST11   003720
TST12   003776
TST13   004212
TST14   004426
TST15   004504
TST16   004570
TST17   004646
TST2    002706
TST20   004724
TST21   005010
TST22   005066
TST23   005144
TST24   005230
TST25   005306
TST26   005522
TST27   005736
TST3    003122
TST30   006152
TST31   006366
TST32   006606
TST33   007332
TST34   010222
TST35   010550
TST36   011344
TST37   011672
TST4    003336
TST40   012166
TST41   012414
TST42   012710
TST43   013136
TST44   013406
TST45   013634
TST46   014032
TST47   014560
TST5    003414
TST50   015226
TST51   015500
TST52   015666
TST53   016072
TST54   016716
TST55   017674
TST56   020546
TST6    003500
TST7    003556
TWOSP   033671
TYPDS = 104405
TYPE  = 104401
TYPOC = 104402
TYPON = 104404
TYPOS = 104403
UI0   = 000400
UI1   = 001000
UI2   = 002000
UI3   = 004000
UVITAB  020536
W     = 000005
$APTHD  001000
$ATYC   027556
$ATY1   027532
$ATY3   027540
$ATY4   027550
$AUTOB  001134
$BASE   001304
$BDADR  001122
$BDDAT  001126
$BELL   001220
$CDW1   001310
$CDW2   001312
$CHARC  026614
$CKSWR  025430
$CMTAG  001100
$CM1  = 000006
$CM2  = 000014
$CM3  = 000006
$CM4  = 000007
$CNTLG  026307
$CNTLU  026302
$CORE   025170
$CPUOP  001256
$CRLF   001225
$CROUT  025220
$DBLK   027262
$DDW0   001314
$DDW1   001316
$DDW10  001340
$DDW11  001342
$DDW12  001344
$DDW13  001346
$DDW14  001350
$DDW15  001352
$DDW2   001320
$DDW3   001322
$DDW4   001324
$DDW5   001326
$DDW6   001330
$DDW7   001332
$DDW8   001334
$DDW9   001336
$DEVCT  001240
$DEVM   001306
$DOAGN  023766
$DTBL   027252
$ENDAD  023756
$ENDCT  023724
$ENDMG  023775
$ENULL  023772
$ENV    001250
$ENVM   001251
$EOP    023670
$EOPCT  023716
$ERFLG  001103
$ERMAX  001115
$ERROR  024264
$ERRPC  001116
$ERRTB  001354
$ERRTY  024606
$ERTTL  001112
$ESCAP  001216
$ETABL  001250
$ETEND  001354
$FATAL  001232
$FFLG   027776
$FILLC  001156
$FILLS  001155
$GDADR  001120
$GDDAT  001124
$GET42  023746
$GTSWR  025500
$HD   = 000000
$HIBTS  001000
$HIOCT  025426
$ICNT   001104
$ILLUP  027514
$INTAG  001135
$ITEMB  001114
$KTNEX  025162
$KTOUT  025152
$KT11   025020
$LF     001226
$LFLG   027775
```

```
$LPADR  001106        $MTYP1  001261        $RDSZ = 000014        $TESTN  001234        $TSTNM  001102
$LPERR  001110        $MTYP2  001265        $REGAD  001160        $TIMES  001214        $TTYIN  026266
$LSTAD  025264        $MTYP3  001271        $REG0   001162        $TKB    001146        $TYPDS  027046
$LSTBK  025266        $MTYP4  001275        $REG1   001164        $TKS    001144        $TYPE   026336
$MADR1  001262        $MXCNT  024262        $REG2   001166        $TMP0   001176        $TYPEC  026550
$MADR2  001266        $NULL   001154        $REG3   001170        $TMP1   001200        $TYPEX  026616
$MADR3  001272        $NWTST= 000001        $REG4   001172        $TMP2   001202        $TYPOC  026644
$MADR4  001276        $OCNT   027042        $REG5   001174        $TMP3   001204        $TYPON  026660
$MAIL   001230        $OMODE  027044        $RTNAD  023770        $TMP4   001206        $TYPOS  026620
$MAMS1  001260        $OVER   024246        $SAVR6  027520        $TMP5   001210        $UNIT   001242
$MAMS2  001264        $PASS   001236        $SCOPE  024012        $TMP6   001212        $UNITM  001010
$MAMS3  001270        $PASTM  001006        $SETUP= 000137        $TN     = 000057      $USWR   001254
$MAMS4  001274        $POWER  027522        $SIZE   024762        $TPB    001152        $VECT1  001300
$MBADR  001002        $PWRDN  027354        $SIZEX  025224        $TPFLG  001157        $VECT2  001302
$MFLG   027774        $PWRMG  027510        $STUP = 177777        $TPS    001150        $XTSTR  024036
$MNEW   026325        $PWRUP  027426        $SVLAD  024212        $TRAP   027272        $$GET4= 000000
$MSGAD  001244        $QUES   001224        $SVPC = 000210        $TRAP2  027314        $OFILL  027043
$MSGLG  001246        $RDCHR  025712        $SWR  = 167000        $TRP  = 000013        $40CAT  000000
$MSGTY  001230        $RDLIN  026032        $SWREG  001252        $TRPAD  027326        .     = 033770
$MSWR   026314        $RDOCT  025270        $SWRMK= 000000        $TSTM   001004        .$X   = 001000

. ABS.  033770     000

 ERRORS DETECTED:  0

 CRIIAA.BIC,CRIIAA/CRIIAA.P11
 RUN-TIME: 129 74 2 SECONDS
 RUN-TIME RATIO: 787/206=3.8
 CORE USED:  28K  (55 PAGES)
```