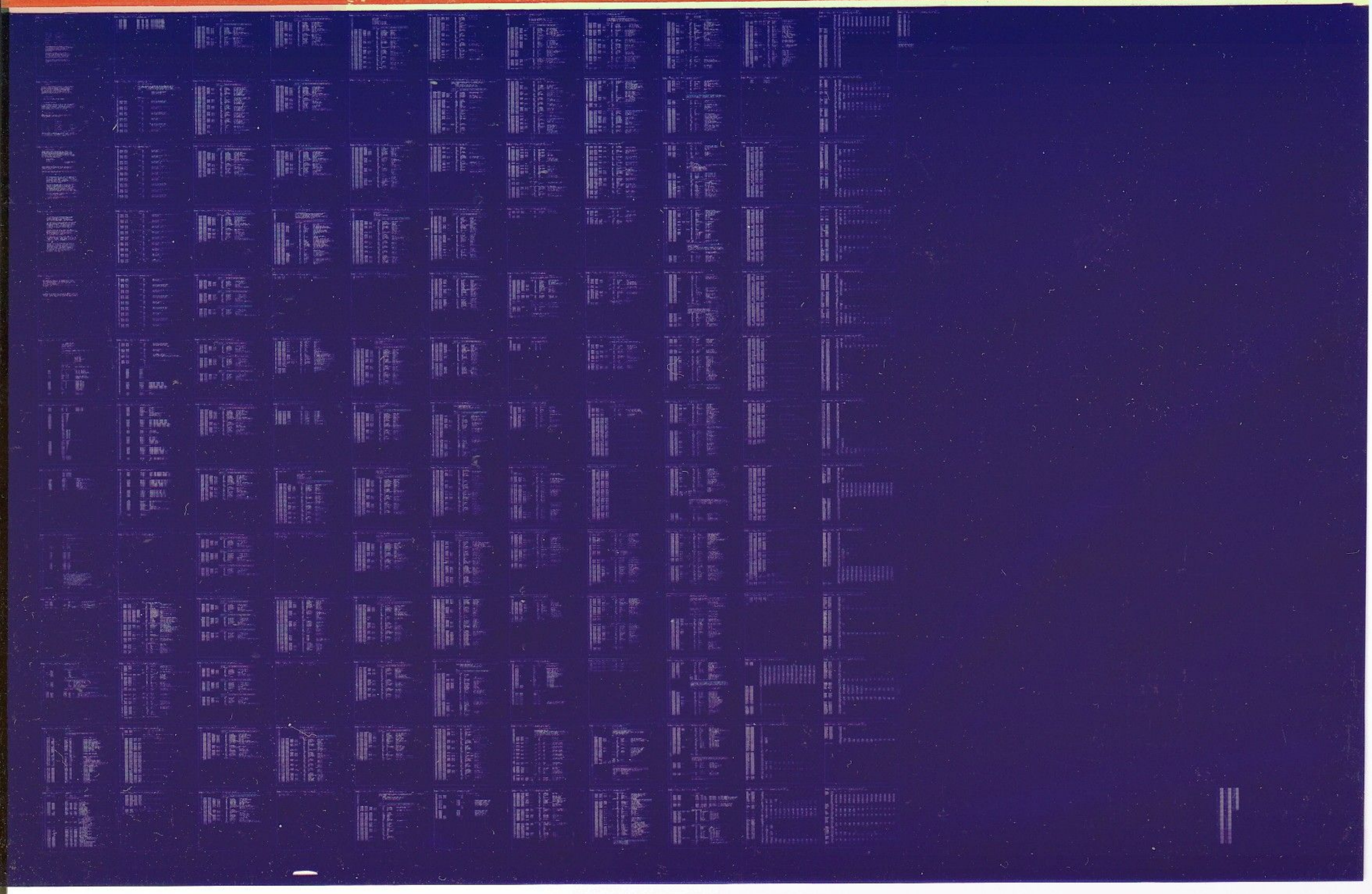
IIST

IIST STND ALN SP
CRIIABO

AH F173B MC

FICHE 1 OF 1

NOV 1979
digital
MADE IN USA



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

IDENTIFICATION

PRODUCT CODE: AC-F172B-MC

PRODUCT NAME: CRIIABO IIST STND ALN SP

DATE CREATED: 15 NOVEMBER 1978

UPDATED : 15 JUNE 1979

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

AUTHOR: ROBERT J. COLLINS / W. WEISKE

UPDATED BY : JAMES M. DUPRE / WILLIAM M. WEISKE

1.0 ABSTRACT

THE DIP11-A SINGLE-INTERFACE 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 12K AND A CONSOLE TERMINAL
B. DIP11-A (IIST) INTERFACE (M8717 MODULE)

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

WHEN A PAPER TAPE IS SUPPLIED, LOAD WITH AN ABS LOADER.
WHEN 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). ON INITIAL PROGRAM LOAD, THE OPERATING PARAMETERS ARE:

```
DEVICE ADDRESS =        777500
INTERRUPT VECTOR =         260
BR LEVEL =                   6
CONFIGURATION =              0  (MAINTENANCE MODE ONLY;
                                 NO COMMUNICATION WITH
                                 IIST BUS)
```

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 TEST SELECTION).

SEVEN TEST SELECTION CODES ARE AVAILABLE FOR SELECTING PROGRAM OPERATION UNDER VARIOUS CONDITIONS. THE CODES ARE USED AS FOLLOWS:

CODE	USE
0	RUNS IN MAINTENANCE MODE ONLY; DOES NOT COMMUNICATE WITH THE IIST BUS DRIVERS OR RECEIVERS. THIS MODE WOULD BE USED IF THERE IS NO TERMINATOR ON THE IIST BUS LINES, IF THE TRANSMIT-INHIBIT INPUT IS ASSERTED, IF THE RECEIVER-INHIBIT INPUT FOR THE LINE CORRESPONDING TO THIS INTERFACE'S SELF-ID IS ASSERTED, OR IF CHECKING OF THE IIST BUS DRIVERS AND RECEIVERS IS NOT DESIRED.
1	THIS CODE IS USED WHEN THERE IS AN IIST BUS TERMINATOR INSTALLED ON THE MODULE BUT NO IIST BUS CABLE IS CONNECTED, OR WHEN THE MODULE IS CONNECTED TO THE IIST BUS AND ALL OTHER INTERFACES ON THE BUS ARE INHIBITED FROM TRANSMITTING OR THIS INTERFACE IS INHIBITED FROM RECEIVING FROM ALL INTERFACES OTHER THAN ITSELF. THIS CODE WOULD BE USED WHEN THE DIP11-C CONTROL PANEL HAS SELECTED THIS INTERFACE TO BE "ONLINE" AND "STANDALONE".
2	THIS CODE IS USED WHEN THE MODULE UNDER TEST IS CONNECTED TO THE IIST BUS AND ALL OTHER INTERFACES ON THE BUS ARE EITHER IDLE (INITIALIZED) OR INHIBITED.
3	THIS CODE IS USED WHEN THE MODULE UNDER TEST IS CONNECTED TO THE IIST BUS AND THERE ARE OTHER ACTIVE, NON-INHIBITED INTERFACES ON THE BUS.

CODE	USE
4	THIS CODE SELECTS THE MAINTENANCE-MODE SELF-BOOT TEST, IN WHICH THE OPERATOR CAN INITIATE A PROGRAMMED OR SANITY-TIMER BOOT OF THIS PROCESSOR VIA A PARTICULAR RECEIVING CHANNEL. THE OPERATOR MUST VERIFY THAT THE PROPER BOOT ACTION TOOK PLACE.
5	THIS CODE SELECTS THE MULTI-PROCESSOR DIALOG-INITIATED BOOT TEST, IN WHICH THE OPERATOR CAN SELECT ANOTHER PROCESSOR TO BE BOOTED. FOLLOWING THE SELECTED BOOT, AN INTERRUPT IS SENT TO THE SELECTED PROCESSOR, SO IF THE BOOT ROUTINE SETS THE INTERRUPT-ENABLE BIT OF THE DIP11-A THE PROGRAM IS REENTERED AND REPORTS THE ACTION.
6	THIS CODE SELECTS THE SANITY TIMEOUT HALT-FREEZE TEST, WHEREBY THE OPERATOR CAN INITIATE A SANITY-TIMER TIMEOUT AND THEN VERIFY THAT THE PROCESSOR WAS PROPERLY HALTED AND/OR THE UNIBUS HUNG. IF AN INDICATOR PANEL, SUCH AS THE DIP11-C, IS CONNECTED, OPERATION OF THE HALT INDICATOR CAN BE VERIFIED.
7	THIS CODE SELECTS THE MANUAL CONTROL PANEL TEST. THE PROGRAM CONTINUALLY CYCLES WHILE MONITORING CHANGES IN THE "DCF" REGISTER. WHEN A CHANGE OCCURS (SUCH AS A "BREAK" BIT CHANGING IN RESPONSE TO AN INHIBIT INPUT IN THE IIST SYSTEM) THE PROGRAM PRINTS OUT THE NEW CONTENTS OF DCF.
10	THIS CODE SELECTS THE "SCOPE-LOOPS" SECTION OF THE PROGRAM. THIS SECTION IS A COLLECTION OF ROUTINES, SELECTABLE BY THE OPERATOR, WHICH REPEATEDLY CYCLE VARIOUS PARTS OF THE LOGIC FOR PURPOSES OF MONITORING WITH AN OSCILLOSCOPE. FOR EXAMPLE, ASSUMING THE "BOOT" CABLE IS NOT CONNECTED TO THE PROCESSOR, ONE SCOPE LOOP CAN CAUSE A STRING OF BOOT PULSES TO BE GENERATED, WHICH CAN THEN BE MONITORED AT J2 PIN B OR J3 PIN 2. OTHER SCOPE LOOPS CAUSE VARIOUS TRANS-MISSIONS ON THE IIST BUS.

7.0 OPERATION

ONCE STARTED WITH CODES 0-3, 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.
SELECTION OF CODES 4-10 CAUSES ENTRY TO OPERATOR-INTERVENTION
OPERATIONS.

8.0 ERRORS

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE
COMMENT AS TO THE TYPE OF FAILURE AND RELEVANT TEST DATA IN TAB-
ULAR 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

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

```
                                .ENABLE ABS,AMA
                        .TITLE  MAINDEC-11-CRIIA-B
                        ;*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
```

```
                                .SBTTL  MEMORY MANAGEMENT DEFINITIONS
                                ;*KT11 VECTOR ADDRESS
000250                          MMVEC=  250
                                ;*KT11 STATUS REGISTER ADDRESSES
177572                          SR0=    177572
177574                          SR1=    177574
177576                          SR2=    177576
172516                          SR3=    172516
                                ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
172300                          KIPDR0= 172300
172302                          KIPDR1= 172302
172304                          KIPDR2= 172304
172306                          KIPDR3= 172306
172310                          KIPDR4= 172310
172312                          KIPDR5= 172312
172314                          KIPDR6= 172314
172316                          KIPDR7= 172316
                                ;*KERNEL "I" PAGE ADDRESS REGISTERS
172340                          KIPAR0= 172340
172342                          KIPAR1= 172342
172344                          KIPAR2= 172344
172346                          KIPAR3= 172346
172350                          KIPAR4= 172350
172352                          KIPAR5= 172352
172354                          KIPAR6= 172354
172356                          KIPAR7= 172356

                                .SBTTL  TRAP CATCHER

000000                                  .=0
                                ;*ALL UNUSED LOCATIONS OF THE VECTOR AREA CONTAIN
                                ;*A ".+2, IOT" SEQUENCE TO CATCH AND PROCESS ILLEGAL
                                ;*TRAPS AND INTERRUPTS THAT MIGHT OCCUR.
                                ;*THE IOT TRAP WHICH IS TAKEN ON THE ILLEGAL TRAP/INT
                                ;*TRAPS TO THE $SCOPE ROUTINE WHICH (IF THE RETURN PC IS
                                ;*LESS THAN 1002) JUMPS TO THE $ERROR ROUTINE.
                                ;*THE $ERROR ROUTINE WILL REPORT THE ERROR AS FOLLOWS:
                                ;*      PC=YYYYYY UNEXPECTED TRAP TO XXX
                                ;*AND RETURN TO THE PROGRAM AT PC=YYYYYY+2
                                ;*WHERE XXX=LOCATION OF ILLEGAL TRAP
                                ;*      YYYYYY=PC AT TIME OF TRAP
                                ;*NOTE: IF THE PROCESSOR IS NOT AN 11/05 THE PROGRAM
                                ;*      CAN BE STARTED AT ADDRESS 0 AS WELL AS ADDRESS 200.

000000  000000  $40CAT: HALT                    ;;HALT
000002  000737          BR      .-100           ;;BRANCH TO 177700 & TIME OUT (NOT ON
                                                ;;11/05)
```

```
(1)  000004  002644                    .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  002644            JMP     @#NOPAR ;;GO TO START OF PROGRAM
246  000204  000137  002654            JMP     SELPAR          ;GO TO SELECT PARAMETERS
247
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
                        000210          $SVPC=.                 ;SAVE PC
                        000046          .=46
000046  032140                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                        000052          .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
                        000210          .=$SVPC                 ;; RESTORE PC
                        001000          .=1000
                                .SBTTL  APT PARAMETER BLOCK

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

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

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  .WORD   0
001102  000                     $TSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
001103  000                     $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DEIECTED
001114  000                     $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
001115  001                     $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TESI
001116  000000                  $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0       ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000                     $AUTOB: .BYTE   0       ;;AUTOMATIC MODE INDICATOR
001135  000                     $INTAG: .BYTE   0       ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570                  SWR:    .WORD   DSWR    ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP   ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560          ;;TTY KBD STATUS
001146  177562                  $TKB:   177562          ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564          ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000                     $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002                     $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012                     $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000                     $TPFLG: .BYTE   0       ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000                  $REGAD: .WORD   0       ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001162  000000                  $REG0:  .WORD   0       ;;CONTAINS (($REGAD)+0)
001164  000000                  $REG1:  .WORD   0       ;;CONTAINS (($REGAD)+2)
001166  000000                  $REG2:  .WORD   0       ;;CONTAINS (($REGAD)+4)
001170  000000                  $REG3:  .WORD   0       ;;CONTAINS (($REGAD)+6)
001172  000000                  $REG4:  .WORD   0       ;;CONTAINS (($REGAD)+10)
001174  000000                  $REG5:  .WORD   0       ;;CONTAINS (($REGAD)+12)
001176  000000                  $TMP0:  .WORD   0       ;;USER DEFINED
001200  000000                  $TMP1:  .WORD   0       ;;USER DEFINED
001202  000000                  $TMP2:  .WORD   0       ;;USER DEFINED
001204  000000                  $TMP3:  .WORD   0       ;;USER DEFINED
001206  000000                  $TMP4:  .WORD   0       ;;USER DEFINED
001210  000000                  $TMP5:  .WORD   0       ;;USER DEFINED
001212  000000                  $TMP6:  .WORD   0       ;;USER DEFINED
001214  000000                  $TIMES: 0               ;;MAX. NUMBER OF ITERATIONS
001216  000000                  $ESCAPE:0               ;;ESCAPE ON ERROR ADDRESS
001220  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
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)
(1)  001354                         $ERRTB:
255                                 ;ERROR PARAMETER TABLE
256
257                                 ;ERROR  1
258  001354  036162                         EM1             ;ACR BIT3-0 LOAD-READER ERROR
259  001356  041056                         DH1             ;PASS PC EXPCTD ACTUAL ACR
260  001360  042134                         DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
261  001362  042202                         DAF1
262
263                                 ;ERROR  2
264  001364  036215                         EM2             ;ACR BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)
265  001366  041122                         DH2             ;PASS PC ACR
266  001370  042150                         DT2             ;$PASS,$ERRPC,DISPLY
267  001372  042202                         DAF1
268
269                                 ;ERROR  3
270  001374  036655                         EM3             ;PGTE BIT11-8 LOAD-READ ERROR
271  001376  041173                         DH4             ;PASS PC EXPCTD ACUTAL PGTE
272  001400  042134                         DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
273  001402  042202                         DAF1
274
275                                 ;ERROR  4
276  001404  036712                         EM4             ;PGTE BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)
277  001406  041146                         DH3             ;PASS PC PGTE
278  001410  042150                         DT2             ;$PASS,$ERRPC,DISPLY
279  001412  042202                         DAF1
280
281                                 ;ERROR  5
282  001414  036773                         EM5             ;PGTE BIT3-0 LOAD-READ ERROR
283  001416  041173                         DH4             ;PASS PC EXPCTD ACUTAL PGTE
284  001420  042134                         DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
285  001422  042202                         DAF1
286
287                                 ;ERROR  6
288  001424  037027                         EM6             ;PGTE BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)
289  001426  041146                         DH3             ;PASS PC PGTE
290  001430  042150                         DT2             ;$PASS,$ERRPC,DISPLY
291  001432  042202                         DAF1
```

```
                                ;ERROR  7
001434  036274                          EM7             ;IE (PGTE BIT 6) CANNOT BE SET
001436  041146                          DH3             ;PASS PC PGTE
001440  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001442  042202                          DAF1

                                ;ERROR  10
001444  036331                          EM10            ;IE (PGTE BIT6) CANNOT BE WRITTEN TO 0
001446  041146                          DH3             ;PASS PC PGTE
001450  042134                          DT1             ;$PASS,$ERRPC,DISPLY
001452  042202                          DAF1

                                ;ERROR  11
001454  036377                          EM11            ;IE (PGTE BIT6) CANNOT BE CLEARED BY RESET (ACR CLR)
001456  041146                          DH3             ;PASS PC PGTE
001460  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001462  042202                          DAF1

                                ;ERROR  12
001464  036463                          EM12            ;PTP (PGTE BIT1) CANNOT BE SET
001466  041146                          DH3             ;PASS PC PGTE
001470  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001472  042202                          DAF1

                                ;ERROR  13
001474  036521                          EM13            ;PTP (PGTE BIT1) CANNOT BE WRITTEN TO 0
001476  041146                          DH3             ;PASS PC PGTE
001500  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001502  042202                          DAF1

                                ;ERROR  14
001504  036570                          EM14            ;PTP (PGTE BIT1) CANNOT BE CLEARED BY RESET (ACR CLR)
001506  041146                          DH3             ;PASS PC PGTE
001510  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001512  042202                          DAF1

                                ;ERROR  15
001514  037107                          EM15            ;STTE BIT11-8 LOAD-READ ERROR
001516  041240                          DH5             ;PASS PC EXPCTD ACUTAL STTE
001520  042134                          DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001522  042202                          DAF1

                                ;ERROR  16
001524  037144                          EM16            ;STTE BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)
001526  041305                          DH6             ;PASS PC STTE
001530  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001532  042202                          DAF1

                                ;ERROR  17
001534  037225                          EM17            ;STTE BIT3-0 LOAD-READ ERROR
001536  041240                          DH5             ;PASS PC EXPCTD ACUTAL STTE
001540  042134                          DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001542  042202                          DAF1
```

```
                                ;ERROR  20
001544  037261                          EM20            ;STIE BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)
001546  041305                          DH6             ;PASS PC STIE
001550  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001552  042202                          DAF1

                                ;ERROR  21
001554  037341                          EM21            ;LKE (STCS BIT2) CANNOT BE SET
001556  041332                          DH7             ;PASS PC STCS
001560  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001562  042202                          DAF1

                                ;ERROR  22
001564  037377                          EM22            ;LKE (STCS BIT2) CANNOT BE WRITTEN TO 0
001566  041332                          DH7             ;PASS PC STCS
001570  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001572  042202                          DAF1

                                ;ERROR  23
001574  037446                          EM23            ;LKE (STCS BIT2) CANNOT BE CLEARED BY RESET (ACR CLR)
001576  041332                          DH7             ;PASS PC STCS
001600  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001602  042202                          DAF1

                                ;ERROR  24
001604  037533                          EM24            ;STP (STCS BIT1) CANNOT BE SET
001606  041332                          DH7             ;PASS PC STCS
001610  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001612  042202                          DAF1

                                ;ERROR  25
001614  037571                          EM25            ;STP (STCS BIT1) CANNOT BE WRITTEN TO 0
001616  041332                          DH7             ;PASS PC STCS
001620  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001622  042202                          DAF1

                                ;ERROR  26
001624  037640                          EM26            ;STP (STCS BIT1) CANNOT BE CLEARED BY RESET (ACR CLR)
001626  041332                          DH7             ;PASS PC STCS
001630  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001632  042202                          DAF1

                                ;ERROR  27
001634  037725                          EM27            ;ENB (STCS BIT0) CANNOT BE SET
001636  041332                          DH7             ;PASS PC STCS
001640  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001642  042202                          DAF1

                                ;ERROR  30
001644  037763                          EM30            ;ENB (STCS BIT0) CANNOT BE WRITTEN TO 0
001646  041332                          DH7             ;PASS PC STCS
001650  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001652  042202                          DAF1
```

```
                                ;ERROR  31
001654  040032                          EM31            ;ENB (STCS BIT0) CANNOT BE CLEARED BY RESET (ACR CLR)
001656  041332                          DH7             ;PASS PC STCS
001660  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001662  042202                          DAF1
                                ;ERROR  32
001664  040117                          EM32            ;STCS BIT15-8 LOAD-READ ERROR
001666  041357                          DH8             ;PASS PC EXPCTD ACUTAL STCS
001670  042134                          DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001672  042202                          DAF1
                                ;ERROR  33
001674  040154                          EM33            ;STCS BIT15-8 WERE NOT CLEARED BY RESET (ACR CLR)
001676  041332                          DH7             ;PASS PC STCS
001700  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001702  042202                          DAF1
                                ;ERROR  34
001704  040235                          EM34            ;IMSK BIT11-8 LOAD-READ ERROR
001706  041424                          DH9             ;PASS PC EXPCTD ACUTAL IMSK
001710  042134                          DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001712  042202                          DAF1
                                ;ERROR  35
001714  040272                          EM35            ;IMSK BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)
001716  041471                          DH10            ;PASS PC IMSK
001720  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001722  042202                          DAF1
                                ;ERROR  36
001724  040353                          EM36            ;IMSK BIT3-0 LOAD-READ ERROR
001726  041424                          DH9             ;PASS PC EXPCTD ACUTAL IMSK
001730  042134                          DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001732  042202                          DAF1
                                ;ERROR  37
001734  040407                          EM37            ;IMSK BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)
001736  041471                          DH10            ;PASS PC IMSK
001740  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001742  042202                          DAF1
                                ;ERROR  40
001744  040467                          EM40            ;MTCE BIT11-8 LOAD-READ ERROR
001746  041516                          DH11            ;PASS PC EXPCTD ACUTAL MTCE
001750  042134                          DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001752  042202                          DAF1
                                ;ERROR  41
001754  040523                          EM41            ;MTCE BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)
001756  041563                          DH12            ;PASS PC MTCE
001760  042150                          DT2             ;$PASS,$ERRPC,DISPLY
001762  042202                          DAF1
```

```
                                ;ERROR  42
001764  040603                          EM42            ;MTCE BIT3-0 LOAD-READ ERROR
001766  041516                          DH11            ;PASS PC EXPCTD ACTUAL MTCE
001770  042134                          DT1             ;$PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY
001772  042202                          DAF1

                                ;ERROR  43
001774  040636                          EM43            ;MTCE BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)
001776  041563                          DH12            ;PASS PC MTCE
002000  042150                          DT2             ;$PASS,$ERRPC,DISPLY
002002  042202                          DAF1
                                ;ERROR  44
002004  040715                          EM44            ;IIST ERROR  - REFER TO THE LISTING AT "PC"
                                                        ;TO DETERMINE THE ERROR.
002006  041610                          DH13            ;PASS PC LINE # EXPCTD ACTUAL DISPLY $TMP0 $TMP1
002010  042160                          DT3             ;$PASS,$ERRPC,LINMBR,$GDDAT,$BDDAT,DISPLY,$TMP0,$TMP1
002012  042202                          DAF1

                                ;ERROR 45
002014  041017                          EM45            ;ACR (3:0) AUTO-INCREMENT ERROR
002016  041056                          DH1
002020  042134                          DT1
002022  042202                          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
```

```
                        STMR=BIT12      ;STTE MODIFICATION REFUSED
                        PRDY=BIT11      ;PG READY
                        SID0=0          ;SELF ID 0
                        SID1=BIT8       ;SELF ID 1
                        SID2=BIT9       ;SELF ID 2
                        SID3=BIT9+BIT8  ;SELF ID 3
                        IP=BIT3         ;INTERRUPT PENDING IP=ERR+FLAGS
                        IE=BIT2         ;1IST INTERRUPT ENABLE
                        PTP=BIT1        ;PGTE PARITY BIT
                        GO=BIT0         ;GENERATE BOOT/INTERRUPT

                        ;STTE BIT DEFINITIONS

                        SB3=BIT11       ;SANITY TIMER BOOT TRANSMIT ENABLE 3
                        SB2=BIT10       ;SANITY TIMER BOOT TRANSMIT ENABLE 2
                        SB1=BIT9        ;SANITY TIMER BOOT TRANSMIT ENABLE 1
                        SB0=BIT8        ;SANITY TIMER BOOT TRANSMIT ENABLE 0
                        SI3=BIT3        ;SANITY TIMER INTERRUPT TRANSMIT ENABLE 3
                        SI2=BIT2        ;SANITY TIMER INTERRUPT TRANSMIT ENABLE 2
                        SI2=BIT1        ;SANITY TIMER INTERRUPT TRANSMIT ENABLE 1
                        SI0=BIT0        ;SANITY TIMER INTERRUPT TRANSMIT ENABLE 0

                        ;STCS BIT DEFINITIONS

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

                        ;IMSK BIT DEFINITIONS

                        BM3=BIT11       ;BOOT INHIBIT 3
                        BM2=BIT10       ;BOOT INHIBIT 2
                        BM1=BIT9        ;BOOT INHIBIT 1
                        BM0=BIT8        ;BOOT INHIBIT 0
                        IM3=BIT3        ;INTERRUPT INHIBIT 3
                        IM2=BIT2        ;INTERRUPT INHIBIT 2
                        IM1=BIT1        ;INTERRUPT INHIBIT 1
                        IM0=BIT0        ;INTERRUPT INHIBIT 0

                        ;PGF BIT DEFINITIONS

                        PBF3=BIT11      ;PROGRAM GENERATED BOOT FLAG 3
                        PBF2=BIT10      ;PROGRAM GENERATED BOOT FLAG 2
                        PBF1=BIT9       ;PROGRAM GENERATED BOOT FLAG 1
                        PBF0=BIT8       ;PROGRAM GENERATED BOOT FLAG 0
                        PIF3=BIT3       ;PROGRAM GENERATED INTERRUPT FLAG 3
                        PIF2=BIT2       ;PROGRAM GENERATED INTERRUPT FLAG 2
                        PIF1=BIT1       ;PROGRAM GENERATED INTERRUPT FLAG 1
                        PIF0=BIT0       ;PROGRAM GENERATED INTERRUPT FLAG 0

                        ;STF BIT DEFINITIONS

                        SBF3=BIT3       ;SANITY TIMER GENERATED BOOT FLAG 3
                        SBF2=BIT2       ;SANITY TIMER GENERATED BOOT FLAG 2
                        SBF1=BIT1       ;SANITY TIMER GENERATED BOOT FLAG 1
```

```
                        SBF0=BIT0       ;SANITY TIMER GENERATED BOOT FLAG 0
                        SIF3=BIT3       ;SANITY TIMER GENERATED INTERRUPT FLAG 3
                        SIF2=BIT2       ;SANITY TIMER GENERATED INTERRUPT FLAG 2
                        SIF1=BIT1       ;SANITY TIMER GENERATED INTERRUPT FLAG 1
                        SIF0=BIT0       ;SANITY TIMER GENERATED INTERRUPT FLAG 0

                        ;DCF BIT DEFINITIONS

                        BRK3=BIT11      ;DISCONNECT OR POWER LOSS STATE OF IIST 3
                        BRK2=BIT10      ;DISCONNECT OR POWER LOSS STATE OF IIST 2
                        BRK1=BIT9       ;DISCONNECT OR POWER LOSS STATE OF IIST 1
                        BRK0=BIT8       ;DISCONNECT OR POWER LOSS STATE OF I!ST0
                        DCF3=BIT3       ;DCLO/DISCONNECT INTERRUPT FLAG 3
                        DCF2=BIT2       ;DCLO/DISCONNECT INTERRUPT FLAG 2
                        DCF1=BIT1       ;DCLO/DISCONNECT INTERRUPT FLAG 1
                        DCF0=BIT0       ;DCLO/DISCONNECT INTERRUPT FLAG 0

                        ;EXC BIT DEFINITIONS

                        UI3=BIT11       ;UNEXPECTED VALID INPUT FLAG 3
                        UI2=BIT10       ;UNEXPECTED VALID INPUT FLAG 2
                        UI1=BIT9        ;UNEXPECTED VALID INPUT FLAG 1
                        UI0=BIT8        ;UNEXPECTED VALID INPUT FLAG 0
                        RTE3=BIT3       ;UNEXPECTED INVALID INPUT FLAG 3
                        RTE2=BIT2       ;UNEXPECTED INVALID INPUT FLAG 2
                        RTE1=BIT1       ;UNEXPECTED INVALID INPUT FLAG 1
                        RTE0=BIT0       ;UNEXPECTED INVALID INPUT FLAG 0

                        ;MTC BIT DEFINITIONS

                        MTYP=BIT11      ;MNT TYPE BIT.0=PG; 1=ST
                        MFRM=BIT10      ;MNT FRAME BIT.0=NORMAL; 1=FRAMING ERROR
                        MID0=0          ;MNT ID 0
                        MID1=BIT8       ;MNT ID 1
                        MID2=BIT9       ;MNT ID 2
                        MID3=BIT9+BIT8  ;MNT ID 3
                        DSBT=BIT3       ;DISABLE BOOT
                        MENB=BIT2       ;ENABLE B11-8 OF MNT
                        MLEN=BIT1       ;ENABLE MAINT. LOOP (IDLE XMIT DRIVERS)
                        MDSD=BIT0       ;DISABLE XMIT DRIVER (CAUSE A BREAK)
                        TMT=DSBT+MENB+MLEN

                        ;OTHER DEFINITIONS

                        POP=5726
                        POPPOP=22626
                        CRLF=200
```

```
                                .SBTTL  TEST MACRO DEFINITIONS
```

```
  703  002024                              START:
  (1)                                      .SBTTL  INITIALIZE THE COMMON TAGS
  (1)                                      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
  (1)  002024  012706  001100                      MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
  (1)  002030  005026                              CLR     (R6)+           ;;CLEAR MEMORY LOCATION
  (1)  002032  022706  001140                      CMP     #SWR,R6 ;;DONE?
  (1)  002036  001374                              BNE     .-6             ;;LOOP BACK IF NO
  (1)  002040  012706  001100                      MOV     #STACK,SP       ;;SETUP THE STACK POINTER
  (1)                                      ;;INITIALIZE A FEW VECTORS
  (1)  002044  012737  032174  000020              MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
  (1)  002052  012737  000340  000022              MOV     #340,@#IOTVEC+2 ;;LEVEL 7
  (1)  002060  012737  032446  000030              MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
  (1)  002066  012737  000340  000032              MOV     #340,@#EMTVEC+2 ;;LEVEL 7
  (1)  002074  012737  035454  000034              MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  002102  012737  000340  000036              MOV     #340,@#TRAPVEC+2;LEVEL 7
  (1)  002110  012737  035536  000024              MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
  (1)  002116  012737  000340  000026              MOV     #340,@#PWRVEC+2 ;;LEVEL 7
  (1)  002124  013737  032106  032100              MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
  (1)  002132  005037  001214                      CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
  (1)  002136  005037  001216                      CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
  (1)  002142  112737  000001  001115              MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
  (1)  002150  012737  002150  001106              MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
  (1)  002156  012737  002156  001110              MOV     #.,$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)  002164  013746  000004                      MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
  (2)  002170  012737  002224  000004              MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
  (2)  002176  012737  177570  001140              MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
  (2)  002204  012737  177570  001142              MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
  (2)  002212  022777  177777  176720              CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
  (2)  002220  001012                              BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  (2)                                                                      ;;AND  THE HARDWARE SWR IS NOT = -1
  (2)  002222  000403                              BR      65$             ;;BRANCH IF NO TIMEOUT
  (2)  002224  012716  002232          64$:        MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
  (2)  002230  000002                              RTI
  (2)  002232  012737  000176  001140  65$:        MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
  (2)  002240  012737  000174  001142              MOV     #DISPREG,DISPLAY
  (2)  002246  012637  000004          66$:        MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
  (1)
  (2)  002252  005037  001236                      CLR     $PASS           ;;CLEAR PASS COUNT
  (2)  002256  132737  000200  001251              BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
  (2)  002264  001403                              BEQ     67$             ;;YES,USE NON-APT SWITCH
  (2)  002266  012737  001252  001140              MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
  (2)  002274                          67$:
  704                                      .SBTTL  TYPE PROGRAM NAME
  (1)                                      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
  (1)  002274  005227  177777                      INC     #-1             ;;FIRST TIME?
  (1)  002300  001070                              BNE     68$             ;;BRANCH IF NO
  (1)  002302  022737  032140  000042              CMP     #$ENDAD,@#42    ;;ACT-11?
  (1)  002310  001464                              BEQ     68$             ;;BRANCH IF YES
  (1)  002312  104401  002360                      TYPE    ,69$            ;;TYPE ASCIZ STRING
  (2)                                      .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
  (2)  002316  005737  000042                      TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
  (2)  002322  001012                              BNE     70$             ;;BRANCH IF YES
  (2)  002324  123727  001250  000001              CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
  (2)  002332  001406                              BEQ     70$             ;;BRANCH IF YES
```

```
    (2)  002334  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
    (2)  002342  001005                          BNE     71$             ;;BRANCH IF NO
    (2)  002344  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
    (2)  002346  000403                          BR      71$
    (2)  002350  112737  000001  001134  70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
    (2)  002356                          71$:
    (1)  002356  000441                          BR      68$             ;;GET OVER THE ASCIZ
    (1)                                  ;;69$:   .ASCIZ  <CRLF>#DIP11-A (IIST) SINGLE-INTERFACE DIAGNOSTIC  MAINDEC-11-CRIIA-B#<C
    (1)  002462                          68$:
    705  002462  012737  000006  000004          MOV     #6,4            ;SET TO TRAP
    706  002470  012737  000004  000006          MOV     #4,6            ;ON AN NXM
    707                                  ;       JSR     PC,$SIZE        ;SIZE MEMORY
    708  002476  005737  025016                  TST     PARFLG  ;SEE IF PARAMETERS TO BE INPUT
    709  002502  001403                          BEQ     1$              ;SKIP IF NOT
    710  002504  004737  023644                  JSR     PC,GETPAR       ;GET PARAMETERS
    711  002510  000446                          BR      2$
    712  002512  005227  177777          1$:     INC     #-1
    713  002516  001043                          BNE     2$              ;FIRST TIME?
    714  002520  022737  032140  000042          CMP     #$ENDAD,@#42    ;ACT-11#
    715  002526  001437                          BEQ     2$              ;DON'T PRINT IF YES.
    716  002530  104401  002536                  TYPE    ,73$            ;;TYPE ASCIZ STRING
    (1)  002534  000434                          BR      72$             ;;GET OVER THE ASCIZ
    (1)                                  ;;73$:   .ASCIZ  <CRLF>/START AT 204 TO SELECT NEW PARAMETERS OR RUNNING MODE/<CRLF>
    (1)  002626                          72$:
    717  002626  012737  000340  177776  2$:     MOV     #340,PSW        ;RISE PRIORITY
    718  002634  005037  025022                  CLR     STCCNT          ;INITIALIZE S.T. COUNT VALUE.
    719  002640  000137  002666                  JMP     BEGIN           ;GO START TESTING
    720
    721
    722                                  ;COME HERE ON START AT 200
    723  002644  005037  025016          NOPAR:  CLR     PARFLG          ;CLEAR PARAMETER FLAG
    724  002650  000137  002024                  JMP     START           ;GO TO STARTUP
    725
    726                                  ;COME HERE ON START AT 204
    727  002654  012737  000001  025016  SELPAR: MOV     #1,PARFLG       ;SET PARAMETER FLAG
    728  002662  000137  002024                  JMP     START
    729
    730  002666  005737  000042          BEGIN:  TST     @#42            ;ARE WE IN ACT?
    731  002672  001026                          BNE     3$              ;IF YES, DON'T PRINT.
    732  002674  104401  002702                  TYPE    ,65$            ;;TYPE ASCIZ STRING
    (1)  002700  000411                          BR      64$             ;;GET OVER THE ASCIZ
    (1)                                  ;;65$:   .ASCIZ  <CRLF>/RUNNING IN MODE /
    (1)  002724                          64$:
    733  002724  013700  025020                  MOV     CONFIG,R0       ;GET SELECTED MODE CODE.
    734  002730  006300                          ASL     R0              ;MAKE WORD INDEX.
    735  002732  016037  002762  002742          MOV     MTAB(R0),2$     ;GET ADDRESS OF MESSAGE TO PRINT.
    736  002740  104401                          TYPE
    737  002742  000000                  2$:     .WORD   0
    738  002744  104401  001225                  TYPE    ,$CRLF          ;NEWLINE.
    739  002750  013700  025020          3$:     MOV     CONFIG,R0
    740  002754  006300                          ASL     R0
    741  002756  000170  003540                  JMP     @DTAB(R0)       ;GO TO SELECTED TEST.
    742  002762                          1$:
    743                                  ;TABLE OF POINTERS TO MESSAGES.
    744  002762  003006  003040  003130  MTAB:   .WORD   TTMG0,TTMG1,TTMG2,TTMG3
         002770  003216
```

```
                        .WORD   TTMG4,TTMG5,TTMG6,TTMG7
                        .WORD   TTMG10

CNFMAX: .-MTAB-2/2      ;MAXIMUM SELECTION CODE ALLOWED.

;MESSAGES DESCRIBING THE CODES:

TTMG0:  .ASCIZ  /0 (MAINTENANCE-MODE ONLY)/
TTMG1:  .ASCIZ  /1 (STANDALONE-ONLINE TO BUS; ISOLATE FROM OTHER IIST'S)/
TTMG2:  .ASCIZ  /2 (ONLINE TO IIST BUS; OTHER IIST'S ISOLATED OR IDLE)/
TTMG3:  .ASCIZ  /3 (ONLINE TO IIST BUS; OTHER IIST'S MIGHT BE ACTIVE)/
TTMG4:  .ASCIZ  /4 (MAINT.-MODE SELF-BOOT TEST)/
TTMG5:  .ASCIZ  /5 (MULTI-PROC DIALOG-INITIATED BOOT TEST)/
TTMG6:  .ASCIZ  /6 (SANITY-TIMEOUT HALT-FREEZE TEST)/
```

```
                003430  042515  052517  020124
                003436  040510  052114  043055
                003444  042522  055105  020105
                003452  042524  052123  000051
   759          003460  020067  046450  047101  TTMG7:  .ASCIZ  /7 (MANUAL CONTROL PANEL TEST)/
                003466  040525  020114  047503
                003474  052116  047522  020114
                003502  040520  042516  020114
                003510  042524  052123  000051
   760          003516  030061  024040  041523  TTMG10: .ASCIZ  /10 (SCOPE LOOPS)/
                003524  050117  020105  047514
                003532  050117  024523     000
   761                  003540                          .EVEN
   762
   763                                          ;DISPATCH TABLE:
   764          003540  003562  003562  003562  DTAB:   TST1,   TST1,   TST1,   TST1
                003546  003562
   765          003550  025030  026160  027500          MMBOOT, MPDIBT, STHLT,  MCP
                003556  027750
   766          003560  030160                          SLOOPS
   767
```

```
                                                .SBTTL
                                                .SBTTL
                                                .SBTTL  BASIC REGISTER TESTS
                                                .SBTTL

                                        ;;******************************************************************
(3)                                     ;*TEST 1        ACR (BIT3-0) WRITE-READ TEST USING A COUNT PATTERN
(3)                                     ;;******************************************************************
(2)  003562  000004                     TST1:   SCOPE
775  003564  004737  023454                    JSR     PC,CLEAR        ;CLEAR THE WORLD
776  003570  012737  003604  001110            MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
777  003576  012737  000017  001124            MOV     #17,$GDDAT      ;INIT EXPECTED TO 17
778  003604  013777  001124  021120  1$:       MOV     $GDDAT,@ACR     ;LOAD ACR WITH $GDDAT
779  003612  017737  021114  024750            MOV     @ACR,DISPLY     ;READ ACR
780  003620  013737  024750  001126            MOV     DISPLY,$BDDAT   ;COPY ACR INTO $BDDAT
781  003626  042737  177760  001126            BIC     #-<17+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
782  003634  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
783  003642  001403                            BEQ     2$              ;SKIP IF OK
784  003644  104001                            ERROR+  1               ;ACR BIT3-0 LOAD-READ ERROR
785  003646  004737  023526                    JSR     PC,DMPREG
786  003652  005237  001124            2$:     INC     $GDDAT          ;BUMP EXPECTED
787  003656  042737  177760  001124            BIC     #-<17+1>,$GDDAT ;MASK TO BITS UNDER TEST
788  003664  023727  001124  000017            CMP     $GDDAT,#17      ;DONE ALL?
789  003672  001344                            BNE     1$              ;TEST AGAIN IF NO
790  003674  012737  003702  001110            MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
791  003702  012777  000017  021022  3$:       MOV     #17,@ACR        ;SET ALL BITS
792  003710  012777  100000  021014            MOV     #CLR,@ACR       ;CLEAR THE WORLD
793  003716  017737  021010  024750            MOV     @ACR,DISPLY     ;READ ACR
794  003724  032737  000017  024750            BIT     #17,DISPLY      ;CHECK BIT3-0 IN ACR FOR 0
795  003732  001403                            BEQ     TST2            ;;SKIP IF RESET CLEARED BIT3-0 IN ACR
796  003734  104002                            ERROR+  2               ;BIT3-0 IN ACR WERE NOT CLEARED BY RESET
797  003736  004737  023526                    JSR     PC,DMPREG
```

```
                                        ;;*****************************************************************
                                        ;*TEST 2         CHECK ACR AUTO-INCREMENT LOGIC
                                        ;;*****************************************************************
003742  000004                  TST2:   SCOPE
003744  005037  024750                  CLR     DISPLY          ;START INITIAL ACR LOAD VALUE AT 0
003750  012737  000001  001124          MOV     #1,$GDDAT       ;INIT EXPECTED VALUE TO 1.
003756  012737  003764  001110          MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
003764  013777  024750  020740  1$:     MOV     DISPLY,@ACR     ;LOAD ACR WITH INITIAL VALUE.
003772  005777  020740                  TST     @ADR            ;DO A DATI TO ADR
003776  012700  000005                  MOV     #5,R0           ;SET LOOP COUNTER FOR REPEAT READS.
004002  017737  020724  001126  2$:     MOV     @ACR,$BDDAT     ;GET ACR CONTENTS
004010  123737  001126  001124          CMPB    $BDDAT,$GDDAT   ;CHECK LOWER BYTE (ACCESS CODE)
004016  001403                          BEQ     3$              ;FOR INCREMENT OR NOT.
004020  104045                          ERROR+  45              ;ACR AUTO-INCR ERROR
004022  004737  023526                  JSR     PC,DMPREG
004026  005300                  3$:     DEC     R0              ;LOOP TO READ ACR SEVERAL TIMES
004030  001364                          BNE     2$              ; TO MAKE SURE IT DOESN'T CHANGE
004032  013777  024750  020672          MOV     DISPLY,@ACR     ;LOAD ACR WITH INITIAL VALUE.
004040  012777  000000  020670          MOV     #0,@ADR         ;DO DATO TO ADR
004046  017737  020660  001126          MOV     @ACR,$BDDAT     ;GET ACR CONTENTS
004054  123737  001126  001124          CMPB    $BDDAT,$GDDAT   ;CHECK LOWER BYTE (ACCESS CODE)
004062  001403                          BEQ     4$              ;FOR INCREMENT OR NOT.
004064  104045                          ERROR+  45              ;ACR AUTO-INCR ERROR
004066  004737  023526                  JSR     PC,DMPREG
004072  013777  024750  020632  4$:     MOV     DISPLY,@ACR     ;LOAD ACR WITH INITIAL VALUE.
004100  042777  177777  020630          BIC     #-1,@ADR        ;DO DATIP-DATO INTO ADR
004106  017737  020620  001126          MOV     @ACR,$BDDAT     ;GET ACR CONTENTS
004114  123737  001126  001124          CMPB    $BDDAT,$GDDAT   ;CHECK LOWER BYTE (ACCESS CODE)
004122  001403                          BEQ     5$              ;FOR INCREMENT OR NOT.
004124  104045                          ERROR+  45              ;ACR AUTO-INCR ERROR
004126  004737  023526                  JSR     PC,DMPREG
004132  013777  024750  020572  5$:     MOV     DISPLY,@ACR     ;LOAD ACR WITH INITIAL VALUE.
004140  112777  000000  020570          MOVB    #0,@ADR         ;DO DATOB TO LOW BYTE OF ADR
004146  017737  020560  001126          MOV     @ACR,$BDDAT     ;GET ACR CONTENTS
004154  123737  001126  001124          CMPB    $BDDAT,$GDDAT   ;CHECK LOWER BYTE (ACCESS CODE)
004162  001403                          BEQ     6$              ;FOR INCREMENT OR NOT.
004164  104045                          ERROR+  45              ;ACR AUTO-INCR ERROR
004166  004737  023526                  JSR     PC,DMPREG
004172  013777  024750  020532  6$:     MOV     DISPLY,@ACR     ;LOAD ACR WITH INITIAL VALUE.
004200  112777  000000  020532          MOVB    #0,@ADRH        ;DO DATOB TO HIGH BYTE OF ADR
004206  017737  020520  001126          MOV     @ACR,$BDDAT     ;GET ACR CONTENTS
004214  123737  001126  001124          CMPB    $BDDAT,$GDDAT   ;CHECK LOWER BYTE (ACCESS CODE)
004222  001403                          BEQ     7$              ;FOR INCREMENT OR NOT.
004224  104045                          ERROR+  45              ;ACR AUTO-INCR ERROR
004226  004737  023526                  JSR     PC,DMPREG
004232  005237  024750          7$:     INC     DISPLY          ;BUMP INITIAL VALUE
004236  005237  001124                  INC     $GDDAT          ;BUMP EXPECTED VALUE
004242  023727  024750  000011          CMP     DISPLY,#11      ;LOOP THRU CODE 0-10 FIRST (AUTO-INCREMENT)
004250  002645                          BLT     1$
004252  003005                          BGT     8$              ;CODES 11-17 DO NOT INCREMENT,
004254  012737  000011  001124          MOV     #11,$GDDAT      ;SO SET INITIAL & EXPECTED EQUAL.
004262  000137  003764                  JMP     1$
004266  023727  024750  000020  8$:     CMP     DISPLY,#20      ;ARE WE DONE?
004274  002633                          BLT     1$              ;LOOP IF NOT.
```

```
854                                     ;;*****************************************************************
(4)                                     ;*TEST 3        PGTE (BIT11-8) WRITE-READ TEST USING PGTE COUNT PATTERN
(4)                                     ;;*****************************************************************
(3)  004276  000004                     TST3:   SCOPE
(1)  004300  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  004304  012737  004320  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  004312  012737  007400  001124             MOV     #7400,$GDDAT    ;INIT EXPECTED TO 7400
(1)  004320  012777  000000  020404  1$:        MOV     #PGTEE,@ACR     ;SET ADR TO PGTE
(1)  004326  013777  001124  020402             MOV     $GDDAT,@PGTE    ;LOAD PGTE WITH $GDDAT
(1)  004334  012777  000000  020370             MOV     #PGTEE,@ACR     ;RESET ADR TO PGTE
(1)  004342  017737  020370  024750             MOV     @PGTE,DISPLY    ;READ PGTE
(1)  004350  013737  024750  001126             MOV     DISPLY,$BDDAT   ;COPY PGTE INTO $BDDAT
(1)  004356  042737  170377  001126             BIC     #-<7400+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  004364  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  004372  001403                             BEQ     2$              ;SKIP IF OK
(1)  004374  104003                             ERROR+  N               ;PGTE BIT11-8 LOAD-READ ERROR
(1)  004376  004737  023526                     JSR     PC,DMPREG
(1)  004402  062737  000400  001124  2$:        ADD     #BIT8,$GDDAT    ;BUMP EXPECTED
(1)  004410  042737  170377  001124             BIC     #-<7400+1>,$GDDAT       ;MASK TO BITS UNDER TEST
(1)  004416  023727  001124  007400             CMP     $GDDAT,#7400    ;DONE ALL?
(1)  004424  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  004426  012737  004434  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  004434  012777  000000  020270  3$:        MOV     #PGTEE,@ACR     ;RESET ADR TO PGTE
(1)  004442  012777  007400  020266             MOV     #7400,@PGTE             ;SET ALL BITS
(1)  004450  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  004454  012777  000000  020250             MOV     #PGTEE,@ACR     ;SET ADDR TO PGTE
(1)  004462  017737  020250  024750             MOV     @PGTE,DISPLY    ;READ PGTE
(1)  004470  032737  007400  024750             BIT     #7400,DISPLY    ;CHECK BIT11-8 IN PGTE FOR 0
(1)  004476  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT11-8 IN PGTE
(1)  004500  104004                             ERROR+  N               ;BIT11-8 IN PGTE WERE NOT CLEARED BY RESET
(1)  004502  004737  023526                     JSR     PC,DMPREG
(1)  004506                             4$:
```

```
856                                     ;;*******************************************************************
(4)                                     ;*TEST 4        PGTE (BIT3-0) WRITE-READ TEST USING PGTE COUNT PATTERN
(4)                                     ;;*******************************************************************
(3)  004506  000004                     TST4:   SCOPE
(1)  004510  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  004514  012737  004530  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  004522  012737  000017  001124             MOV     #17,$GDDAT      ;INIT EXPECTED TO 17
(1)  004530  012777  000000  020174  1$:        MOV     #PGTEE,@ACR     ;SET ADR TO PGTE
(1)  004536  013777  001124  020172             MOV     $GDDAT,@PGTE    ;LOAD PGTE WITH $GDDAT
(1)  004544  012777  000000  020160             MOV     #PGTEE,@ACR     ;RESET ADR TO PGTE
(1)  004552  017737  020160  024750             MOV     @PGTE,DISPLY    ;READ PGTE
(1)  004560  013737  024750  001126             MOV     DISPLY,$BDDAT   ;COPY PGTE INTO $BDDAT
(1)  004566  042737  177760  001126             BIC     #-<17+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  004574  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  004602  001403                             BEQ     2$              ;SKIP IF OK
(1)  004604  104005                             ERROR+  N               ;PGTE BIT3-0 LOAD-READ ERROR
(1)  004606  004737  023526                     JSR     PC,DMPREG
(1)  004612  062737  000001  001124  2$:        ADD     #BIT0,$GDDAT    ;BUMP EXPECTED
(1)  004620  042737  177760  001124             BIC     #-<17+1>,$GDDAT ;MASK TO BITS UNDER TEST
(1)  004626  023727  001124  000017             CMP     $GDDAT,#17      ;DONE ALL?
(1)  004634  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  004636  012737  004644  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  004644  012777  000000  020060  3$:        MOV     #PGTEE,@ACR     ;RESET ADR TO PGTE
(1)  004652  012777  000017  020056             MOV     #17,@PGTE               ;SET ALL BITS
(1)  004660  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  004664  012777  000000  020040             MOV     #PGTEE,@ACR     ;SET ADDR TO PGTE
(1)  004672  017737  020040  024750             MOV     @PGTE,DISPLY    ;READ PGTE
(1)  004700  032737  000017  024750             BIT     #17,DISPLY      ;CHECK BIT3-0 IN PGTE FOR 0
(1)  004706  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT3-0 IN PGTE
(1)  004710  104006                             ERROR+  N               ;BIT3-0 IN PGTE WERE NOT CLEARED BY RESET
(1)  004712  004737  023526                     JSR     PC,DMPREG
(1)  004716                             4$:
```

```
858
(4)                                     ;;*****************************************************************
(4)                                     ;*TEST 5        TEST THAT IE (PGCS BIT2) CAN BE SET
(4)                                     ;;*****************************************************************
(3)  004716  000004                     TST5:   SCOPE
(1)  004720  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE INTERFACE
(1)  004724  012777  000001  020000            MOV     #PGCSE,@ACR     ;SET ADR TO PGCS
(1)  004732  012777  000004  017776            MOV     #IE,@PGCS               ;SET IE IN PGCS
(1)  004740  012777  000001  017764            MOV     #PGCSE,@ACR     ;RESET ADR TO PGCS
(1)  004746  017737  017764  024750            MOV     @PGCS,DISPLY    ;READ PGCS
(1)  004754  032737  000004  024750            BIT     #IE,DISPLY      ;CHECK FOR IE=1
(3)  004762  001003                             BNE     TST6            ;;SKIP IF IE IS SET IN PGCS
(1)  004764  104007                             ERROR+  N               ;IE (PGCS BIT2) CANNOT BE SET
(1)  004766  004737  023526                     JSR     PC,DMPREG
(1)
(5)                                     ;;*****************************************************************
(4)                                     ;*TEST 6        TEST THAT IE (PGCS BIT2) CAN BE CLEARED
(4)                                     ;;*****************************************************************
(3)  004772  000004                     TST6:   SCOPE
(1)  004774  012777  000001  017730            MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  005002  012777  000004  017726            MOV     #IE,@PGCS               ;SET IE IN PGCS
(1)  005010  012777  000001  017714            MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  005016  042777  000004  017712            BIC     #IE,@PGCS               ;CLEAR IE IN PGCS
(1)  005024  012777  000001  017700            MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  005032  017737  017700  024750            MOV     @PGCS,DISPLY    ;READ PGCS
(1)  005040  032737  000004  024750            BIT     #IE,DISPLY      ;CHECK FOR IE=0
(3)  005046  001403                             BEQ     TST7            ;;SKIP IF IE IS CLEAR IN PGCS
(1)  005050  104010                             ERROR+  N               ;IE (PGCS BIT2) CANNOT BE CLEARED
(1)  005052  004737  023526                     JSR     PC,DMPREG
(1)                                                                     ;BY WRITING IT TO 0
(1)
(5)                                     ;;*****************************************************************
(4)                                     ;*TEST 7        TEST THAT IE (PGCS BIT2) CAN BE CLEARED BY RESET
(4)                                     ;;*****************************************************************
(3)  005056  000004                     TST7:   SCOPE
(1)  005060  012777  000001  017644            MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  005066  012777  000004  017642            MOV     #IE,@PGCS               ;SET IE IN PGCS
(1)  005074  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE INTERFACE
(1)  005100  012777  000001  017624            MOV     #PGCSE,@ACR     ;SET ADR TO PGCS
(1)  005106  017737  017624  024750            MOV     @PGCS,DISPLY    ;READ PGCS
(1)  005114  032737  000004  024750            BIT     #IE,DISPLY      ;CHECK FOR IE=0
(3)  005122  001403                             BEQ     TST10           ;;SKIP IS IE IS CLEARED BY RESET IN PGCS
(1)  005124  104011                             ERROR+  N               ;IE (PGCS BIT2) CANNOT BE CLEARED BY RESET
(1)  005126  004737  023526                     JSR     PC,DMPREG
```

```
860
(4)                                     ;;*********************************************************************
(4)                                     ;*TEST 10       TEST THAT PTP (PGCS BIT1) CAN BE SET
(4)                                     ;;*********************************************************************
(3)  005132  000004                     TST10:  SCOPE
(1)  005134  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE INTERFACE
(1)  005140  012777  000001  017564             MOV     #PGCSE,@ACR     ;SET ADR TO PGCS
(1)  005146  012777  000002  017562             MOV     #PTP,@PGCS              ;SET PTP IN PGCS
(1)  005154  012777  000001  017550             MOV     #PGCSE,@ACR     ;RESET ADR TO PGCS
(1)  005162  017737  017550  024750             MOV     @PGCS,DISPLY    ;READ PGCS
(1)  005170  032737  000002  024750             BIT     #PTP,DISPLY     ;CHECK FOR PTP=1
(3)  005176  001003                             BNE     TST11           ;;SKIP IF PTP IS SET IN PGCS
(1)  005200  104012                             ERROR+  N               ;PTP (PGCS BIT1) CANNOT BE SET
(1)  005202  004737  023526                     JSR     PC,DMPREG
(1)
(5)                                     ;;*********************************************************************
(4)                                     ;*TEST 11       TEST THAT PTP (PGCS BIT1) CAN BE CLEARED
(4)                                     ;;*********************************************************************
(3)  005206  000004                     TST11:  SCOPE
(1)  005210  012777  000001  017514             MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  005216  012777  000002  017512             MOV     #PTP,@PGCS              ;SET PTP IN PGCS
(1)  005224  012777  000001  017500             MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  005232  042777  000002  017476             BIC     #PTP,@PGCS              ;CLEAR PTP IN PGCS
(1)  005240  012777  000001  017464             MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  005246  017737  017464  024750             MOV     @PGCS,DISPLY    ;READ PGCS
(1)  005254  032737  000002  024750             BIT     #PTP,DISPLY     ;CHECK FOR PTP=0
(3)  005262  001403                             BEQ     TST12           ;;SKIP IF PTP IS CLEAR IN PGCS
(1)  005264  104013                             ERROR+  N               ;PTP (PGCS BIT1) CANNOT BE CLEARED
(1)  005266  004737  023526                     JSR     PC,DMPREG
(1)                                                                     ;BY WRITING IT TO 0
(1)
(5)                                     ;;*********************************************************************
(4)                                     ;*TEST 12       TEST THAT PTP (PGCS BIT1) CAN BE CLEARED BY RESET
(4)                                     ;;*********************************************************************
(3)  005272  000004                     TST12:  SCOPE
(1)  005274  012777  000001  017430             MOV     #PGCSE,@ACR     ;SET ADDRESS TO PGCS
(1)  005302  012777  000002  017426             MOV     #PTP,@PGCS              ;SET PTP IN PGCS
(1)  005310  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE INTERFACE
(1)  005314  012777  000001  017410             MOV     #PGCSE,@ACR     ;SET ADR TO PGCS
(1)  005322  017737  017410  024750             MOV     @PGCS,DISPLY    ;READ PGCS
(1)  005330  032737  000002  024750             BIT     #PTP,DISPLY     ;CHECK FOR PTP=0
(3)  005336  001403                             BEQ     TST13           ;;SKIP IS PTP IS CLEARED BY RESET IN PGCS
(1)  005340  104014                             ERROR+  N               ;PTP (PGCS BIT1) CANNOT BE CLEARED BY RESET
(1)  005342  004737  023526                     JSR     PC,DMPREG
```

```
                                ;;*****************************************************************
                                ;*TEST 13       STTE (BIT11-8) WRITE-READ TEST USING STTE COUNT PATTERN
                                ;;*****************************************************************
005346  000004                  TST13:  SCOPE
005350  004737  023454                  JSR     PC,CLEAR        ;CLEAR THE WORLD
005354  012737  005370  001110          MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
005362  012737  007400  001124          MOV     #7400,$GDDAT    ;INIT EXPECTED TO 7400
005370  012777  000002  017334  1$:     MOV     #STTEE,@ACR     ;SET ADR TO STTE
005376  013777  001124  017332          MOV     $GDDAT,@STTE    ;LOAD STTE WITH $GDDAT
005404  012777  000002  017320          MOV     #STTEE,@ACR     ;RESET ADR TO STTE
005412  017737  017320  024750          MOV     @STTE,DISPLY    ;READ STTE
005420  013737  024750  001126          MOV     DISPLY,$BDDAT   ;COPY STTE INTO $BDDAT
005426  042737  170377  001126          BIC     #-<7400+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
005434  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
005442  001403                          BEQ     2$              ;SKIP IF OK
005444  104015                          ERROR+  N               ;STTE BIT11-8 LOAD-READ ERROR
005446  004737  023526                  JSR     PC,DMPREG
005452  062737  000400  001124  2$:     ADD     #BIT8,$GDDAT    ;BUMP EXPECTED
005460  042737  170377  001124          BIC     #-<7400+1>,$GDDAT       ;MASK TO BITS UNDER TEST
005466  023727  001124  007400          CMP     $GDDAT,#7400    ;DONE ALL?
005474  001335                          BNE     1$              ;TEST AGAIN IF NO
005476  012737  005504  001110          MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
005504  012777  000002  017220  3$:     MOV     #STTEE,@ACR     ;RESET ADR TO STTE
005512  012777  007400  017216          MOV     #7400,@STTE             ;SET ALL BITS
005520  004737  023454                  JSR     PC,CLEAR        ;CLEAR THE WORLD
005524  012777  000002  017200          MOV     #STTEE,@ACR     ;SET ADDR TO STTE
005532  017737  017200  024750          MOV     @STTE,DISPLY    ;READ STTE
005540  032737  007400  024750          BIT     #7400,DISPLY    ;CHECK BIT11-8 IN STTE FOR 0
005546  001403                          BEQ     4$              ;SKIP IF RESET CLEARED BIT11-8 IN STTE
005550  104016                          ERROR+  N               ;BIT11-8 IN STTE WERE NOT CLEARED BY RESET
005552  004737  023526                  JSR     PC,DMPREG
005556                          4$:
```

```
                                        ;;*****************************************************************
                                        ;*TEST 14       STTE (BIT3-0) WRITE-READ TEST USING STTE COUNT PATTERN
                                        ;;*****************************************************************
005556  000004                  TST14:  SCOPE
005560  004737  023454                  JSR     PC,CLEAR        ;CLEAR THE WORLD
005564  012737  005600  001110          MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
005572  012737  000017  001124          MOV     #17,$GDDAT      ;INIT EXPECTED TO 17
005600  012777  000002  017124  1$:     MOV     #STTEE,@ACR     ;SET ADR TO STTE
005606  013777  001124  017122          MOV     $GDDAT,@STTE    ;LOAD STTE WITH $GDDAT
005614  012777  000002  017110          MOV     #STTEE,@ACR     ;RESET ADR TO STTE
005622  017737  017110  024750          MOV     @STTE,DISPLY    ;READ STTE
005630  013737  024750  001126          MOV     DISPLY,$BDDAT   ;COPY STTE INTO $BDDAT
005636  042737  177760  001126          BIC     #-<17+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
005644  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
005652  001403                          BEQ     2$              ;SKIP IF OK
005654  104017                          ERROR+  N               ;STTE BIT3-0 LOAD-READ ERROR
005656  004737  023526                  JSR     PC,DMPREG
005662  062737  000001  001124  2$:     ADD     #BIT0,$GDDAT    ;BUMP EXPECTED
005670  042737  177760  001124          BIC     #-<17+1>,$GDDAT ;MASK TO BITS UNDER TEST
005676  023727  001124  000017          CMP     $GDDAT,#17      ;DONE ALL?
005704  001335                          BNE     1$              ;TEST AGAIN IF NO
005706  012737  005714  001110          MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
005714  012777  000002  017010  3$:     MOV     #STTEE,@ACR     ;RESET ADR TO STTE
005722  012777  000017  017006          MOV     #17,@STTE               ;SET ALL BITS
005730  004737  023454                  JSR     PC,CLEAR        ;CLEAR THE WORLD
005734  012777  000002  016770          MOV     #STTEE,@ACR     ;SET ADDR TO STTE
005742  017737  016770  024750          MOV     @STTE,DISPLY    ;READ STTE
005750  032737  000017  024750          BIT     #17,DISPLY      ;CHECK BIT3-0 IN STTE FOR 0
005756  001403                          BEQ     4$              ;SKIP IF RESET CLEARED BIT3-0 IN STTE
005760  104020                          ERROR+  N               ;BIT3-0 IN STTE WERE NOT CLEARED BY RESET
005762  004737  023526                  JSR     PC,DMPREG
005766                          4$:
```

```
866
(4)                                     ;;*****************************************************************
(4)                                     ;*TEST 15       TEST THAT LKE (STCS BIT2) CAN BE SET
(4)                                     ;;*****************************************************************
(3)  005766  000004                     TST15:  SCOPE
(1)  005770  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE INTERFACE
(1)  005774  012777  000003  016730            MOV     #STCSE,@ACR     ;SET ADR TO STCS
(1)  006002  012777  000004  016726            MOV     #LKE,@STCS              ;SET LKE IN STCS
(1)  006010  012777  000003  016714            MOV     #STCSE,@ACR     ;RESET ADR TO STCS
(1)  006016  017737  016714  024750            MOV     @STCS,DISPLY    ;READ STCS
(1)  006024  032737  000004  024750            BIT     #LKE,DISPLY     ;CHECK FOR LKE=1
(3)  006032  001003                             BNE     TST16           ;;SKIP IF LKE IS SET IN STCS
(1)  006034  104021                             ERROR+  N               ;LKE (STCS BIT2) CANNOT BE SET
(1)  006036  004737  023526                     JSR     PC,DMPREG
(1)
(5)                                     ;;*****************************************************************
(4)                                     ;*TEST 16       TEST THAT LKE (STCS BIT2) CAN BE CLEARED
(4)                                     ;;*****************************************************************
(3)  006042  000004                     TST16:  SCOPE
(1)  006044  012777  000003  016660            MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006052  012777  000004  016656            MOV     #LKE,@STCS              ;SET LKE IN STCS
(1)  006060  012777  000003  016644            MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006066  042777  000004  016642            BIC     #LKE,@STCS              ;CLEAR LKE IN STCS
(1)  006074  012777  000003  016630            MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006102  017737  016630  024750            MOV     @STCS,DISPLY    ;READ STCS
(1)  006110  032737  000004  024750            BIT     #LKE,DISPLY     ;CHECK FOR LKE=0
(3)  006116  001403                             BEQ     TST17           ;;SKIP IF LKE IS CLEAR IN STCS
(1)  006120  104022                             ERROR+  N               ;LKE (STCS BIT2) CANNOT BE CLEARED
(1)  006122  004737  023526                     JSR     PC,DMPREG
(1)                                                                     ;BY WRITING IT TO 0
(1)
(5)                                     ;;*****************************************************************
(4)                                     ;*TEST 17       TEST THAT LKE (STCS BIT2) CAN BE CLEARED BY RESET
(4)                                     ;;*****************************************************************
(3)  006126  000004                     TST17:  SCOPE
(1)  006130  012777  000003  016574            MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006136  012777  000004  016572            MOV     #LKE,@STCS              ;SET LKE IN STCS
(1)  006144  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE INTERFACE
(1)  006150  012777  000003  016554            MOV     #STCSE,@ACR     ;SET ADR TO STCS
(1)  006156  017737  016554  024750            MOV     @STCS,DISPLY    ;READ STCS
(1)  006164  032737  000004  024750            BIT     #LKE,DISPLY     ;CHECK FOR LKE=0
(3)  006172  001403                             BEQ     TST20           ;;SKIP IS LKE IS CLEARED BY RESET IN STCS
(1)  006174  104023                             ERROR+  N               ;LKE (STCS BIT2) CANNOT BE CLEARED BY RESET
(1)  006176  004737  023526                     JSR     PC,DMPREG
```

```
868
(4)                                     ;;*****************************************************************
(4)                                     ;*TEST 20       TEST THAT STP (STCS BIT1) CAN BE SET
(4)                                     ;;*****************************************************************
(3)  006202  000004                     TST20:  SCOPE
(1)  006204  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE INTERFACE
(1)  006210  012777  000003  016514             MOV     #STCSE,@ACR     ;SET ADR TO STCS
(1)  006216  012777  000002  016512             MOV     #STP,@STCS              ;SET STP IN STCS
(1)  006224  012777  000003  016500             MOV     #STCSE,@ACR     ;RESET ADR TO STCS
(1)  006232  017737  016500  024750             MOV     @STCS,DISPLY    ;READ STCS
(1)  006240  032737  000002  024750             BIT     #STP,DISPLY     ;CHECK FOR STP=1
(3)  006246  001003                             BNE     TST21           ;;SKIP IF STP IS SET IN STCS
(1)  006250  104024                             ERROR+  N               ;STP (STCS BIT1) CANNOT BE SET
(1)  006252  004737  023526                     JSR     PC,DMPREG
(1)
(5)                                     ;;*****************************************************************
(4)                                     ;*TEST 21       TEST THAT STP (STCS BIT1) CAN BE CLEARED
(4)                                     ;;*****************************************************************
(3)  006256  000004                     TST21:  SCOPE
(1)  006260  012777  000003  016444             MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006266  012777  000002  016442             MOV     #STP,@STCS              ;SET STP IN STCS
(1)  006274  012777  000003  016430             MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006302  042777  000002  016426             BIC     #STP,@STCS              ;CLEAR STP IN STCS
(1)  006310  012777  000003  016414             MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006316  017737  016414  024750             MOV     @STCS,DISPLY    ;READ STCS
(1)  006324  032737  000002  024750             BIT     #STP,DISPLY     ;CHECK FOR STP=0
(3)  006332  001403                             BEQ     TST22           ;;SKIP IF STP IS CLEAR IN STCS
(1)  006334  104025                             ERROR+  N               ;STP (STCS BIT1) CANNOT BE CLEARED
(1)  006336  004737  023526                     JSR     PC,DMPREG
(1)                                                                     ;BY WRITING IT TO 0
(1)
(5)                                     ;;*****************************************************************
(4)                                     ;*TEST 22       TEST THAT STP (STCS BIT1) CAN BE CLEARED BY RESET
(4)                                     ;;*****************************************************************
(3)  006342  000004                     TST22:  SCOPE
(1)  006344  012777  000003  016360             MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006352  012777  000002  016356             MOV     #STP,@STCS              ;SET STP IN STCS
(1)  006360  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE INTERFACE
(1)  006364  012777  000003  016340             MOV     #STCSE,@ACR     ;SET ADR TO STCS
(1)  006372  017737  016340  024750             MOV     @STCS,DISPLY    ;READ STCS
(1)  006400  032737  000002  024750             BIT     #STP,DISPLY     ;CHECK FOR STP=0
(3)  006406  001403                             BEQ     TST23           ;;SKIP IS STP IS CLEARED BY RESET IN STCS
(1)  006410  104026                             ERROR+  N               ;STP (STCS BIT1) CANNOT BE CLEARED BY RESET
(1)  006412  004737  023526                     JSR     PC,DMPREG
```

```
870
(4)                                     ;;*****************************************************************
(4)                                     ;*TEST 23       TEST THAT ENB (STCS BIT0) CAN BE SET
(4)                                     ;;*****************************************************************
(3)  006416  000004                     TST23:  SCOPE
(1)  006420  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE INTERFACE
(1)  006424  012777  000003  016300             MOV     #STCSE,@ACR     ;SET ADR TO STCS
(1)  006432  012777  000001  016276             MOV     #ENB,@STCS              ;SET ENB IN STCS
(1)  006440  012777  000003  016264             MOV     #STCSE,@ACR     ;RESET ADR TO STCS
(1)  006446  017737  016264  024750             MOV     @STCS,DISPLY    ;READ STCS
(1)  006454  032737  000001  024750             BIT     #ENB,DISPLY     ;CHECK FOR ENB=1
(3)  006462  001003                             BNE     TST24           ;;SKIP IF ENB IS SET IN STCS
(1)  006464  104027                             ERROR+  N               ;ENB (STCS BIT0) CANNOT BE SET
(1)  006466  004737  023526                     JSR     PC,DMPREG
(1)
(5)                                     ;;*****************************************************************
(4)                                     ;*TEST 24       TEST THAT ENB (STCS BIT0) CAN BE CLEARED
(4)                                     ;;*****************************************************************
(3)  006472  000004                     TST24:  SCOPE
(1)  006474  012777  000003  016230             MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006502  012777  000001  016226             MOV     #ENB,@STCS              ;SET ENB IN STCS
(1)  006510  012777  000003  016214             MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006516  042777  000001  016212             BIC     #ENB,@STCS              ;CLEAR ENB IN STCS
(1)  006524  012777  000003  016200             MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006532  017737  016200  024750             MOV     @STCS,DISPLY    ;READ STCS
(1)  006540  032737  000001  024750             BIT     #ENB,DISPLY     ;CHECK FOR ENB=0
(3)  006546  001403                             BEQ     TST25           ;;SKIP IF ENB IS CLEAR IN STCS
(1)  006550  104030                             ERROR+  N               ;ENB (STCS BIT0) CANNOT BE CLEARED
(1)  006552  004737  023526                     JSR     PC,DMPREG
(1)                                                                     ;BY WRITING IT TO 0
(1)
(5)                                     ;;*****************************************************************
(4)                                     ;*TEST 25       TEST THAT ENB (STCS BIT0) CAN BE CLEARED BY RESET
(4)                                     ;;*****************************************************************
(3)  006556  000004                     TST25:  SCOPE
(1)  006560  012777  000003  016144             MOV     #STCSE,@ACR     ;SET ADDRESS TO STCS
(1)  006566  012777  000001  016142             MOV     #ENB,@STCS              ;SET ENB IN STCS
(1)  006574  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE INTERFACE
(1)  006600  012777  000003  016124             MOV     #STCSE,@ACR     ;SET ADR TO STCS
(1)  006606  017737  016124  024750             MOV     @STCS,DISPLY    ;READ STCS
(1)  006614  032737  000001  024750             BIT     #ENB,DISPLY     ;CHECK FOR ENB=0
(3)  006622  001403                             BEQ     TST26           ;;SKIP IS ENB IS CLEARED BY RESET IN STCS
(1)  006624  104031                             ERROR+  N               ;ENB (STCS BIT0) CANNOT BE CLEARED BY RESET
(1)  006626  004737  023526                     JSR     PC,DMPREG
```

```
872
(4)                                     ;;*******************************************************************
(4)                                     ;*TEST 26       STCS (BIT15-8) WRITE-READ TEST USING STCS COUNT PATTERN
(4)                                     ;;*******************************************************************
(3)  006632  000004                     TST26:  SCOPE
(1)  006634  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  006640  012737  006654  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  006646  012737  177400  001124             MOV     #177400,$GDDAT  ;INIT EXPECTED TO 177400
(1)  006654  012777  000003  016050  1$:        MOV     #STCSE,@ACR     ;SET ADR TO STCS
(1)  006662  013777  001124  016046             MOV     $GDDAT,@STCS    ;LOAD STCS WITH $GDDAT
(1)  006670  012777  000003  016034             MOV     #STCSE,@ACR     ;RESET ADR TO STCS
(1)  006676  017737  016034  024750             MOV     @STCS,DISPLY    ;READ STCS
(1)  006704  013737  024750  001126             MOV     DISPLY,$BDDAT   ;COPY STCS INTO $BDDAT
(1)  006712  042737  000377  001126             BIC     #-<177400+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  006720  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  006726  001403                             BEQ     2$              ;SKIP IF OK
(1)  006730  104032                             ERROR+  N               ;STCS BIT15-8 LOAD-READ ERROR
(1)  006732  004737  023526                     JSR     PC,DMPREG
(1)  006736  062737  000400  001124  2$:        ADD     #BIT8,$GDDAT    ;BUMP EXPECTED
(1)  006744  042737  000377  001124             BIC     #-<177400+1>,$GDDAT     ;MASK TO BITS UNDER TEST
(1)  006752  023727  001124  177400             CMP     $GDDAT,#177400  ;DONE ALL?
(1)  006760  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  006762  012737  006770  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  006770  012777  000003  015734  3$:        MOV     #STCSE,@ACR     ;RESET ADR TO STCS
(1)  006776  012777  177400  015732             MOV     #177400,@STCS           ;SET ALL BITS
(1)  007004  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  007010  012777  000003  015714             MOV     #STCSE,@ACR     ;SET ADDR TO STCS
(1)  007016  017737  015714  024750             MOV     @STCS,DISPLY    ;READ STCS
(1)  007024  032737  177400  024750             BIT     #177400,DISPLY  ;CHECK BIT15-8 IN STCS FOR 0
(1)  007032  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT15-8 IN STCS
(1)  007034  104033                             ERROR+  N               ;BIT15-8 IN STCS WERE NOT CLEARED BY RESET
(1)  007036  004737  023526                     JSR     PC,DMPREG
(1)  007042                             4$:
```

```
874
(4)                                     ;;*****************************************************************
(4)                                     ;*TEST 27       IMSK (BIT11-8) WRITE-READ TEST USING IMSK COUNT PATTERN
(4)                                     ;;*****************************************************************
(3)  007042  000004                     TST27:  SCOPE
(1)  007044  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  007050  012737  007064  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  007056  012737  007400  001124             MOV     #7400,$GDDAT    ;INIT EXPECTED TO 7400
(1)  007064  012777  000004  015640  1$:        MOV     #IMSKE,@ACR     ;SET ADR TO IMSK
(1)  007072  013777  001124  015636             MOV     $GDDAT,@IMSK    ;LOAD IMSK WITH $GDDAT
(1)  007100  012777  000004  015624             MOV     #IMSKE,@ACR     ;RESET ADR TO IMSK
(1)  007106  017737  015624  024750             MOV     @IMSK,DISPLY    ;READ IMSK
(1)  007114  013737  024750  001126             MOV     DISPLY,$BDDAT   ;COPY IMSK INTO $BDDAT
(1)  007122  042737  170377  001126             BIC     #-<7400+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  007130  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  007136  001403                             BEQ     2$              ;SKIP IF OK
(1)  007140  104034                             ERROR+  N               ;IMSK BIT11-8 LOAD-READ ERROR
(1)  007142  004737  023526                     JSR     PC,DMPREG
(1)  007146  062737  000400  001124  2$:        ADD     #BIT8,$GDDAT    ;BUMP EXPECTED
(1)  007154  042737  170377  001124             BIC     #-<7400+1>,$GDDAT       ;MASK TO BITS UNDER TEST
(1)  007162  023727  001124  007400             CMP     $GDDAT,#7400    ;DONE ALL?
(1)  007170  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  007172  012737  007200  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  007200  012777  000004  015524  3$:        MOV     #IMSKE,@ACR     ;RESET ADR TO IMSK
(1)  007206  012777  007400  015522             MOV     #7400,@IMSK             ;SET ALL BITS
(1)  007214  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  007220  012777  000004  015504             MOV     #IMSKE,@ACR     ;SET ADDR TO IMSK
(1)  007226  017737  015504  024750             MOV     @IMSK,DISPLY    ;READ IMSK
(1)  007234  032737  007400  024750             BIT     #7400,DISPLY    ;CHECK BIT11-8 IN IMSK FOR 0
(1)  007242  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT11-8 IN IMSK
(1)  007244  104035                             ERROR+  N               ;BIT11-8 IN IMSK WERE NOT CLEARED BY RESET
(1)  007246  004737  023526                     JSR     PC,DMPREG
(1)  007252                             4$:
```

```
876                                     ;;*****************************************************************
(4)                                     ;*TEST 30       IMSK (BIT3-0) WRITE-READ TEST USING IMSK COUNT PATTERN
(4)                                     ;;*****************************************************************
(3)  007252  000004                     TST30:  SCOPE
(1)  007254  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  007260  012737  007274  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  007266  012737  000017  001124             MOV     #17,$GDDAT      ;INIT EXPECTED TO 17
(1)  007274  012777  000004  015430  1$:        MOV     #IMSKE,@ACR     ;SET ADR TO IMSK
(1)  007302  013777  001124  015426             MOV     $GDDAT,@IMSK    ;LOAD IMSK WITH $GDDAT
(1)  007310  012777  000004  015414             MOV     #IMSKE,@ACR     ;RESET ADR TO IMSK
(1)  007316  017737  015414  024750             MOV     @IMSK,DISPLY    ;READ IMSK
(1)  007324  013737  024750  001126             MOV     DISPLY,$BDDAT   ;COPY IMSK INTO $BDDAT
(1)  007332  042737  177760  001126             BIC     #-<17+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  007340  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  007346  001403                             BEQ     2$              ;SKIP IF OK
(1)  007350  104036                             ERROR+  N               ;IMSK BIT3-0 LOAD-READ ERROR
(1)  007352  004737  023526                     JSR     PC,DMPREG
(1)  007356  062737  000001  001124  2$:        ADD     #BIT0,$GDDAT    ;BUMP EXPECTED
(1)  007364  042737  177760  001124             BIC     #-<17+1>,$GDDAT ;MASK TO BITS UNDER TEST
(1)  007372  023727  001124  000017             CMP     $GDDAT,#17      ;DONE ALL?
(1)  007400  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  007402  012737  007410  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  007410  012777  000004  015314  3$:        MOV     #IMSKE,@ACR     ;RESET ADR TO IMSK
(1)  007416  012777  000017  015312             MOV     #17,@IMSK               ;SET ALL BITS
(1)  007424  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  007430  012777  000004  015274             MOV     #IMSKE,@ACR     ;SET ADDR TO IMSK
(1)  007436  017737  015274  024750             MOV     @IMSK,DISPLY    ;READ IMSK
(1)  007444  032737  000017  024750             BIT     #17,DISPLY      ;CHECK BIT3-0 IN IMSK FOR 0
(1)  007452  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT3-0 IN IMSK
(1)  007454  104037                             ERROR+  N               ;BIT3-0 IN IMSK WERE NOT CLEARED BY RESET
(1)  007456  004737  023526                     JSR     PC,DMPREG
(1)  007462                             4$:
```

```
878                                     ;;*******************************************************************
(4)                                     ;*TEST 31       MTC (BIT11-8) WRITE-READ TEST USING MTC COUNT PATTERN
(4)                                     ;;*******************************************************************
(3)  007462  000004                     TST31:  SCOPE
(1)  007464  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  007470  012737  007504  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  007476  012737  007400  001124             MOV     #7400,$GDDAT    ;INIT EXPECTED TO 7400
(1)  007504  012777  000015  015220  1$:        MOV     #MTCE,@ACR      ;SET ADR TO MTC
(1)  007512  013777  001124  015216             MOV     $GDDAT,@MTC     ;LOAD MTC WITH $GDDAT
(1)  007520  012777  000015  015204             MOV     #MTCE,@ACR      ;RESET ADR TO MTC
(1)  007526  017737  015204  024750             MOV     @MTC,DISPLY     ;READ MTC
(1)  007534  013737  024750  001126             MOV     DISPLY,$BDDAT   ;COPY MTC INTO $BDDAT
(1)  007542  042737  170377  001126             BIC     #-<7400+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  007550  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  007556  001403                             BEQ     2$              ;SKIP IF OK
(1)  007560  104040                             ERROR+  N               ;MTC BIT11-8 LOAD-READ ERROR
(1)  007562  004737  023526                     JSR     PC,DMPREG
(1)  007566  062737  000400  001124  2$:        ADD     #BIT8,$GDDAT    ;BUMP EXPECTED
(1)  007574  042737  170377  001124             BIC     #-<7400+1>,$GDDAT       ;MASK TO BITS UNDER TEST
(1)  007602  023727  001124  007400             CMP     $GDDAT,#7400    ;DONE ALL?
(1)  007610  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  007612  012737  007620  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  007620  012777  000015  015104  3$:        MOV     #MTCE,@ACR      ;RESET ADR TO MTC
(1)  007626  012777  007400  015102             MOV     #7400,@MTC              ;SET ALL BITS
(1)  007634  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  007640  012777  000015  015064             MOV     #MTCE,@ACR      ;SET ADDR TO MTC
(1)  007646  017737  015064  024750             MOV     @MTC,DISPLY     ;READ MTC
(1)  007654  032737  007400  024750             BIT     #7400,DISPLY    ;CHECK BIT11-8 IN MTC FOR 0
(1)  007662  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT11-8 IN MTC
(1)  007664  104041                             ERROR+  N               ;BIT11-8 IN MTC WERE NOT CLEARED BY RESET
(1)  007666  004737  023526                     JSR     PC,DMPREG
(1)  007672                             4$:
```

```
880                                     ;;*****************************************************************
(4)                                     ;*TEST 32       MTC (BIT3-0) WRITE-READ TEST USING MTC COUNT PATTERN
(4)                                     ;;*****************************************************************
(3)  007672  000004                     TST32:  SCOPE
(1)  007674  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  007700  012737  007714  001110             MOV     #1$,$LPERR      ;SET SCOPE LOOP POINTER
(1)  007706  012737  000017  001124             MOV     #17,$GDDAT      ;INIT EXPECTED TO 17
(1)  007714  012777  000015  015010  1$:        MOV     #MTCE,@ACR      ;SET ADR TO MTC
(1)  007722  013777  001124  015006             MOV     $GDDAT,@MTC     ;LOAD MTC WITH $GDDAT
(1)  007730  012777  000015  014774             MOV     #MTCE,@ACR      ;RESET ADR TO MTC
(1)  007736  017737  014774  024750             MOV     @MTC,DISPLY     ;READ MTC
(1)  007744  013737  024750  001126             MOV     DISPLY,$BDDAT   ;COPY MTC INTO $BDDAT
(1)  007752  042737  177760  001126             BIC     #-<17+1>, $BDDAT;CLEAR ALL BUT BITS UNDER TEST
(1)  007760  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK BITS LOADED AGAINST BITS READ
(1)  007766  001403                             BEQ     2$              ;SKIP IF OK
(1)  007770  104042                             ERROR+  N               ;MTC BIT3-0 LOAD-READ ERROR
(1)  007772  004737  023526                     JSR     PC,DMPREG
(1)  007776  062737  000001  001124  2$:        ADD     #BIT0,$GDDAT    ;BUMP EXPECTED
(1)  010004  042737  177760  001124             BIC     #-<17+1>,$GDDAT ;MASK TO BITS UNDER TEST
(1)  010012  023727  001124  000017             CMP     $GDDAT,#17      ;DONE ALL?
(1)  010020  001335                             BNE     1$              ;TEST AGAIN IF NO
(1)  010022  012737  010030  001110             MOV     #3$,$LPERR      ;CHANGE LOOP CONTROL
(1)  010030  012777  000015  014674  3$:        MOV     #MTCE,@ACR      ;RESET ADR TO MTC
(1)  010036  012777  000017  014672             MOV     #17,@MTC                ;SET ALL BITS
(1)  010044  004737  023454                     JSR     PC,CLEAR        ;CLEAR THE WORLD
(1)  010050  012777  000015  014654             MOV     #MTCE,@ACR      ;SET ADDR TO MTC
(1)  010056  017737  014654  024750             MOV     @MTC,DISPLY     ;READ MTC
(1)  010064  032737  000017  024750             BIT     #17,DISPLY      ;CHECK BIT3-0 IN MTC FOR 0
(1)  010072  001403                             BEQ     4$              ;SKIP IF RESET CLEARED BIT3-0 IN MTC
(1)  010074  104043                             ERROR+  N               ;BIT3-0 IN MTC WERE NOT CLEARED BY RESET
(1)  010076  004737  023526                     JSR     PC,DMPREG
(1)  010102                             4$:
881
882  010102  004737  024634                     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.

                                ;;*****************************************************************
                                ;*TEST 33       DUAL ADDRESSING TEST OF ALL REGISTERS
                                ;;*****************************************************************
010106  000004                  TST33:  SCOPE
010110  023727  025020  000003          CMP     CONFIG,#3       ;SEE IF CONFIG #3 (OTHER ACTIVE UNITS)
010116  001002                        BNE     11$             ;SKIP IF NO
010120  000137  010536                JMP     99$             ;DON'T DO TEST IF YES
010124  012705  000020          11$:  MOV     #16.,R5         ;SET COUNTER FOR 16 REGISTERS
010130  012704  000000                MOV     #0,R4           ;SET POINTER FOR REGISTER ADDRESS
010134  012703  024756                MOV     #SNAP0,R3       ;SET POINTER FOR INITIAL REGISTER DATA
010140  004737  023450                JSR     PC,CLEARX       ;CLEAR THE IIST
010144  005037  001176                CLR     $TMP0           ;CLEAR THE REGISTER NUMBER
010150  110477  014556          1$:   MOVB    R4,@ACR         ;SELECT REGISTER FROM TABLE
010154  017723  014556                MOV     @ADR,(3)+       ;PUT REGISTER CONTENTS INTO SNAP
010160  005204                        INC     R4              ;BUMP ADR CODE
010162  005305                        DEC     R5              ;DONE 10?
010164  001371                        BNE     1$              ;DO 10
010166  005000                        CLR     R0              ;CLEAR TEST DATA WORD
010170  010077  014536          2$:   MOV     R0,@ACR         ;LOAD DATA INTO REGISTER
010174  005200                        INC     R0              ;BUMP DATA
010176  100374                        BPL     2$              ;DO 0-77777. (DO NOT SET "CLR")
010200  012705  000020                MOV     #16.,R5         ;SET COUNTER FOR 16 REGISTERS
010204  012704  000000                MOV     #0,R4           ;SET POINTER FOR REGISTER ADDRESS TABLE
010210  012703  024756                MOV     #SNAP0,R3       ;SET POINTER FOR INITIAL REGISTER DATA
010214  110437  001176          3$:   MOVB    R4,$TMP0        ;SELECT REGISTER FROM TABLE
010220  013777  001176  014504        MOV     $TMP0,@ACR      ;LOAD ACR WITH ADDRESS
010226  017737  014504  001126        MOV     @ADR, $BDDAT    ;READ REGISTER INTO $BDDAT
010234  012337  001124                MOV     (3)+,$GDDAT     ;FETCH ORIGINAL REGISTER DATA
010240  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;CHECK FOR ANY CHANGE IN REGISTER
010246  001403                        BEQ     4$              ;SKIP IF NO CHANGE
010250  104044                        ERROR+  44              ;ADDRESSING ACR ALSO ADDRESSED REGISTER IN $TMP0
010252  004737  023526                JSR     PC,DMPREG
010256  005204                  4$:   INC     R4              ;BUMP ADR CODE
010260  005305                        DEC     R5              ;CHECKED ALL REGISTERS?
010262  001354                        BNE     3$              ;CHECK ALL REGISTERS
010264  012705  000020                MOV     #16.,R5         ;SET COUNTER FOR 16 REGISTERS
010270  012704  000000                MOV     #0,R4           ;SET POINTER FOR REGISTER ADDRESS
010274  012737  010306  001110        MOV     #5$,$LPERR      ;SET SCOPE LOOP POINT
010302  005037  001204                CLR     $TMP3           ;INITIALIZE $TMP3
010306  012777  100000  014416  5$:   MOV     #CLR,@ACR       ;CLEAR THE IIST
010314  004737  023450                JSR     PC,CLEARX
010320  012703  024756                MOV     #SNAP0,R3       ;SET POINTER FOR INITIAL REGISTER DATA
010324  012701  000020                MOV     #16.,R1         ;SET COUNTER TO READ 16 REGISTERS
010330  012702  000000                MOV     #0,R2           ;SET POINTER FOR REGISTER ADDRESS
010334  110277  014372          6$:   MOVB    R2,@ACR         ;SELECT REGISTER
010340  017723  014372                MOV     @ADR,(3)+       ;READ INITIAL REGISTER DATA
010344  005202                        INC     R2              ;BUMP ADR CODE
010346  005301                        DEC     R1              ;DONE 16?
```

```
   937  010350  001371                    BNE     6$              ;DO 16
   938  010352  110437  001176            MOVB    R4,$TMPO        ;$TMPO=REGISTER BEING WRITTEN
   939  010356  005000                    CLR     R0              ;START AT 0
```

```
                                                                    ;SELECT REGISTER
                                                                    ;COPY R0
                                                                    ;PGCS?
                                                                    ;SKIP IF YES
                                                                    ;STCS?
                                                                    ;SKIP IF YES
                                                                    ;MTC?
                                                                    ;SKIP IF NO
                                                                    ;CLEAR MAINT BITS
                                                                    ;SKIP
                                                                    ;DO NOT SET GO
                                                                    ;LOAD DATA INTO REGISTER
                                                                    ;BUMP DATA
                                                                    ;LOAD 0-77777 INTO REGISTER
                                                                    ;SET POINTER FOR REGISTER ADDRESS TABLE
                                                                    ;SET COUNTER FOR 16 REGISTERS
                                                                    ;SET POINTER FOR INITIAL CONTENTS TABLE
                                                                    ;SELECT A REGISTER
                                                                    ;FETCH INITIAL CONTENTS OF REGISTER
                                                                    ;IS SELECTED REGISTER THE ONE WRITTEN?
                                                                    ;SKIP TESTING IF YES
                                                                    ;SAVE ADDRESS OF REGISTER CHECKED IN $TMP1
                                                                    ;FETCH REGISTER DATA
                                                                    ;CHECK REGISTER FOR ANY CHANGES
                                                                    ;SKIP IF NO CHANGE
                                                                    ;ADDRESSING REGISTER IN $TMP0 ALSO
                                                                    ;ADDRESSED REGISTER IN $TMP1
                                                                    ;BUMP ADR CODE
                                                                    ;CHECKED ALL REGISTERS?
                                                                    ;CHECK ALL REGISTERS
                                                                    ;BUMP POINTER FOR REGISTER ADDRESS
                                                                    ;DONE ALL?
                                                                    ;DO ALL
```

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

010536  012704  000000          LINE0:  MOV     #0,R4           ;SET INDEX TO 0
010542  004737  010626                  JSR     PC,LINTST       ;TEST LINE
010546  012704  000001                  MOV     #1,R4           ;SET INDEX TO 1
010552  004737  010626                  JSR     PC,LINTST       ;TEST LINE
010556  012704  000002                  MOV     #2,R4           ;SET INDEX TO 2
010562  004737  010626                  JSR     PC,LINTST       ;TEST LINE
010566  012704  000003                  MOV     #3,R4           ;SET INDEX TO 3
010572  004737  010626                  JSR     PC,LINTST       ;TEST LINE
010576  013704  024730          LINE1:  MOV     SIDNUM,R4       ;SET INDEX TO ID #
010602  006304                          ASL     R4              ;SHIFT
010604  004737  017400                  JSR     PC,PGONL        ;TEST PGF ONLINE
010610  013704  024730          LINE2:  MOV     SIDNUM,R4       ;SET INDEX TO ID #
010614  006304                          ASL     R4              ;SHIFT
010616  004737  021134                  JSR     PC,STONL        ;TEST STF ONLINE
010622  000137  022012                  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

                                                ;;*****************************************************************
                                                ;*TEST 34       CHECK A LEGAL PGM PI REQUEST FOR PROPER OPERATION
                                                ;;*****************************************************************
(2)  010634  000004                             TST34:  SCOPE
1009 010636  005037  177776                             CLR     PSW             ;LOWER PRIORITY TO ALLOW INTERUPTS
1010 010642  004737  023306                             JSR     PC,SETMNT       ;ENABLE MAINT LOOPBACK
1011 010646  012777  000000  014056                     MOV     #PGTEE,@ACR     ;SELECT PGTE
1012 010654  016477  021772  014054                     MOV     PGITAB(4),@PGTE ;LOAD PIE
1013 010662  012777  000001  014046                     MOV     #GO,@PGCS       ;TRANSMIT
1014 010670  012777  000001  014034                     MOV     #PGCSE,@ACR     ;SELECT PGCS
1015 010676  017737  014034  001126                     MOV     @PGCS,$BDDAT    ;READ PGCS
1016 010704  032737  004000  001126                     BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
1017 010712  001406                                     BEQ     1$              ;=0
1018 010714  016437  021762  001124                     MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED PGCS
1019 010722  104044                                     ERROR+  44              ;SETTING PGCS GO DID NOT CLEAR RDY
1020 010724  004737  023526                             JSR     PC,DMPREG
1021 010730  005000                             1$:     CLR     R0              ;INITIALIZE TIMER
1022 010732  012777  000001  013772             2$:     MOV     #PGCSE,@ACR     ;SELECT PGCS
1023 010740  017737  013772  001126                     MOV     @PGCS,$BDDAT    ;READ PGCS
1024 010746  032737  004000  001126                     BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
1025 010754  001013                                     BNE     3$              ;SKIP IF SET
1026 010756  005300                                     DEC     R0              ;TIMER =0?
1027 010760  001364                                     BNE     2$              ;TRY AGAIN IF NO
1028 010762  016437  021762  001124                     MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED WITH
1029 010770  062737  004000  001124                     ADD     #PRDY,$GDDAT    ;SID + RDY
1030 010776  104044                                     ERROR+  44              ;PGCS RDY DID NOT SET ON A PGM PI
1031 011000  004737  023526                             JSR     PC,DMPREG
1032 011004  012777  000005  013720             3$:     MOV     #PGFE,@ACR      ;SELECT PGF
1033 011012  017737  013720  001126                     MOV     @PGF,$BDDAT     ;READ PGF
1034 011020  012737  000017  001124                     MOV     #17,$GDDAT      ;FETCH EXPECTED FLAG BITS
1035 011026  023737  001124  001126                     CMP     $GDDAT,$BDDAT   ;CHECK FOR CORRECT FLAG
1036 011034  001403                                     BEQ     4$              ;SKIP IF OK
1037 011036  104044                                     ERROR+  44              ;PGM PI DID NOT SET CORRECT FLAG
1038 011040  004737  023526                             JSR     PC,DMPREG
1039 011044  012777  000007  013660             4$:     MOV     #DCFE,@ACR      ;SELECT DCF
1040 011052  017737  013660  024750                     MOV     @DCF,DISPLY     ;READ DCF
1041 011060  001403                                     BEQ     5$              ;SKIP IF 0
1042 011062  104044                                     ERROR+  44              ;PGM PI SET A DCF CONDITION
1043 011064  004737  023526                             JSR     PC,DMPREG
1044 011070  012777  000010  013634             5$:     MOV     #EXCE,@ACR              ;SELECT EXC
1045 011076  017737  013634  024750                     MOV     @EXC,DISPLY     ;READ EXC
1046 011104  001403                                     BEQ     6$              ;SKIP IF 0
1047 011106  104044                                     ERROR+  44              ;PGM PI SET AN EXC CONDITION
1048 011110  004737  023526                             JSR     PC,DMPREG
1049 011114  012777  000006  013610             6$:     MOV     #STFE,@ACR      ;SELECT STF
1050 011122  017737  013610  024750                     MOV     @STF,DISPLY     ;READ STF
```

```
  1051  011130  001403                    BEQ     7$              ;SKIP IF CLEAR
  1052  011132  104044                    ERROR+  44              ;PGM PI SET AN STF CONDITION
  1053  011134  004737  023526            JSR     PC,DMPREG
```

```
1055  011140  012700  000017          7$:   MOV   #17,R0          ;SET ALL 4 LINES
1056  011144  046400  021772                BIC   PGITAB(4),R0    ;CLEAR LINE UNDER TEST
1057  011150  012777  000005  013554        MOV   #PGFE,@ACR      ;SELECT PGF
1058  011156  010077  013554                MOV   R0,@PGF         ;CLEAR ALL PIFS EXCEPT LINE UNDER TEST
1059  011162  012777  000005  013542        MOV   #PGFE,@ACR      ;SELECT PGF
1060  011170  017737  013542  001202        MOV   @PGF,$TMP2      ;READ PGF
1061  011176  012777  000001  013526        MOV   #PGCSE,@ACR     ;SELECT PGCS
1062  011204  017737  013526  024750        MOV   @PGCS,DISPLY    ;READ PGCS
1063  011212  032737  000010  024750        BIT   #IP,DISPLY      ;CHECK FOR IP
1064  011220  001003                        BNE   8$              ;SKIP IF SET
1065  011222  104044                        ERROR+ 44             ;IP DID NOT SET ON PGF
1066  011224  004737  023526                JSR   PC,DMPREG
1067  011230  017746  013510          8$:   MOV   @#IIIP,-(SP)   ;SAVE OLD
1068  011234  017746  013502                MOV   @#IIIV,-(SP)    ;PI INFORMATION
1069  011240  012777  011272  013474        MOV   #9$,@#IIIV      ;LOAD NEW PI VECTOR
1070  011246  012777  000340  013470        MOV   #340,@#IIIP     ;LOAD NEW INT. LEVEL
1071  011254  005037  177776                CLR   PS              ;ALLOW PI
1072  011260  000240                        NOP                   ;STALL
1073  011262  012737  000340  177776        MOV   #340,PS         ;LOCKOUT PI
1074  011270  000404                        BR    10$             ;SKIP OVER ERROR
1075  011272  022626          9$:   POPPOP                        ;FIX STACK
1076  011274  104044                        ERROR+ 44             ;PGF PI WITH IE=0
1077  011276  004737  023526                JSR   PC,DMPREG
1078  011302  012777  011376  013432  10$:   MOV   #12$,@#IIIV     ;LOAD NEW PI VECTOR
1079  011310  012777  000001  013414        MOV   #PGCSE,@ACR     ;SELECT PGCS
1080  011316  012777  000004  013412        MOV   #IE,@PGCS       ;SET IE
1081  011324  012737  000340  001212        MOV   #340,$TMP6      ;INIT PI LEVEL
1082  011332  162737  000040  001212  11$:   SUB   #40,$TMP6      ;DROP PI LEVEL BY 1
1083  011340  013737  001212  177776        MOV   $TMP6,PS        ;LOWER PSW
1084  011346  000240                        NOP                   ;STALL
1085  011350  023727  001212  000140        CMP   $TMP6,#140      ;TRIED ALL LEVELS?
1086  011356  001365                        BNE   11$             ;BR IF NO
1087  011360  012737  000340  177776        MOV   #340,PS         ;LOCKOUT PI
1088  011366  104044                        ERROR+ 44             ;NO PGF PI WITH IE=1
1089  011370  004737  023526                JSR   PC,DMPREG
1090  011374  000422                        BR    13$             ;SKIP NEXT TEST
1091  011376  022626          12$:  POPPOP                        ;FIX STACK
1092  011400  006337  001212                ASL   $TMP6           ;RIGHT JUSTIFY
1093  011404  006337  001212                ASL   $TMP6
1094  011410  006337  001212                ASL   $TMP6
1095  011414  000337  001212                SWAB  $TMP6           ;THE PI LEVEL
1096  011420  005237  001212                INC   $TMP6           ;ADD 1 TO MAKE CORECT LEVEL
1097  011424  023737  001212  024746        CMP   $TMP6,IIIL      ;CHECK AGAINST KNOWN LEVEL
1098  011432  001403                        BEQ   13$             ;SKIP IF OK
1099  011434  104044                        ERROR+ 44             ;PGF PI TO THE WRONG LEVEL
1100  011436  004737  023526                JSR   PC,DMPREG
1101  011442  012677  013274          13$:  MOV   (SP)+,@#IIIV    ;RESTORE
1102  011446  012677  013272                MOV   (SP)+,@#IIIP    ;OLD PI INFO
1103  011452  012777  000005  013252  14$:   MOV   #PGFE,@ACR      ;SELECT PGF
1104  011460  016437  021772  001124        MOV   PIFTAB(4),$GDDAT;FETCH FLAG BIT
1105  011466  013777  001124  013242        MOV   $GDDAT,@PGF     ;CLEAR FLAG
1106  011474  005037  001124                CLR   $GDDAT          ;CLEAR EXPECTED
1107  011500  012777  000005  013224        MOV   #PGFE,@ACR      ;SELECT PGF
1108  011506  017737  013224  001126        MOV   @PGF,$BDDAT     ;READ PGF
1109  011514  001403                        BEQ   15$             ;SKIP IF OK
1110  011516  104044                        ERROR+ 44             ;WRITING A 1 TO PGF FLAG DID NOT
```

```
 1111  011520  004737  023526                JSR     PC,DMPREG
 1112                                                        ;CLEAR IT
 1113  011524                        15$:
```

```
                                ;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

                                ;;*****************************************************************
                                ;*TEST 35       CHECK A LEGAL PGM BOOT REQUEST FOR PROPER OPERATION
                                ;;*****************************************************************
 (2) 011524  000004             TST35:  SCOPE
1122 011526  004737  023306             JSR     PC,SETMNT       ;ENABLE MAINT LOOPBACK
1123 011532  012777  000000  013172     MOV     #PGTEE,@ACR     ;SELECT PGTE
1124 011540  016477  022002  013170     MOV     PGBTAB(4),@PGTE ;LOAD PIB
1125 011546  012777  000001  013162     MOV     #GO,@PGCS       ;TRANSMIT
1126 011554  005000                1$:  CLR     R0              ;INITIALIZE TIMER
1127 011556  012777  000001  013146  2$: MOV     #PGCSE,@ACR     ;SELECT PGCS
1128 011564  017737  013146  001126     MOV     @PGCS,$BDDAT    ;READ PGCS
1129 011572  032737  004000  001126     BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
1130 011600  001013                     BNE     3$              ;SKIP IF SET
1131 011602  005300                     DEC     R0              ;TIMER =0?
1132 011604  001364                     BNE     2$              ;TRY AGAIN IF NO
1133 011606  016437  021762  001124     MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED WITH
1134 011614  062737  004000  001124     ADD     #PRDY,$GDDAT    ;SID + RDY
1135 011622  104044                     ERROR+  44              ;PGCS RDY DID NOT SET ON A PGM BOOT
1136 011624  004737  023526             JSR     PC,DMPREG
1137 011630  012777  000005  013074  3$: MOV     #PGFE,@ACR      ;SELECT PGF
1138 011636  017737  013074  001126     MOV     @PGF,$BDDAT     ;READ PGF
1139 011644  012737  007400  001124     MOV     #7400,$GDDAT    ;FETCH EXPECTED FLAG BITS
1140 011652  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;CHECK FOR CORRECT FLAG
1141 011660  001403                     BEQ     4$              ;SKIP IF OK
1142 011662  104044                     ERROR+  44              ;PGM BOOT DID NOT SET CORRECT FLAG
1143 011664  004737  023526             JSR     PC,DMPREG
1144 011670  012777  000007  013034  4$: MOV     #DCFE,@ACR      ;SELECT DCF
1145 011676  017737  013034  024750     MOV     @DCF,DISPLY     ;READ DCF
1146 011704  001403                     BEQ     5$              ;SKIP IF 0
1147 011706  104044                     ERROR+  44              ;PGM BOOT SET A DCF CONDITION
1148 011710  004737  023526             JSR     PC,DMPREG
1149 011714  012777  000010  013010  5$: MOV     #EXCE,@ACR      ;SELECT EXC
1150 011722  017737  013010  024750     MOV     @EXC,DISPLY     ;READ EXC
1151 011730  001403                     BEQ     6$              ;SKIP IF 0
1152 011732  104044                     ERROR+  44              ;PGM BOOT SET AN EXC CONDITION
1153 011734  004737  023526             JSR     PC,DMPREG
1154 011740  012777  000006  012764  6$: MOV     #STFE,@ACR      ;SELECT STF
1155 011746  017737  012764  024750     MOV     @STF,DISPLY     ;READ STF
1156 011754  001403                     BEQ     7$              ;SKIP IF 0
1157 011756  104044                     ERROR+  44              ;PGM BOOT SET AN STF CONDITION
1158 011760  004737  023526             JSR     PC,DMPREG
1159 011764  012777  000005  012740  7$: MOV     #PGFE,@ACR      ;SELECT PGF
1160 011772  016437  022002  001124     MOV     PBFTAB(4),$GDDAT;FETCH FLAG BIT
1161 012000  013777  001124  012730     MOV     $GDDAT,@PGF     ;CLEAR FLAG
1162 012006  005137  001124             COM     $GDDAT          ;FLIP GDDAT
1163 012012  042737  170377  001124     BIC     #170377,$GDDAT  ;MASK OUT JUNK
1164 012020  012777  000005  012704     MOV     #PGFE,@ACR      ;SELECT PGF
1165 012026  017737  012704  001126     MOV     @PGF,$BDDAT     ;READ PGF
1166 012034  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
1167 012042  001403                     BEQ     8$              ;SKIP IF OK
```

```
 1168  012044  104044                          ERROR+  44              ;WRITING A 1 TO PGF FLAG DID NOT
 1169  012046  004737  023526                   JSR     PC,DMPREG
 1170                                                                  ;CLEAR IT
 1171  012052                            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

                                        ;;*****************************************************************
  (3)                                   ;*TEST 36       CHECK A LEGAL ST INTR REQUEST FOR PROPER OPERATION
  (3)                                   ;;*****************************************************************
  (2) 012052 000004                     TST36:  SCOPE
1183  012054 004737 023476               JSR     PC,SSTMNT       ;ENABLE MAINT LOOPBACK
1184  012060 012777 000000 012644       MOV     #PGTEE,@ACR     ;SELECT PGTE
1185  012066 016477 021772 012642       MOV     PGITAB(4),@PGTE ;LOAD PIE
1186  012074 012777 000003 012634       MOV     #PTP+GO,@PGCS   ;TRANSMIT
1187  012102 012777 000001 012622       MOV     #PGCSE,@ACR     ;SELECT PGCS
1188  012110 017737 012622 001126       MOV     @PGCS,$BDDAT    ;READ PGCS
1189  012116 032737 004000 001126       BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
1190  012124 001406                     BEQ     1$              ;=0
1191  012126 016437 021762 001124       MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED PGCS
1192  012134 104044                     ERROR+  44              ;SETTING PGCS GO DID NOT CLEAR RDY
1193  012136 004737 023526               JSR     PC,DMPREG
1194  012142 005000                     1$:     CLR     R0              ;INITIALIZE TIMER
1195  012144 012777 000001 012560       2$:     MOV     #PGCSE,@ACR     ;SELECT PGCS
1196  012152 017737 012560 001126               MOV     @PGCS,$BDDAT    ;READ PGCS
1197  012160 032737 004000 001126               BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
1198  012166 001013                             BNE     3$              ;SKIP IF SET
1199  012170 005300                             DEC     R0              ;TIMER =0?
1200  012172 001364                             BNE     2$              ;TRY AGAIN IF NO
1201  012174 016437 021762 001124               MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED WITH
1202  012202 062737 004000 001124               ADD     #PRDY,$GDDAT    ;SID + RDY
1203  012210 104044                             ERROR+  44              ;PGCS RDY DID NOT SET ON A PGM PI
1204  012212 004737 023526                      JSR     PC,DMPREG
1205  012216 012777 000006 012506       3$:     MOV     #STFE,@ACR      ;SELECT STF
1206  012224 017737 012506 001126               MOV     @STF,$BDDAT     ;READ STF
1207  012232 012737 000017 001124               MOV     #17,$GDDAT      ;FETCH EXPECTED FLAG BITS
1208  012240 023737 001124 001126               CMP     $GDDAT,$BDDAT   ;CHECK FOR CORRECT FLAG
1209  012246 001403                             BEQ     4$              ;SKIP IF OK
1210  012250 104044                             ERROR+  44              ;ST INTR DID NOT SET CORRECT FLAG
1211  012252 004737 023526                      JSR     PC,DMPREG
1212  012256 012777 000007 012446       4$:     MOV     #DCFE,@ACR      ;SELECT DCF
1213  012264 017737 012446 024750               MOV     @DCF,DISPLY     ;READ DCF
1214  012272 001403                             BEQ     5$              ;SKIP IF 0
1215  012274 104044                             ERROR+  44              ;ST INTR SET A DCF CONDITION
1216  012276 004737 023526                      JSR     PC,DMPREG
1217  012302 012777 000010 012422       5$:     MOV #EXCE,@ACR          ;SELECT EXC
1218  012310 017737 012422 024750               MOV     @EXC,DISPLY     ;READ EXC
1219  012316 001403                             BEQ     6$              ;SKIP IF 0
1220  012320 104044                             ERROR+  44              ;ST INTR SET AN EXC CONDITION
1221  012322 004737 023526                      JSR     PC,DMPREG
1222  012326 012777 000005 012376       6$:     MOV     #PGFE,@ACR      ;SELECT PGF
1223  012334 017737 012376 024750               MOV     @PGF,DISPLY     ;READ PGF
1224  012342 001403                             BEQ     7$              ;SKIP IF CLEAR
1225  012344 104044                             ERROR+  44              ;ST INTR SET A PGF CONDITION
```

```
 1226  012346  004737  023526                JSR     PC,DMPREG
```

```
1228  012352  012700  000017          7$:     MOV     #17,R0          ;SET ALL 4 LINES
1229  012356  046400  021772                  BIC     STIIAB(4),R0    ;CLEAR LINE UNDER TEST
1230  012362  012777  000006  012342          MOV     #STFE,@ACR      ;SELECT STF
1231  012370  010077  012342                  MOV     R0,@STF         ;CLEAR ALL SIFS EXCEPT LINE UNDER TEST
1232  012374  012777  000006  012330          MOV     #STFE,@ACR      ;SELECT STF
1233  012402  017737  012330  001202          MOV     @STF,$TMP2      ;READ STF
1234  012410  012777  000001  012314          MOV     #PGCSE,@ACR     ;SELECT PGCS
1235  012416  017737  012314  024750          MOV     @PGCS,DISPLY    ;READ PGCS
1236  012424  032737  000010  024750          BIT     #IP,DISPLY      ;CHECK FOR IP
1237  012432  001003                          BNE     8$              ;SKIP IF SET
1238  012434  104044                          ERROR+  44              ;IP DID NOT SET ON STF
1239  012436  004737  023526                  JSR     PC,DMPREG
1240  012442  017746  012276          8$:     MOV     @1111P,-(SP)    ;SAVE OLD
1241  012446  017746  012270                  MOV     @1111V,-(SP)    ;PI INFORMATION
1242  012452  012777  012504  012262          MOV     #9$,@1111V      ;LOAD NEW PI VECTOR
1243  012460  012777  000340  012256          MOV     #340,@1111P     ;LOAD NEW INT. LEVEL
1244  012466  005037  177776                  CLR     PS              ;ALLOW PI
1245  012472  000240                          NOP                     ;STALL
1246  012474  012737  000340  177776          MOV     #340,PS         ;LOCKOUT PI
1247  012502  000404                          BR      10$             ;SKIP OVER ERROR
1248  012504  022626          9$:     POPPOP                          ;FIX STACK
1249  012506  104044                          ERROR+  44              ;STF PI WITH IE=0
1250  012510  004737  023526                  JSR     PC,DMPREG
1251  012514  012777  012562  012220  10$:     MOV     #11$,@1111V     ;LOAD NEW PI VECTOR
1252  012522  012777  000001  012202          MOV     #PGCSE,@ACR     ;SELECT PGCS
1253  012530  012777  000004  012200          MOV     #IE,@PGCS       ;SET IE
1254  012536  005037  177776                  CLR     PS              ;ALLOW PI
1255  012542  000240                          NOP                     ;STALL
1256  012544  012737  000340  177776          MOV     #340,PS         ;LOCKOUT PI
1257  012552  104044                          ERROR+  44              ;NO STF PI WITH IE=1
1258  012554  004737  023526                  JSR     PC,DMPREG
1259  012560  000401                          BR      12$             ;SKIP NEXT TEST
1260  012562  022626          11$:    POPPOP                          ;FIX STACK
1261  012564  012677  012152          12$:    MOV     (SP)+,@1111V    ;RESTORE
1262  012570  012677  012150                  MOV     (SP)+,@1111P    ;OLD PI INFO
1263  012574  012777  000006  012130  13$:     MOV     #STFE,@ACR      ;SELECT STF
1264  012602  016437  021772  001124          MOV     SIFTAB(4),$GDDAT;FETCH FLAG BIT
1265  012610  013777  001124  012120          MOV     $GDDAT,@STF     ;CLEAR FLAG
1266  012616  005037  001124                  CLR     $GDDAT          ;CLEAR EXPECTED
1267  012622  012777  000006  012102          MOV     #STFE,@ACR      ;SELECT STF
1268  012630  017737  012102  001126          MOV     @STF,$BDDAT     ;READ STF
1269  012636  001403                          BEQ     14$             ;SKIP IF OK
1270  012640  104044                          ERROR+  44              ;WRITING A 1 TO STF FLAG DID NOT
1271  012642  004737  023526                  JSR     PC,DMPREG
1272                                                                  ;CLEAR IT
1273  012646                          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 37       CHECK A LEGAL ST BOOT REQUEST FOR PROPER OPERATION
                                        ;;*****************************************************************
012646  000004                  TST37:  SCOPE
012650  004737  023476                  JSR     PC,SSTMNT       ;ENABLE MAINT LOOPBACK
012654  012777  000000  012050          MOV     #PGTEE,@ACR     ;SELECT PGTE
012662  016477  022002  012046          MOV     PGBTAB(4),@PGTE ;LOAD PIB
012670  012777  000003  012040          MOV     #PTP+GO,@PGCS   ;TRANSMIT
012676  005000                  1$:     CLR     R0              ;INITIALIZE TIMER
012700  012777  000001  012024  2$:     MOV     #PGCSE,@ACR     ;SELECT PGCS
012706  017737  012024  001126          MOV     @PGCS,$BDDAT    ;READ PGCS
012714  032737  004000  001126          BIT     #PRDY,$BDDAT    ;CHECK FOR RDY
012722  001013                          BNE     3$              ;SKIP IF SET
012724  005300                          DEC     R0              ;TIMER =0?
012726  001364                          BNE     2$              ;TRY AGAIN IF NO
012730  016437  021762  001124          MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED WITH
012736  062737  004000  001124          ADD     #PRDY,$GDDAT    ;SID + RDY
012744  104044                          ERROR+  44              ;PGCS RDY DID NOT SET ON A ST BOOT
012746  004737  023526                  JSR     PC,DMPREG
012752  012777  000006  011752  3$:     MOV     #STFE,@ACR      ;SELECT STF
012760  017737  011752  001126          MOV     @STF,$BDDAT     ;READ STF
012766  012737  007400  001124          MOV     #7400,$GDDAT    ;FETCH EXPECTED FLAG BITS
012774  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK FOR CORRECT FLAG
013002  001403                          BEQ     4$              ;SKIP IF OK
013004  104044                          ERROR+  44              ;ST BOOT DID NOT SET CORRECT FLAG
013006  004737  023526                  JSR     PC,DMPREG
013012  012777  000007  011712  4$:     MOV     #DCFE,@ACR      ;SELECT DCF
013020  017737  011712  024750          MOV     @DCF,DISPLY     ;READ DCF
013026  001403                          BEQ     5$              ;SKIP IF 0
013030  104044                          ERROR+  44              ;ST BOOT SET A DCF CONDITION
013032  004737  023526                  JSR     PC,DMPREG
013036  012777  000010  011666  5$:     MOV     #EXCE,@ACR      ;SELECT EXC
013044  017737  011666  024750          MOV     @EXC,DISPLY     ;READ EXC
013052  001403                          BEQ     6$              ;SKIP IF 0
013054  104044                          ERROR+  44              ;ST BOOT SET AN EXC CONDITION
013056  004737  023526                  JSR     PC,DMPREG
013062  012777  000005  011642  6$:     MOV     #PGFE,@ACR      ;SELECT PGF
013070  017737  011642  024750          MOV     @PGF,DISPLY     ;READ PGF
013076  001403                          BEQ     7$              ;SKIP IF 0
013100  104044                          ERROR+  44              ;ST BOOT SET A PGF CONDITION
013102  004737  023526                  JSR     PC,DMPREG
013106  012777  000006  011616  7$:     MOV     #STFE,@ACR      ;SELECT STF
013114  016437  022002  001124          MOV     SBFTAB(4),$GDDAT;FETCH FLAG BIT
013122  013777  001124  011606          MOV     $GDDAT,@STF     ;CLEAR FLAG
013130  005137  001124                  COM     $GDDAT          ;FLIP GDDAT
013134  042737  170377  001124          BIC     #170377,$GDDAT  ;MASK OUT JUNK
013142  012777  000006  011562          MOV     #STFE,@ACR      ;SELECT STF
013150  017737  011562  001126          MOV     @STF,$BDDAT     ;READ STF
013156  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
013164  001403                          BEQ     8$              ;SKIP IF OK
```

```
1328  013166  104044                        ERROR+  44              ;WRITING A 1 TO STF FLAG DID NOT
1329  013170  004737  023526                JSR     PC,DMPREG
1330                                                                 ;CLEAR IT
1331  013174                          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

                                ;;*****************************************************************
 (3)                            ;*TEST 40       CHECK THE IMSK BIT FOR PROPER OPERATION
 (3)                            ;;*****************************************************************
 (2) 013174 000004              TST40:  SCOPE
1339 013176 004737 023306               JSR     PC,SETMNT       ;ENABLE MAINT. LOOP
1340 013202 012777 000000 011522        MOV     #PGTEE,@ACR     ;SELECT PGTE
1341 013210 016477 021772 011520        MOV     PGITAB(4),@PGTE ;LOAD PIB
1342 013216 012777 000004 011506        MOV     #IMSKE,@ACR     ;SELECT IMSK
1343 013224 016477 021772 011504        MOV     IMITAB(4),@IMSK;INHIBIT LINE
1344 013232 012777 000001 011472        MOV     #PGCSE,@ACR     ;SELECT PGCS
1345 013240 012777 000001 011470        MOV     #GO,@PGCS       ;TRANSMIT
1346 013246 012777 000001 011456 1$:    MOV     #PGCSE,@ACR     ;SELECT PGCS
1347 013254 032777 004000 011454        BIT     #PRDY,@PGCS     ;CHECK FOR
1348 013262 001771                      BEQ     1$              ;READY
1349 013264 012777 000005 011440        MOV     #PGFE,@ACR      ;SELECT PGF
1350 013272 017737 011440 001126        MOV     @PGF,$BDDAT     ;READ PGF
1351 013300 012737 000017 001124        MOV     #17,$GDDAT      ;FETCH EXPECTED FLAGS
1352 013306 046437 021772 001124        BIC     PGITAB(4),$GDDAT;MINUS EXPECTED FLAG
1353 013314 023737 001124 001126        CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
1354 013322 001403                      BEQ     2$              ;SKIP IF OK
1355 013324 104044                      ERROR+  44              ;IMSK BIT DID NOT INHIBIT ITS PGF BIT
1356 013326 004737 023526               JSR     PC,DMPREG
1357 013332 016437 022002 001124 2$:    MOV     UVITAB(4),$GDDAT;LOAD EXPECTED UI BIT
1358 013340 012777 000010 011364        MOV     #EXCE,@ACR      ;SELECT EXC
1359 013346 017737 011364 001126        MOV     @EXC,$BDDAT     ;READ EXC
1360 013354 023737 001124 001126        CMP     $GDDAT,$BDDAT   ;CHECK THE UI BIT
1361 013362 001403                      BEQ     3$              ;SKIP IF SET
1362 013364 104044                      ERROR+  44              ;UI DID NOT SET ON A PIF WITH IMSK=1
1363 013366 004737 023526               JSR     PC,DMPREG
1364 013372 005037 001124       3$:     CLR     $GDDAT          ;EXPECT 0
1365 013376 012777 000010 011326        MOV     #EXCE,@ACR      ;SELECT EXC
1366 013404 016477 022002 011324        MOV     UVITAB(4),@EXC  ;CLEAR UI BIT
1367 013412 012777 000010 011312        MOV     #EXCE,@ACR      ;SELECT EXC
1368 013420 017737 011312 024750        MOV     @EXC,DISPLY     ;READ EXC
1369 013426 013737 024750 001126        MOV     DISPLY,$BDDAT   ;TEST EXC
1370 013434 001403                      BEQ     4$              ;SKIP IF 0
1371 013436 104044                      ERROR+  44              ;WRITING A 1 TO UI BIT DID NOT CLEAR IT
1372 013440 004737 023526               JSR     PC,DMPREG
1373 013444 012777 000006 011260 4$:    MOV     #STFE,@ACR      ;SELECT STF REGISTER
1374 013452 017737 011260 001126        MOV     @STF,$BDDAT     ;GET AND TEST STF REGISTER
1375 013460 001403                      BEQ     5$              ;STF SHOULD BE 0
1376 013462 104044                      ERROR+  44              ;STF NOT ZERO ON A PGI WITH IMSK=1
1377 013464 004737 023526               JSR     PC,DMPREG
1378 013470                     5$:
```

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

                                        ;;*****************************************************************
                                        ;*TEST 41       CHECK THE IMSK BIT FOR PROPER OPERATION ON PGM BOOT
                                        ;;*****************************************************************
013470  000004                  TST41:  SCOPE
013472  004737  023306                  JSR     PC,SETMNT       ;ENABLE MAINT. LOOP
013476  012777  000000  011226          MOV     #PGTEE,@ACR     ;SELECT PGTE
013504  016477  022002  011224          MOV     PGBTAB(4),@PGTE ;LOAD PBB
013512  012777  000004  011212          MOV     #IMSKE,@ACR     ;SELECT IMSK
013520  016477  022002  011210          MOV     BMITAB(4),@IMSK ;INHIBIT LINE
013526  012777  000001  011176          MOV     #PGCSE,@ACR     ;SELECT PGCS
013534  012777  000001  011174          MOV     #GO,@PGCS       ;TRANSMIT
013542  012777  000001  011162  1$:     MOV     #PGCSE,@ACR     ;SELECT PGCS
013550  032777  004000  011160          BIT     #PRDY,@PGCS     ;CHECK FOR
013556  001771                          BEQ     1$              ;READY
013560  012777  000005  011144          MOV     #PGFE,@ACR      ;SELECT PGF
013566  017737  011144  001126          MOV     @PGF,$BDDAT     ;READ PGF
013574  012737  007400  001124          MOV     #7400,$GDDAT    ;FETCH EXPECTED FLAGS
013602  046437  022002  001124          BIC     PGBTAB(4),$GDDAT;MINUS EXPECTED FLAG
013610  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
013616  001403                          BEQ     2$              ;SKIP IF OK
013620  104044                          ERROR*  44              ;IMSK BIT DID NOT INHIBIT ITS PGF BIT
013622  004737  023526                  JSR     PC,DMPREG
013626  016437  022002  001124  2$:     MOV     UVITAB(4),$GDDAT;LOAD EXPECTED UI BIT
013634  012777  000010  011070          MOV     #EXCE,@ACR      ;SELECT EXC
013642  017737  011070  001126          MOV     @EXC,$BDDAT     ;READ EXC
013650  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK THE UI BIT
013656  001403                          BEQ     3$      ;SKIP IF SET
013660  104044                          ERROR*  44              ;UI DID NOT SET ON A PIF WITH IMSK=1
013662  004737  023526                  JSR     PC,DMPREG
013666  005037  001124          3$:     CLR     $GDDAT          ;EXPECT STF TO BE 0
013672  012777  000006  011032          MOV     #STFE,@ACR      ;SELECT STF
013700  017737  011032  001126          MOV     @STF,$BDDAT     ;GET AND TEST STF
013706  001403                          BEQ     4$              ;IT SHOULD BE 0
013710  104044                          ERROR*  44
013712  004737  023526                  JSR     PC,DMPREG
013716                          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

                                        ;;*******************************************************************
 (3)                                    ;*TEST 42       CHECK THE IMSK BIT FOR PROPER OPERATION ON ST INTR
 (3)                                    ;;*******************************************************************
 (2) 013716 000004                      TST42:  SCOPE
1424 013720 004737 023476               JSR     PC,SSTMNT       ;ENABLE MAINT. LOOP
1425 013724 012777 000000 011000        MOV     #PGTEE,@ACR     ;SELECT PGTE
1426 013732 016477 021772 010776        MOV     PGITAB(4),@PGTE ;LOAD PIB
1427 013740 012777 000004 010764        MOV     #IMSKE,@ACR     ;SELECT IMSK
1428 013746 016477 021772 010762        MOV     IMITAB(4),@IMSK;INHIBIT LINE
1429 013754 012777 000001 010750        MOV     #PGCSE,@ACR     ;SELECT PGCS
1430 013762 012777 000003 010746        MOV     #PTP+GO,@PGCS   ;TRANSMIT
1431 013770 012777 000001 010734 1$:    MOV     #PGCSE,@ACR     ;SELECT PGCS
1432 013776 032777 004000 010732        BIT     #PRDY,@PGCS     ;CHECK FOR
1433 014004 001771                      BEQ     1$              ;READY
1434 014006 012777 000006 010716        MOV     #STFE,@ACR      ;SELECT STF
1435 014014 017737 010716 001126        MOV     @STF,$BDDAT     ;READ STF
1436 014022 012737 000017 001124        MOV     #17,$GDDAT      ;FETCH EXPECTED FLAGS
1437 014030 046437 021772 001124        BIC     PGITAB(4),$GDDAT;MINUS EXPECTED FLAG
1438 014036 023737 001124 001126        CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
1439 014044 001403                      BEQ     2$              ;SKIP IF OK
1440 014046 104044                      ERROR+  44              ;IMSK BIT DID NOT INHIBIT ITS SIF BIT
1441 014050 004737 023526               JSR     PC,DMPREG
1442 014054 016437 022002 001124 2$:    MOV     UVITAB(4),$GDDAT;LOAD EXPECTED UI BIT
1443 014062 012777 000010 010642        MOV     #EXCE,@ACR      ;SELECT EXC
1444 014070 017737 010642 001126        MOV     @EXC,$BDDAT     ;READ EXC
1445 014076 023737 001124 001126        CMP     $GDDAT,$BDDAT   ;CHECK THE UI BIT
1446 014104 001403                      BEQ     3$              ;SKIP IF SET
1447 014106 104044                      ERROR+  44              ;UI DID NOT SET ON A SIF WITH IMSK=1
1448 014110 004737 023526               JSR     PC,DMPREG
1449 014114 005037 001124       3$:     CLR     $GDDAT          ;EXPECT 0
1450 014120 012777 000010 010604        MOV     #EXCE,@ACR      ;SELECT EXC
1451 014126 016477 022002 010602        MOV     UVITAB(4),@EXC  ;CLEAR UI BIT
1452 014134 012777 000010 010570        MOV     #EXCE,@ACR      ;SELECT EXC
1453 014142 017737 010570 024750        MOV     @EXC,DISPLY     ;READ EXC
1454 014150 013737 024750 001126        MOV     DISPLY,$BDDAT   ;TEST EXC
1455 014156 001403                      BEQ     4$              ;SKIP IF 0
1456 014160 104044                      ERROR+  44              ;WRITING A 1 TO UI BIT DID NOT CLEAR IT
1457 014162 004737 023526               JSR     PC,DMPREG
1458 014166 012777 000005 010536 4$:    MOV     #PGFE,@ACR      ;SELECT PGF REGIPGER
1459 014174 017737 010536 001126        MOV     @PGF,$BDDAT     ;GET AND TEST STF REGISTER
1460 014202 001403                      BEQ     5$              ;PGF SHOULD BE 0
1461 014204 104044                      ERROR+  44              ;PGF NOT ZERO ON A STI WITH IMSK=1
1462 014206 004737 023526               JSR     PC,DMPREG
1463 014212                     5$:
```

```
                                ;ISSUE A SANITY TIMER BOOT REQUEST WITH BIMSK BIT =1 AND CHECK:
                                ;SBF BIT TO STAY 0
                                ;UI BIT TO SET

                                ;;*****************************************************************
                                ;*TEST 43       CHECK THE IMSK BIT FOR PROPER OPERATION ON ST BOOT
                                ;;*****************************************************************
 (2) 014212  000004             TST43:  SCOPE
1470 014214  004737  023476             JSR     PC,SSTMNT       ;ENABLE MAINT. LOOP
1471 014220  012777  000000  010504     MOV     #PGTEE,@ACR     ;SELECT PGTE
1472 014226  016477  022002  010502     MOV     PGBTAB(4),@PGTE ;LOAD PBB
1473 014234  012777  000004  010470     MOV     #IMSKE,@ACR     ;SELECT IMSK
1474 014242  016477  022002  010466     MOV     BMITAB(4),@IMSK ;INHIBIT LINE
1475 014250  012777  000001  010454     MOV     #PGCSE,@ACR     ;SELECT PGCS
1476 014256  012777  000003  010452     MOV     #PTP+GO,@PGCS   ;TRANSMIT
1477 014264  012777  000001  010440  1$: MOV     #PGCSE,@ACR     ;SELECT PGCS
1478 014272  032777  004000  010436     BIT     #PRDY,@PGCS     ;CHECK FOR
1479 014300  001771                     BEQ     1$              ;READY
1480 014302  012777  000006  010422     MOV     #STFE,@ACR      ;SELECT STF
1481 014310  017737  010422  001126     MOV     @STF,$BDDAT     ;READ STF
1482 014316  012737  007400  001124     MOV     #7400,$GDDAT    ;FETCH EXPECTED FLAGS
1483 014324  046437  022002  001124     BIC     PGBTAB(4),$GDDAT;MINUS EXPECTED FLAG
1484 014332  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;CHECK FLAG FOR 0
1485 014340  001403                     BEQ     2$              ;SKIP IF OK
1486 014342  104044                     ERROR+  44              ;IMSK BIT DID NOT INHIBIT ITS SBF BIT
1487 014344  004737  023526             JSR     PC,DMPREG
1488 014350  016437  022002  001124  2$: MOV     UVITAB(4),$GDDAT;LOAD EXPECTED UI BIT
1489 014356  012777  000010  010346     MOV     #EXCE,@ACR      ;SELECT EXC
1490 014364  017737  010346  001126     MOV     @EXC,$BDDAT     ;READ EXC
1491 014372  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;CHECK THE UI BIT
1492 014400  001403                     BEQ     3$      ;SKIP IF SET
1493 014402  104044                     ERROR+  44              ;UI DID NOT SET ON A SBF WITH IMSK=1
1494 014404  004737  023526             JSR     PC,DMPREG
1495 014410  005037  001124        3$:  CLR     $GDDAT          ;EXPECT PGF TO BE 0
1496 014414  012777  000005  010310     MOV     #PGFE,@ACR      ;SELECT PGF
1497 014422  017737  010310  001126     MOV     @PGF,$BDDAT     ;GET AND TEST PGF
1498 014430  001403                     BEQ     4$              ;IT SHOULD BE 0
1499 014432  104044                     ERROR+  44
1500 014434  004737  023526             JSR     PC,DMPREG
1501 014440                        4$:
```

```
                                ;TEST OPERATION OF THE TIMER LOGIC
                                ;
                                ;;*****************************************************************
                                ;*TEST 44       CHECK THE TIMER INTERRUPT LOGIC AND COUNT REGISTER
                                ;;*****************************************************************
014440  000004                  TST44:  SCOPE
014442  004737  023476                  JSR     PC,SSTMNT       ;SET MAINT. LOOPBACK
014446  005000                          CLR     R0              ;INIT. TIMER COUNTER
014450  012777  000002  010254          MOV     #STTEE,@ACR     ;SELECT STTE
014456  016477  021772  010252          MOV     SIFTAB(4),@STTE ;ENABLE SI LINE
014464  012777  000001  010244          MOV     #ENB,@STCS      ;ENABLE SANITY TIMER
014472  012777  000003  010232  1$:     MOV     #STCSE,@ACR     ;SELECT STCS
014500  017737  010232  024750          MOV     @STCS,DISPLY    ;READ STCS
014506  032737  177410  024750          BIT     #177410,DISPLY  ;CHECK FOR TMO OR COUNT=-1
014514  001007                          BNE     2$              ;SKIP IF ANY BITS SET
014516  005200                          INC     R0              ;COUNT BACKGROUND TIMER
014520  001364                          BNE     1$              ;KEEP COUNTING UNTIL OVERFLOW
014522  104044                          ERROR+  44              ;TIMER DID NOT COUNTDOWN WHEN ENABLED
014524  004737  023526                  JSR     PC,DMPREG
014530  012700  177777                  MOV     #-1,R0          ;SET TIME TO MAX
014534  010037  024752          2$:     MOV     R0,TIMPER       ;SAVE TIME PERIOD CONSTANT
014540  006200                          ASR     R0              ;DIVIDE BY
014542  006200                          ASR     R0              ;8
014544  006200                          ASR     R0
014546  060037  024752                  ADD     R0,TIMPER       ;ADD 12%
014552  012777  000003  010152          MOV     #STCSE,@ACR     ;;**
014560  017737  010152  024750          MOV     @STCS,DISPLY    ;;**
014566  005037  001126                  CLR     $BDDAT          ;INIT. DISPLAY
014572  113737  024751  001126          MOVB    DISPLY+1,$BDDAT ;FETCH COUNT
014600  012737  000377  001124          MOV     #377,$GDDAT     ;LOAD EXPECTED
014606  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK FOR TIMER = -1
014614  001403                          BEQ     3$              ;SKIP IF OK
014616  104044                          ERROR+  44              ;TIMER DID NOT COUNT FROM 0 TO -1
014620  004737  023526                  JSR     PC,DMPREG
014624  012737  000011  001124  3$:     MOV     #TMO+ENB,$GDDAT ;LOAD EXPECTED
014632  113737  024750  001126          MOVB    DISPLY,$BDDAT   ;LOAD ACTUAL
014640  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK FOR TMO
014646  001403                          BEQ     4$              ;SKIP IF SET
014650  104044                          ERROR+  44              ;TMO DID NOT SET ON TIMER UNDERFLOW
014652  004737  023526                  JSR     PC,DMPREG
014656  012777  000001  010046  4$:     MOV     #PGCSE,@ACR     ;SELECT PGCS
014664  017737  010046  024750          MOV     @PGCS,DISPLY    ;READ PGCS
014672  032737  000010  024750          BIT     #IP,DISPLY      ;TEST FOR IP
014700  001003                          BNE     5$              ;SKIP IF SET
014702  104044                          ERROR+  44              ;SIF DID NOT SET IP
014704  004737  023526                  JSR     PC,DMPREG
014710                          5$:
```

```
1548                                    ;;*****************************************************************
 (3)                                    ;*TEST 45       CHECK THE TIMER BOOT LOGIC AND COUNT REGISTER
 (3)                                    ;;*****************************************************************
 (2)  014710  000004                    TST45:  SCOPE
1549  014712  004737  023476                    JSR     PC,SSTMNT       ;SET MAINT. LOOPBACK
1550  014716  012777  000002  010006            MOV     #STTEE,@ACR     ;SELECT STTE
1551  014724  016477  022002  010004            MOV     SBFTAB(4),@STTE ;ENABLE SB LINE
1552  014732  012777  000001  007776            MOV     #ENB,@STCS      ;ENABLE SANITY TIMER
1553  014740  005000                            CLR     R0              ;CLR COUNTER
1554  014742  012777  000003  007762  1$:       MOV     #STCSE,@ACR     ;SELECT STCS
1555  014750  017737  007762  024750            MOV     @STCS,DISPLY    ;READ STCS
1556  014756  032737  177410  024750            BIT     #177410,DISPLY  ;CHECK FOR TMO OR COUNT=-1
1557  014764  001005                            BNE     2$              ;SKIP IF ANY BITS SET
1558  014766  005200                            INC     R0              ;COUNT BACKGROUND TIMER
1559  014770  001364                            BNE     1$              ;KEEP COUNTING UNTIL OVERFLOW
1560  014772  104044                            ERROR+  44              ;TIMER DID NOT COUNTDOWN WHEN ENABLED
1561  014774  004737  023526                    JSR     PC,DMPREG
1562  015000  012777  000003  007724  2$:       MOV     #STCSE,@ACR     ;;**
1563  015006  017737  007724  024750            MOV     @STCS,DISPLY    ;;**
1564  015014  005037  001126                    CLR     $BDDAT          ;INIT. DISPLAY
1565  015020  113737  024751  001126            MOVB    DISPLY+1,$BDDAT ;FETCH COUNT
1566  015026  012737  000377  001124            MOV     #377,$GDDAT     ;LOAD EXPECTED
1567  015034  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CHECK FOR TIMER = -1
1568  015042  001403                            BEQ     3$              ;SKIP IF OK
1569  015044  104044                            ERROR+  44              ;TIMER DID NOT COUNT FROM 0 TO -1
1570  015046  004737  023526                    JSR     PC,DMPREG
1571  015052  012737  000011  001124  3$:       MOV     #TMO+ENB,$GDDAT ;LOAD EXPECTED
1572  015060  113737  024750  001126            MOVB    DISPLY,$BDDAT   ;LOAD ACTUAL
1573  015066  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CHECK FOR TMO
1574  015074  001403                            BEQ     4$              ;SKIP IF SET
1575  015076  104044                            ERROR+  44              ;TMO DID NOT SET ON TIMER UNDERFLOW
1576  015100  004737  023526                    JSR     PC,DMPREG
1577  015104  012777  000001  007620  4$:       MOV     #PGCSE,@ACR     ;SELECT PGCS
1578  015112  017737  007620  024750            MOV     @PGCS,DISPLY    ;READ PGCS
1579  015120  032737  000010  024750            BIT     #IP,DISPLY      ;TEST FOR IP
1580  015126  001403                            BEQ     5$              ;SKIP IF CLR
1581  015130  104044                            ERROR+  44              ;SBF SET IP
1582  015132  004737  023526                    JSR     PC,DMPREG
1583  015136                          5$:
```

```
 1585                                   ;;*********************************************************************
  (3)                                   ;*TEST 46       CHECK THE COUNTER FOR PROPER DECREMENTATION
  (3)                                   ;;*********************************************************************
  (2)  015136  000004                   TST46:  SCOPE
 1586  015140  012737  000001  001204           MOV     #1,$TMP3        ;INIT THE COUNT VALUE
 1587  015146  005037  001126                   CLR     $BDDAT          ;ZERO THE HIGH BYTE OF $BDDAT
 1588  015152  012737  015160  001110           MOV     #1$,$LPERR      ;SET SCOPE POINT
 1589  015160  004737  023476           1$:     JSR     PC,SSTMNT       ;ENTER MAINT. LOOP MODE
 1590  015164  012777  000002  007540           MOV     #STTEE,@ACR     ;SELECT STTE
 1591  015172  016477  021772  007536           MOV     SIFTAB(4),@STTE ;SET LINE ENABLE
 1592  015200  112777  000001  007530           MOVB    #ENB,@STCS      ;ENABLE COUNTING
 1593  015206  005377  007520                   DEC     @ACR            ;SELECT STCS
 1594  015212  113777  001204  007520           MOVB    $TMP3,@STCSH    ;LOAD COUNT AND START
 1595  015220  013737  001204  001124           MOV     $TMP3,$GDDAT    ;GET STARTING COUNT VALUE
 1596  015226  005337  001124                   DEC     $GDDAT          ;MINUS 1
 1597  015232  013700  024752                   MOV     TIMPER,R0       ;GET BASIC TIME CONSTANT
 1598  015236  006300                           ASL     R0              ;*2
 1599  015240  005377  007466           2$:     DEC     @ACR            ;SELECT STCS
 1600  015244  117737  007470  001126           MOVB    @STCSH,$BDDAT   ;READ COUNT VALUE
 1601  015252  123737  001204  001126           CMPB    $TMP3,$BDDAT    ;CHECK FOR CHANGE
 1602  015260  001006                           BNE     3$              ;SKIP IF CHANGED
 1603  015262  005300                           DEC     R0              ;DEFLATE COUNTER
 1604  015264  001365                           BNE     2$              ;CHECK AGAIN
 1605  015266  104044                           ERROR+  44              ;TIMER DID NOT DECREMENT
 1606  015270  004737  023526                   JSR     PC,DMPREG
 1607  015274  000414                           BR      4$              ;SKIP NEXT CHECK
 1608  015276  005377  007430           3$:     DEC     @ACR            ;SELECT STCS
 1609  015302  117737  007432  001126           MOVB    @STCSH,$BDDAT   ;REREAD TIMER
 1610  015310  023737  001124  001126           CMP     $GDDAT,$BDDAT   ;CHECK FOR CORRECT DECREMENTATION
 1611  015316  001403                           BEQ     4$              ;SKIP IF OK
 1612  015320  104044                           ERROR+  44              ;COUNT REGISTER DID NOT DECREMENT CORRECTLY
 1613  015322  004737  023526                   JSR     PC,DMPREG
 1614  015326  105237  001204           4$:     INCB    $TMP3           ;BUMP TEST VALUE
 1615  015332  001312                           BNE     1$              ;DO 1 - 377
```

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

                 000005                 W=5

                                        ;;*****************************************************************
                                        ;*TEST 47       CHECK OPERATION OF PARITY & FRAMING ERRORS, IM(3:0)=0
                                        ;;*****************************************************************
 015334  000004                 TST47:  SCOPE
 015336  004737  023306                 JSR     PC,SETMNT       ;ENABLE MAINTENANCE LOOPBACK
 015342  012737  015366  001110         MOV     #1$,$LPERR      ;FIX START OF LOOP
 015350  012777  000007  007354         MOV     #DCFE,@ACR
 015356  017737  007354  024756         MOV     @DCF,SNAP0      ;SAVE IMAGE OF DCF REGISTER
 015364  005000                         CLR     R0              ;INIT FRAME, TYPE, PGTE DATA SETUP

 015366  010037  001200         1$:     MOV     R0,$TMP1
 015372  004737  023506                 JSR     PC,PARCAL       ;CALCULATE PARITY OF DATA
 015376  113737  001200  001201         MOVB    $TMP1,$TMP1+1
 015404  106037  001201                 RORB    $TMP1+1         ;POSITION DATA
 015410  106037  001201                 RORB    $TMP1+1
 015414  106037  001201                 RORB    $TMP1+1
 015420  106037  001201                 RORB    $TMP1+1
 015424  012777  000000  007300         MOV     #PGTEE,@ACR
 015432  013777  001200  007276         MOV     $TMP1,@PGTE     ;LOAD TRANSMIT ENABLES
 015440  006301                         ASL     R1              ;GET PARITY
 015442  042701  177775         2$:     BIC     #177775,R1
 015446  016437  021762  001124         MOV     SIDTAB(4),$GDDAT        ;GET SELF-ID
 015454  050137  001124                 BIS     R1,$GDDAT
 015460  052701  000001                 BIS     #1,R1
 015464  012777  000001  007240         MOV     #PGCSE,@ACR
 015472  010177  007240                 MOV     R1,@PGCS        ;LOAD PARITY, SET GO
 015476  005001                         CLR     R1
 015500  052737  004010  001124         BIS     #PRDY+IP,$GDDAT
 015506  105377  007220         3$:     DECB    @ACR
 015512  017737  007220  001126         MOV     @PGCS,$BDDAT
 015520  032737  004000  001126         BIT     #PRDY,$BDDAT    ;WAIT FOR READY
 015526  001005                         BNE     4$
 015530  005301                         DEC     R1              ;BUT DON'T WAIT TOO LONG
 015532  001365                         BNE     3$
 015534  104044                         ERROR+  44
 015536  004737  023526                 JSR     PC,DMPREG
 015542  023737  001126  001124 4$:     CMP     $BDDAT,$GDDAT   ;IS PGCS OK?
 015550  001403                         BEQ     5$
 015552  104044                         ERROR+  44
 015554  004737  023526                 JSR     PC,DMPREG
```

```
1665  015560  012737  000017  001124  5$:     MOV     #17,$GDDAT      ;GET EXPECTED RTE FLAGS
1666  015566  012777  000010  007136          MOV     #EXCE,@ACR
1667  015574  017737  007136  001126          MOV     @EXC,$BDDAT     ;GET EXC REGISTER
1668  015602  023737  001124  001126          CMP     $GDDAT,$BDDAT
1669  015610  001403                          BEQ     6$
1670  015612  104044                          ERROR+  44
1671  015614  004737  023526                  JSR     PC,DMPREG
1672  015620  105377  007106          6$:     DECB    @ACR
1673  015624  016437  021772  001124          MOV     PIFTAB(4),$GDDAT
1674  015632  013777  001124  007076          MOV     $GDDAT,@EXC     ;CLEAR FLAG
1675  015640  005137  001124                  COM     $GDDAT          ;MAKE FLAG DATA
1676  015644  042737  177760  001124          BIC     #177760,$GDDAT
1677  015652  105377  007054                  DECB    @ACR
1678  015656  017737  007054  001126          MOV     @EXC,$BDDAT
1679  015664  023737  001126  001124          CMP     $BDDAT,$GDDAT
1680  015672  001403                          BEQ     7$
1681  015674  104044                          ERROR+  44
1682  015676  004737  023526                  JSR     PC,DMPREG
1683  015702  005037  001124          7$:     CLR     $GDDAT
1684  015706  012777  000005  007016          MOV     #PGFE,@ACR
1685  015714  017737  007016  001126          MOV     @PGF,$BDDAT     ;PGF SHOULD BE 0
1686  015722  001403                          BEQ     8$
1687  015724  104044                          ERROR+  44
1688  015726  004737  023526                  JSR     PC,DMPREG
1689  015732  017737  007000  001126  8$:     MOV     @STF,$BDDAT     ;GET STF
1690  015740  001403                          BEQ     9$              ;STF SHOULD BE 0
1691  015742  104044                          ERROR+  44
1692  015744  004737  023526                  JSR     PC,DMPREG
1693  015750  013737  024756  001124  9$:     MOV     SNAPO,$GDDAT    ;GET EXPECTED DATA FOR DCF
1694  015756  017737  006754  001126          MOV     @DCF,$BDDAT     ;GET DCF REGISTER
1695  015764  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;CHECK THAT THERE IS NO CHANGE
1696  015772  001403                          BEQ     10$
1697  015774  104044                          ERROR+  44
1698  015776  004737  023526                  JSR     PC,DMPREG
1699  016002  005200                  10$:    INC     R0              ;INCREMENT TEST DATA
1700  016004  012777  000015  006720          MOV     #MTCE,@ACR      ;POINT TO MTC
1701  016012  042777  006000  006716          BIC     #MTYP+MFRM,@MTC ;CLEAR TYPE AND FRAME ERROR
1702  016020  032700  000400                  BIT     #400,R0
1703  016024  001403                          BEQ     11$             ;IF R0(8)=1, SET MTYPE
1704  016026  052777  004000  006702          BIS     #MTYP,@MTC
1705  016034  032700  001000          11$:    BIT     #1000,R0
1706  016040  001403                          BEQ     12$
1707  016042  052777  002000  006666          BIS     #MFRM,@MTC      ;IF R0(9)=1, SET FRAMING ERROR
1708  016050  020027  002000          12$:    CMP     R0,#2000
1709  016054  103002                          BHIS    13$             ;LOOP UNTIL ALL COMBOS OF DATA, TYPE &
1710                                                                  ;FRAMING HAVE BEEN TESTED.
1711  016056  000137  015366                  JMP     1$
1712  016062                          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)

                                ;;*********************************************************************
 (3)                            ;*TEST 50       CHECK OPERATION OF PARITY & FRAMING ERRORS, IM(3:0)=1
 (3)                            ;;*********************************************************************
 (2) 016062  000004             1ST50:  SCOPE
1722 016064  004737  023306             JSR     PC,SETMNT       ;ENABLE MAINTENANCE LOOPBACK
1723 016070  012737  016130  001110     MOV     #1$,$LPERR      ;FIX START OF LOOP
1724 016076  012777  000007  006626     MOV     #DCFE,@ACR
1725 016104  017737  006626  024756     MOV     @DCF,SNAP0      ;SAVE IMAGE OF DCF REGISTER
1726 016112  012777  000004  006612     MOV     #IMSKE,@ACR     ;POINT TO IMASK
1727 016120  012777  000017  006610     MOV     #17,@IMSK       ;SET ALL INTERRUPT MASKS
1728 016126  005000                     CLR     R0              ;INIT FRAME, TYPE, PGTE DATA SETUP
1729
1730 016130  010037  001200     1$:     MOV     R0,$TMP1
1731 016134  004737  023506             JSR     PC,PARCAL       ;CALCULATE PARITY OF DATA
1732 016140  113737  001200  001201     MOVB    $TMP1,$TMP1+1
1733 016146  106037  001201             RORB    $TMP1+1         ;POSITION DATA
1734 016152  106037  001201             RORB    $TMP1+1
1735 016156  106037  001201             RORB    $TMP1+1
1736 016162  106037  001201             RORB    $TMP1+1
1737 016166  012777  000000  006536     MOV     #PGTEE,@ACR
1738 016174  013777  001200  006534     MOV     $TMP1,@PGTE     ;LOAD TRANSMIT ENABLES
1739 016202  006301                     ASL     R1              ;GET PARITY
1740 016204  042701  177775     2$:     BIC     #177775,R1
1741 016210  016437  021762  001124     MOV     SIDTAB(4),$GDDAT        ;GET SELF-ID
1742 016216  050137  001124             BIS     R1,$GDDAT
1743 016222  052701  000001             BIS     #1,R1
1744 016226  012777  000001  006476     MOV     #PGCSE,@ACR
1745 016234  010177  006476             MOV     R1,@PGCS        ;LOAD PARITY, SET GO
1746 016240  005001                     CLR     R1
1747 016242  052737  004000  001124     BIS     #PRDY,$GDDAT
1748 016250  105377  006456     3$:     DECB    @ACR
1749 016254  017737  006456  001126     MOV     @PGCS,$BDDAT
1750 016262  032737  004000  001126     BIT     #PRDY,$BDDAT    ;WAIT FOR READY
1751 016270  001005                     BNE     4$
1752 016272  005301                     DEC     R1              ;BUT DON'T WAIT TOO LONG
1753 016274  001365                     BNE     3$
1754 016276  104044                     ERROR+  44
1755 016300  004737  023526             JSR     PC,DMPREG
1756 016304  023737  001126  001124 4$: CMP     $BDDAT,$GDDAT   ;IS PGCS OK?
1757 016312  001403                     BEQ     5$
1758 016314  104044                     ERROR+  44
1759 016316  004737  023526             JSR     PC,DMPREG
1760 016322  012737  000000  001124 5$: MOV     #0,$GDDAT       ;GET EXPECTED RTE FLAGS
1761 016330  012777  000010  006374     MOV     #EXCE,@ACR
1762 016336  017737  006374  001126     MOV     @EXC,$BDDAT     ;GET EXC REGISTER
1763 016344  001403                     BEQ     6$
1764 016346  104044                     ERROR+  44
1765 016350  004737  023526             JSR     PC,DMPREG
1766 016354                         6$:
```

```
1767  016354  005037  001124          7$:     CLR     $GDDAT
1768  016360  012777  000005  006344          MOV     #PGFE,@ACR
1769  016366  017737  006344  001126          MOV     @PGF,$BDDAT     ;PGF SHOULD BE 0
1770  016374  001403                          BEQ     8$
1771  016376  104044                          ERROR+  44
1772  016400  004737  023526                  JSR     PC,DMPREG
1773  016404  017737  006326  001126  8$:     MOV     @STF,$BDDAT     ;GET STF
1774  016412  001403                          BEQ     9$              ;STF SHOULD BE 0
1775  016414  104044                          ERROR+  44
1776  016416  004737  023526                  JSR     PC,DMPREG
1777  016422  013737  024756  001124  9$:     MOV     SNAP0,$GDDAT    ;GET EXPECTED DATA FOR DCF
1778  016430  017737  006302  001126          MOV     @DCF,$BDDAT     ;GET DCF REGISTER
1779  016436  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;CHECK THAT THERE IS NO CHANGE
1780  016444  001403                          BEQ     10$
1781  016446  104044                          ERROR+  44
1782  016450  004737  023526                  JSR     PC,DMPREG
1783  016454  005200                  10$:    INC     R0              ;INCREMENT TEST DATA
1784  016456  012777  000015  006246          MOV     #MTCE,@ACR      ;POINT TO MTC
1785  016464  042777  006000  006244          BIC     #MTYP+MFRM,@MTC ;CLEAR TYPE AND FRAME ERROR
1786  016472  032700  000400                  BIT     #400,R0
1787  016476  001403                          BEQ     11$             ;IF R0(8)=1, SET MTYPE
1788  016500  052777  004000  006230          BIS     #MTYP,@MTC
1789  016506  032700  001000          11$:    BIT     #1000,R0
1790  016512  001403                          BEQ     12$
1791  016514  052777  002000  006214          BIS     #MFRM,@MTC      ;IF R0(9)=1, SET FRAMING ERROR
1792  016522  020027  002000          12$:    CMP     R0,#2000
1793  016526  103600                          BLO     1$              ;LOOP UNTIL ALL COMBOS OF DATA, TYPE &
1794                                                                  ;FRAMING HAVE BEEN TESTED.
```

```
                                ;TEST ALL THE PGCS ERROR BITS FOR PROPER OPERATION
                                ;;*****************************************************************
                                ;*TEST 51        TEST GRJ FOR PROPER OPERATION
                                ;;*****************************************************************
(2)  016530  000004             TST51:  SCOPE
1799 016532  004737  023306             JSR     PC,SETMNT       ;CLR IIST AND ENTER MAINT. MODE
1800 016536  013701  024732             MOV     ACR,R1          ;R1=ACR
1801 016542  013702  024736             MOV     ADR,R2          ;R2=ADR
1802 016546  012711  000000             MOV     #PGTEE,@R1      ;SELECT PGTE
1803 016552  016412  021772             MOV     PGITAB(4),@R2   ;LOAD PGTE WITH ENABLE
1804 016556  012712  000001             MOV     #GO,@R2         ;TRANSMIT
1805 016562  005311                     DEC     @R1             ;SELECT PGCS
1806 016564  005212                     INC     @R2             ;SET GO AGAIN
1807 016566  016437  021762  001124     MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED
1808 016574  062737  144010  001124     ADD     #ERR+GRJ+PRDY+IP,$GDDAT
1809 016602  012711  000001         1$: MOV     #PGCSE,@R1      ;SELECT PGCS
1810 016606  011237  001126             MOV     @R2,$BDDAT      ;READ PGCS
1811 016612  032737  004000  001126     BIT     #PRDY,$BDDAT    ;TEST FOR RDY
1812 016620  001770                     BEQ     1$              ;LOOP UNTIL SET
1813 016622  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;CHECK STATUS
1814 016630  001403                     BEQ     2$              ;SKIP IF OK
1815 016632  104044                     ERROR+  44              ;ERR,GRJ, OR IP DID NOT SET ON
1816 016634  004737  023526             JSR     PC,DMPREG
1817                                                            ;A GRJ ERROR
1818 016640  012711  000005         2$: MOV     #PGFE,@R1       ;SELECT PGF
1819 016644  012712  000017             MOV     #17,@R2         ;CLEAR PGF
1820 016650  017746  006070             MOV     @11IP,-(SP)     ;SAVE OLD
1821 016654  017746  006062             MOV     @11IV,-(SP)     ;PI DATA
1822 016660  012777  016730  006054     MOV     #3$,@11IV       ;LOAD NEW
1823 016666  012777  000340  006050     MOV     #340,@11IP      ;PI DATA
1824 016674  012711  000001             MOV     #PGCSE,@R1      ;SELECT PGCS
1825 016700  012712  000004             MOV     #1E,@R2         ;ENABLE PI
1826 016704  005037  177776             CLR     PS              ;ALLOW PI
1827 016710  000240                     NOP                     ;STALL
1828 016712  012737  000340  177776     MOV     #340,PS         ;LOCKOUT PI
1829 016720  104044                     ERROR+  44              ;NO PI ON ERR WITH IE=1
1830 016722  004737  023526             JSR     PC,DMPREG
1831 016726  000401                     BR      4$              ;SKIP
1832 016730  022626                 3$: POPPOP                  ;FIX STACK
1833 016732  012677  006004         4$: MOV     (SP)+,@11IV     ;RESTORE OLD
1834 016736  012677  006002             MOV     (SP)+,@11IP     ;PI INFO
1835 016742  005311                     DEC     @R1             ;SELECT PGCS
1836 016744  012712  100000             MOV     #ERR,@R2        ;CLEAR ERR BY WRITING 1 TO IT
1837 016750  042737  140010  001124     BIC     #ERR+GRJ+IP,$GDDAT
1838 016756  005311                     DEC     @R1             ;SELECT PGCS
1839 016760  011237  001126             MOV     @R2,$BDDAT      ;READ PGCS
1840 016764  023737  001124  001126     CMP     $GDDAT,$BDDAT   ;CHECK STATUS
1841 016772  001403                     BEQ     5$              ;SKIP IF OK
1842 016774  104044                     ERROR+  44              ;WRITING A ONE TO ERR DID NOT
1843 016776  004737  023526             JSR     PC,DMPREG
1844                                                            ;CLEAR ERR,GRJ, AND IP
1845 017002                         5$:
```

```
                        ;;*****************************************************************
                        ;*TEST 52       TEST PG RMR FOR PROPER OPERATION
                        ;;*****************************************************************
017002  000004          TST52:  SCOPE
017004  004737  023306          JSR     PC,SETMNT       ;CLR IIST AND ENTER MAINT. MODE
017010  013701  024732          MOV     ACR,R1          ;R1=ACR
017014  013702  024736          MOV     ADR,R2          ;R2=ADR
017020  012700  000001          MOV     #GO,R0          ;R0=GO
017024  012703  000000          MOV     #PGTEE,R3       ;R3=PGTE SELECT CODE
017030  010311                  MOV     R3,@R1          ;SELECT PGTE
017032  016412  021772          MOV     PGITAB(4),@R2   ;LOAD PGTE WITH ENABLE
017036  012712  000001          MOV     #GO,@R2         ;TRANSMIT
017042  010311                  MOV     R3,@R1          ;SELECT PGTE
017044  010112                  MOV     R1,@R2          ;ACCESS PGTE AGAIN
017046  016437  021762  001124  MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED
017054  062737  124010  001124  ADD     #ERR+PGMR+PRDY+IP,$GDDAT
017062  012711  000001  1$:     MOV     #PGCSE,@R1      ;SELECT PGCS
017066  011237  001126          MOV     @R2,$BDDAT      ;READ PGCS
017072  032737  004000  001126  BIT     #PRDY,$BDDAT    ;TEST FOR RDY
017100  001770                  BEQ     1$              ;LOOP UNTIL SET
017102  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;CHECK STATUS
017110  001403                  BEQ     2$              ;SKIP IF OK
017112  104044                  ERROR+  44              ;ERR,PGMR, OR IP DID NOT SET ON
017114  004737  023526          JSR     PC,DMPREG
                                                        ;A PGMR ERROR
017120  012711  000005  2$:     MOV     #PGFE,@R1       ;SELECT PGF
017124  012712  000017          MOV     #17,@R2         ;CLEAR PGF
017130  012711  000001          MOV     #PGCSE,@R1      ;SELECT PGCS
017134  012712  100000          MOV     #ERR,@R2        ;CLEAR ERR BY WRITING 1 TO IT
017140  042737  120010  001124  BIC     #ERR+PGMR+IP,$GDDAT
017146  012711  000001          MOV     #PGCSE,@R1      ;SELECT PGCS
017152  011237  001126          MOV     @R2,$BDDAT      ;READ PGCS
017156  023737  001124  001126  CMP     $GDDAT,$BDDAT   ;CHECK STATUS
017164  001403                  BEQ     5$              ;SKIP IF OK
017166  104044                  ERROR+  44              ;WRITING A ONE TO ERR DID NOT
017170  004737  023526          JSR     PC,DMPREG
                                                        ;CLEAR ERR,PGMR, AND IP
017174                  5$:
```

```
                                ;;*****************************************************************
                                ;*TEST 53       TEST ST RMR FOR PROPER OPERATION
                                ;;*****************************************************************
017174  000004                  TST53:  SCOPE
017176  004737  023476                  JSR     PC,SSTMNT       ;CLR IIST AND ENTER MAINT. MODE
017202  013701  024732                  MOV     ACR,R1          ;R1=ACR
017206  013702  024736                  MOV     ADR,R2          ;R2=ADR
017212  012711  000002                  MOV     #STTEE,@R1      ;SELECT STTE
017216  016412  021772                  MOV     STITAB(4),@R2   ;LOAD STTE WITH ENABLE
017222  012712  000001                  MOV     #ENB,@R2        ;START TIMER
017226  016437  021762  001124          MOV     SIDTAB(4),$GDDAT;LOAD EXPECTED
017234  062737  114010  001124          ADD     #ERR+STMR+PRDY+IP,$GDDAT
017242  012711  000003          1$:     MOV     #STCSE,@R1      ;SELECT STCS
017246  032712  000010                  BIT     #TMO,@R2        ;TEST FOR TMO
017252  001773                          BEQ     1$              ;LOOP UNTIL SET
017254  012711  000002                  MOV     #STTEE,@R1      ;SELECT STTE
017260  005212                          INC     @R2             ;ACCESS STTE AGAIN
017262  012711  000001                  MOV     #PGCSE,@R1      ;SELECT PGCS
017266  011237  001126                  MOV     @R2,$BDDAT      ;READ PGCS
017272  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK STATUS
017300  001403                          BEQ     2$              ;SKIP IF OK
017302  104044                          ERROR+  44              ;ERR,STMR, OR IP DID NOT SET ON
017304  004737  023526                  JSR     PC,DMPREG
                                                                ;A STMR ERROR
017310  012746  100000          2$:     MOV     #100000,-(SP)   ;PUSH VALUE
017314  005216                  4$:     INC     (SP)            ;STALL
017316  001376                          BNE     4$
017320  005026                          CLR     (SP)+
017322  012711  000006                  MOV     #STFE,@R1       ;SELECT STF
017326  012712  000017                  MOV     #17,@R2         ;CLEAR STF
017332  012711  000001                  MOV     #PGCSE,@R1      ;SELECT PGCS
017336  012712  100000                  MOV     #ERR,@R2        ;CLEAR ERR BY WRITING 1 TO IT
017342  042737  110010  001124          BIC     #ERR+STMR+IP,$GDDAT
017350  012711  000001                  MOV     #PGCSE,@R1      ;SELECT PGCS
017354  011237  001126                  MOV     @R2,$BDDAT      ;READ PGCS
017360  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK STATUS
017366  001403                          BEQ     5$              ;SKIP IF OK
017370  104044                          ERROR+  44              ;WRITING A ONE TO ERR DID NOT
017372  004737  023526                  JSR     PC,DMPREG
                                                                ;CLEAR ERR,STMR, AND IP
017376                          5$:
017376  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)

017400                                  PGONL:
                                        ;;*****************************************************************
 (3)                                    ;*TEST 54       CHECK OPERATION OF PG REQUESTS IN NORMAL MODE
 (3)                                    ;;*****************************************************************
 (2) 017400  000004                     TST54:  SCOPE
1931 017402  012777  100000  005322             MOV     #CLR,@ACR       ;CLEAR THE IIST
1932 017410  005737  025020             TST     CONFIG          ;ARE WE PREVENTED FROM ONLINE? (CONFIG=0)
1933 017414  001002                     BNE     .+6             ;SKIP IF NOT
1934 017416  000137  020162             JMP     10$             ;IF YES, GO TO NEXT TEST.
1935 017422  004737  023450             JSR     PC,CLEARX       ;CLEAR THE IIST AND FLAGS
1936 017426  012737  017452  001110     MOV     #1$,$LPERR      ;FIX START OF LOOP
1937 017434  012777  000007  005270     MOV     #DCFE,@ACR
1938 017442  017737  005270  024756     MOV     @DCF,SNAP0      ;SAVE IMAGE OF DCF REGISTER
1939 017450  005000                     CLR     R0              ;INIT PGTE DATA SETUP
1940 017452  004737  023450     1$:     JSR     PC,CLEARX       ;CLEAR IIST AND FLAGS
1941 017456  012777  000015  005246     MOV     #MTCE,@ACR      ;SELECT MTC
1942 017464  012777  000010  005244     MOV     #DSBT,@MTC      ;DISABLE BOOTS
1943 017472  010037  001200             MOV     R0,$TMP1
1944 017476  023727  025020  000003     CMP     CONFIG,#3       ;SHOULD WE PREVENT XMIT OF ALL CODES?
1945 017504  001013                     BNE     11$             ;NO IF NOT CONFIG #3.
1946 017506  016401  021772             MOV     PGITAB(4),R1    ;YES -- GET MASK FOR SELF.
1947 017512  010146                     MOV     R1,-(SP)        ;SAVE INTR ENB
1948 017514  006301                     ASL     R1              ;SHIFT 4X TO BOOT ENB
1949 017516  006301                     ASL     R1
1950 017520  006301                     ASL     R1
1951 017522  006301                     ASL     R1
1952 017524  052601                     BIS     (SP)+,R1        ;GET INTR ENB MASK
1953 017526  005101                     COM     R1
1954 017530  040137  001200             BIC     R1,$TMP1        ;CLEAR BITS FOR DESTINATION OTHER THAN SELF.
1955 017534                     11$:
1956 017534  010046                     MOV     R0,-(SP)        ;SAVE COUNT VALUE
1957 017536  013700  001200             MOV     $TMP1,R0        ;GET ENABLE FOR PARITY GENERATOR
1958 017542  004737  023506             JSR     PC,PARCAL       ;CALCULATE PARITY OF DATA
1959 017546  012600                     MOV     (SP)+,R0        ;RESTORE COUNT VALUE.
1960 017550  113737  001200  001201     MOVB    $TMP1,$TMP1+1
1961 017556  106037  001201             RORB    $TMP1+1         ;POSITION DATA
1962 017562  106037  001201             RORB    $TMP1+1
1963 017566  106037  001201             RORB    $TMP1+1
1964 017572  106037  001201             RORB    $TMP1+1
1965 017576  012777  000000  005126     MOV     #PGTEE,@ACR
1966 017604  013777  001200  005124     MOV     $TMP1,@PGTE     ;LOAD TRANSMIT ENABLES
1967 017612  006301                     ASL     R1              ;GET PARITY
1968 017614  005101                     COM     R1              ;FIX PAR BIT
1969 017616  042701  177775     2$:     BIC     #177775,R1
1970 017622  016437  021762  001124     MOV     SIDTAB(4),$GDDAT        ;GET SELF-ID
1971 017630  050137  001124             BIS     R1,$GDDAT
1972 017634  052701  000001             BIS     #1,R1
1973 017640  012777  000001  005064     MOV     #PGCSE,@ACR
1974 017646  010177  005064             MOV     R1,@PGCS        ;LOAD PARITY, SET GO
1975 017652  005001                     CLR     R1
```

```
1976  017654  052737  004000  001124          BIS     #PRDY,$GDDAT
1977  017662  036437  021772  001200          BIT     PG!TAB(4),$TMP1 ;TEST FOR PGI ENABLE
1978  017670  001403                          BEQ     3$              ;SKIP IF CLEAR
1979  017672  052737  000010  001124          BIS     #IP,$GDDAT      ;SET IP IF RELEVANT
1980  017700  105377  005026          3$:     DECB    @ACR
1981  017704  017737  005026  001126          MOV     @PGCS,$BDDAT
1982  017712  032737  004000  001126          BIT     #PRDY,$BDDAT    ;WAIT FOR READY
1983  017720  001005                          BNE     4$
1984  017722  005301                          DEC     R1              ;BUT DON'T WAIT TOO LONG
1985  017724  001365                          BNE     3$
1986  017726  104044                          ERROR+  44              ;PRDY DID NOT SET ON A PG XFER
1987  017730  004737  023526                  JSR     PC,DMPREG
1988  017734  023737  001126  001124  4$:     CMP     $BDDAT,$GDDAT   ;IS PGCS OK?
1989  017742  001403                          BEQ     5$
1990  017744  104044                          ERROR+  44              ;PGCS STATUS ERROR ON PG XFER
1991  017746  004737  023526                  JSR     PC,DMPREG
1992  017752  005037  001124          5$:     CLR     $GDDAT          ;SET EXPECTED RTE FLAGS
1993  017756  012777  000010  004746          MOV     #EXCE,@ACR
1994  017764  017737  004746  001126          MOV     @EXC,$BDDAT     ;GET EXC REGISTER
1995  017772  023737  001124  001126          CMP     $GDDAT,$BDDAT
1996  020000  001403                          BEQ     6$
1997  020002  104044                          ERROR+  44              ;EXC NOT 0 ON A PG XFER
1998  020004  004737  023526                  JSR     PC,DMPREG
1999  020010  036437  021772  001200  6$:     BIT     PGITAB(4),$TMP1 ;CHECK FOR LEGAL PGF
2000  020016  001403                          BEQ     61$             ;SKIP IF NO
2001  020020  056437  021772  001124          BIS     PGITAB(4),$GDDAT;LOAD PGF
2002  020026  036437  022002  001200  61$:    BIT     PGBTAB(4),$TMP1 ;CHECK FOR LEGAL PGB
2003  020034  001403                          BEQ     62$             ;SKIP IF NO
2004  020036  056437  022002  001124          BIS     PGBTAB(4),$GDDAT;LOAD PGB
2005  020044  012777  000005  004660  62$:    MOV     #PGFE,@ACR
2006  020052  017737  004660  001126          MOV     @PGF,$BDDAT     ;READ PGF
2007  020060  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK PGF
2008  020066  001403                          BEQ     7$
2009  020070  104044                          ERROR+  44              ;PGF ERROR ON PG XFER
2010  020072  004737  023526                  JSR     PC,DMPREG
2011  020076  005037  001124          7$:     CLR     $GDDAT          ;EXPECTED = 0
2012  020102  017737  004630  001126          MOV     @STF,$BDDAT     ;GET STF
2013  020110  001403                          BEQ     8$              ;STF SHOULD BE 0
2014  020112  104044                          ERROR+  44              ;STF NO 0 ON A PG XFER
2015  020114  004737  023526                  JSR     PC,DMPREG
2016  020120  013737  024756  001124  8$:     MOV     SNAP0,$GDDAT    ;GET EXPECTED DATA FOR DCF
2017  020126  017737  004604  001126          MOV     @DCF,$BDDAT     ;GET DCF REGISTER
2018  020134  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;CHECK THAT THERE IS NO CHANGE
2019  020142  001403                          BEQ     9$
2020  020144  104044                          ERROR+  44              ;DCF ERROR ON A PG XFER
2021  020146  004737  023526                  JSR     PC,DMPREG
2022  020152  105200                  9$:     INCB    R0              ;INCREMENT TEST DATA
2023  020154  001402                          BEQ     10$             ;SKIP WHEN DONE
2024  020156  000137  017452                  JMP     1$
2025  020162                          10$:
```

```
                                        ;FORCE A BREAK USING THE MAINTENENCE REGISTER AND TEST
                                        ;THE DCF REGISTER
                                        ;
                                        ;;*****************************************************************
                                        ;*TEST 55       TEST THE DCF REGISTER
                                        ;;*****************************************************************
(2)  020162  000004                     TST55:  SCOPE
2031 020164  004737  023450                     JSR     PC,CLEARX       ;INIT THE IIST
2032 020170  012737  000010  001200             MOV     #DSBT,$TMP1     ;SET VALUE FOR MTC TO CLEAR BREAK.
2033 020176  012737  000000  001202             MOV     #0,$TMP2        ;SET VALUE FOR CLEARING UNWANTED BITS.
2034 020204  005737  025020                     TST     CONFIG          ;SEE WHAT CONFIGURATION.
2035 020210  001005                             BNE     11$             ;IF NOT MAINT., SKIP ON.
2036 020212  004737  023306                     JSR     PC,SETMNT       ;IF MAINT. ONLY, SET MAINT. LOOPBACK.
2037 020216  012737  000012  001200             MOV     #MLEN+DSBT,$TMP1        ;NEW VALUE FOR CLEARING BRK.
2038 020224  016401  022002             11$:    MOV     BRKTAB(4),R1    ;GET BRK BIT FOR OWN LINE.
2039 020230  056401  021772                     BIS     DCFTAB(4),R1    ;AND DCF BIT.
2040 020234  005101                             COM     R1              ;MAKE "BIC" MASK.
2041 020236  042701  170360                     BIC     #170360,R1
2042 020242  010137  001202                     MOV     R1,$TMP2        ;STORE THE VALUE.
2043 020246  012777  000007  004456             MOV     #DCFE,@ACR      ;SELECT DCF
2044 020254  017737  004456  024756             MOV     @DCF,SNAP0      ;SAVE THE DCF CONTENTS
2045 020262  013737  024756  001124             MOV     SNAP0,$GDDAT    ;COPY DCF
2046 020270  023727  025020  000001             CMP     CONFIG,#1       ;ANY OTHER INTERFACES?
2047 020276  001003                             BNE     12$
2048 020300  012737  007400  001124             MOV     #7400,$GDDAT    ;SAY EXPECTED HAS OTHER BRK BITS SET.
2049 020306  005737  025020             12$:    TST     CONFIG          ;IN MAINT. MODE?
2050 020312  001002                             BNE     13$
2051 020314  005037  001124                     CLR     $GDDAT          ;YES -- ALL DCF BITS CLEAR.
2052 020320  046437  022002  001124     13$:    BIC     BRKTAB(4),$GDDAT;REMOVE SID BRK
2053 020326  013737  024756  001126             MOV     SNAP0,$BDDAT    ;READ ACTUAL
2054 020334  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK FOR LEGAL DCF
2055 020342  001403                             BEQ     1$              ;SKIP IF OK
2056 020344  104044                             ERROR+  44              ;INCORRECT DCF FOR SELECTED IIST
2057 020346  004737  023526                     JSR     PC,DMPREG
2058 020352  012777  000015  004352     1$:     MOV     #MTCE,@ACR      ;SELECT MTC
2059 020360  012777  000011  004350             MOV     #DSBT+MDSD,@MTC ;FORCE A BREAK
2060 020366  012777  000007  004336             MOV     #DCFE,@ACR      ;SELECT DCF
2061 020374  017737  004336  001126             MOV     @DCF,$BDDAT     ;READ DCF
2062 020402  043737  001202  001126             BIC     $TMP2,$BDDAT    ;CLEAR BITS NOT OUR OWN.
2063 020410  016437  022002  001124             MOV     BRKTAB(4),$GDDAT;SET RELEVANT BRK BIT
2064 020416  056437  021772  001124             BIS     DCFTAB(4),$GDDAT;SET RELEVANT DCF BIT
2065 020424  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK DCF
2066 020432  001403                             BEQ     2$              ;SKIP IF OK
2067 020434  104044                             ERROR+  44              ;INCORRECT DCF AFTER FORCING A BREAK
2068 020436  004737  023526                     JSR     PC,DMPREG
2069 020442  012777  000015  004262     2$:     MOV     #MTCE,@ACR      ;SELECT MTC
2070 020450  013777  001200  004260             MOV     $TMP1,@MTC              ;CLEAR BRK BIT
2071 020456  012777  000007  004246             MOV     #DCFE,@ACR      ;SELECT DCF
2072 020464  017737  004246  001126             MOV     @DCF,$BDDAT     ;READ DCF
2073 020472  043737  001202  001126             BIC     $TMP2,$BDDAT
2074 020500  046437  022002  001124             BIC     BRKTAB(4),$GDDAT;CLEAR OUT BRK BIT
2075 020506  023737  001124  001126             CMP     $GDDAT,$BDDAT   ;CHECK DCF
2076 020514  001403                             BEQ     3$              ;SKIP IF BRK=0
2077 020516  104044                             ERROR+  44              ;INCORRECT DCF AFTER REMOVING THE BRK CONDITION
2078 020520  004737  023526                     JSR     PC,DMPREG
2079 020524  016437  021762  001124     3$:     MOV     SIDTAB(4),$GDDAT;LOAD SELF ID #
```

```
2080  020532  052737  004010  001124          BIS     #IP+PRDY,$GDDAT ;ADD OTHER EXPECTED DATA
2081  020540  012777  000007  004164          MOV     #DCFE,@ACR      ;SELECT DCF
2082  020546  013777  001202  004162          MOV     $TMP2,@DCF      ;CLEAR UNWANTED FLAGS.
2083  020554  012777  000001  004150          MOV     #PGCSE,@ACR     ;SELECT PGCS
2084  020562  017737  004150  001126          MOV     @PGCS,$BDDAT    ;READ PGCS
2085  020570  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK PGCS FOR IP
2086  020576  001403                          BEQ     4$              ;SKIP IF OK
2087  020600  104044                          ERROR+  44              ;DCF FLAG DID NOT SET IP
2088  020602  004737  023526                  JSR     PC,DMPREG
2089  020606  017746  004130          4$:     MOV     @1111V,-(SP)    ;SAVE OLD PI
2090  020612  017746  004126                  MOV     @1111P,-(SP)    ;DATA
2091  020616  012777  020672  004116          MOV     #5$,@1111V      ;LOAD NEW
2092  020624  012777  000340  004112          MOV     #340,@1111P     ;PI DATA
2093  020632  012777  000001  004072          MOV     #PGCSE,@ACR     ;SELECT PGCS
2094  020640  012777  000004  004070          MOV     #1E,@PGCS       ;ENABLE PI
2095  020646  005037  177776                  CLR     PS              ;ALLOW PI
2096  020652  000240                          NOP                     ;STALL
2097  020654  012737  000340  177776          MOV     #340,PS         ;LOCKOUT PI
2098  020662  104044                          ERROR+  44              ;NO PI ON A DCF FLAG
2099  020664  004737  023526                  JSR     PC,DMPREG
2100  020670  000401                          BR      6$              ;SKIP
2101  020672  022626          5$:     POPPOP                          ;FIX STACK
2102  020674  012677  004044          6$:     MOV     (SP)+,@1111P    ;RESTORE OLD
2103  020700  012677  004036                  MOV     (SP)+,@1111V    ;PI DATA
2104  020704  012737  000000  001124          MOV     #0,$GDDAT       ;RELOAD EXPECTED DCF VALUE
2105  020712  012777  000007  004012          MOV     #DCFE,@ACR      ;SELECT DCF
2106  020720  016477  021772  004010          MOV     DCFTAB(4),@DCF  ;CLEAR DCF FLAG
2107  020726  005377  004000                  DEC     @ACR            ;RESELECT DCF
2108  020732  017737  004000  001126          MOV     @DCF,$BDDAT     ;READ DCF
2109  020740  043737  001202  001126          BIC     $TMP2,$BDDAT    ;MASK OUT JUNK.
2110  020746  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK DCF
2111  020754  001403                          BEQ     7$              ;SKIP IF DCF-0
2112  020756  104044                          ERROR+  44              ;WRITING A 1 TO DCF FLAG DID NOT CLEAR IT
2113  020760  004737  023526                  JSR     PC,DMPREG
2114  020764  004737  023450          7$:     JSR     PC,CLEARX       ;INIT. THE INTERFACE.
2115  020770  012777  000004  003734          MOV     #IMSKE,@ACR     ;SELECT THE IMSK REGISTER.
2116  020776  012777  000017  003732          MOV     #17,@IMSK       ;SET ALL INTERRUPT MASKS.
2117  021004  012777  000015  003720          MOV     #MTCE,@ACR      ;SELECT MTC.
2118  021012  012777  000012  003716          MOV     #DSBT+MLEN,@MTC ;DO MAINT. LOOP TO CLEAR BREAKS.
2119  021020  027777  003706  003704          CMP     @ACR,@ACR       ;DELAY
2120  021026  027777  003700  003676          CMP     @ACR,@ACR       ;  "
2121  021034  027777  003672  003670          CMP     @ACR,@ACR
2122  021042  012777  000011  003666          MOV     #DSBT+MDSD,@MTC ;CAUSE BREAK
2123  021050  027777  003656  003654          CMP     @ACR,@ACR
2124  021056  027777  003650  003646          CMP     @ACR,@ACR
2125  021064  012737  000000  001124          MOV     #0,$GDDAT       ;SAY GOOD IS 0
2126  021072  012777  000007  003632          MOV     #DCFE,@ACR      ;SELECT DCF
2127  021100  017737  003632  001126          MOV     @DCF,$BDDAT     ;GET DCF
2128  021106  042737  007400  001126          BIC     #7400,$BDDAT    ;CLEAR BREAKS
2129  021114  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;VERIFY
2130  021122  001403                          BEQ     8$
2131  021124  104044                          ERROR+  44
2132  021126  004737  023526                  JSR     PC,DMPREG
2133  021132  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)

021134                                  STONL:
                                        ;;*****************************************************************
                                        ;*TEST 56       CHECK OPERATION OF ST REQUESTS IN NORMAL MODE
                                        ;;*****************************************************************
021134  000004                          1ST56:  SCOPE
021136  012777  100000  003566                  MOV     #CLR,@ACR       ;CLEAR THE IIST
021144  004737  023450                          JSR     PC,CLEARX
021150  005737  025020                          TST     CONFIG
021154  001001                                  BNE     .+4
021156  000207                                  RTS     PC              ;EXIT IF WE CAN'T XMIT
021160  012737  021204  001110                  MOV     #1$,$LPERR      ;FIX START OF LOOP
021166  012777  000007  003536                  MOV     #DCFE,@ACR
021174  017737  003536  024756                  MOV     @DCF,SNAP0      ;SAVE IMAGE OF DCF REGISTER
021202  005000                                  CLR     R0              ;INIT STTE DATA SETUP
021204  012777  100000  003520  1$:             MOV     #CLR,@ACR       ;CLEAR THE IIST
021212  012777  000005  003512                  MOV     #PGFE,@ACR      ;SELECT PGF
021220  012777  007400  003510                  MOV     #7400,@PGF      ;CLEAR ALL
021226  012777  007400  003502                  MOV     #7400,@STF      ;FLAGS
021234  012777  000017  003474                  MOV     #17,@DCF        ;CLEAR BRK
021242  012777  000015  003462                  MOV     #MTCE,@ACR      ;SELECT MTC
021250  012777  000010  003460                  MOV     #DSBT,@MTC      ;DISABLE BOOTS
021256  010037  001200                          MOV     R0,$TMP1
021262  023727  025020  000003                  CMP     CONFIG,#3       ;SHOULD WE PREVENT XMIT OF ALL CODES?
021270  001013                                  BNE     11$             ;NO IF NOT CONFIG #3.
021272  016401  021772                          MOV     STITAB(4),R1    ;YES -- GET MASK FOR SELF.
021276  010146                                  MOV     R1,-(SP)        ;SAVE INTR ENB
021300  006301                                  ASL     R1              ;SHIFT 4X TO BOOT ENB
021302  006301                                  ASL     R1
021304  006301                                  ASL     R1
021306  006301                                  ASL     R1
021310  052601                                  BIS     (SP)+,R1        ;GET INTR ENB MASK
021312  005101                                  COM     R1
021314  040137  001200                          BIC     R1,$TMP1        ;CLEAR BITS FOR DESTINATION OTHER THAN SELF.
021320                                  11$:
021320  010046                                  MOV     R0,-(SP)        ;SAVE COUNT VALUE
021322  013700  001200                          MOV     $TMP1,R0        ;GET ENABLE FOR PARITY GENERATOR
021326  004737  023506                          JSR     PC,PARCAL       ;CALCULATE PARITY OF DATA
021332  012600                                  MOV     (SP)+,R0        ;RESTORE COUNT VALUE.
021334  113737  001200  001201                  MOVB    $TMP1,$TMP1+1
021342  106037  001201                          RORB    $TMP1+1         ;POSITION DATA
021346  106037  001201                          RORB    $TMP1+1
021352  106037  001201                          RORB    $TMP1+1
021356  106037  001201                          RORB    $TMP1+1
021362  012777  000002  003342                  MOV     #STTEE,@ACR
021370  013777  001200  003340                  MOV     $TMP1,@STTE     ;LOAD TRANSMIT ENABLES
021376  006301                                  ASL     R1              ;GET PARITY
021400  005101                                  COM     R1              ;FIX PAR BIT
021402  042701  177775                  2$:     BIC     #177775,R1
021406  010137  001124                          MOV     R1,$GDDAT       ;LOAD PARITY
021412  052701  000001                          BIS     #1,R1
```

```
2188  021416  012777  000003  003306          MOV     #STCSE,@ACR
2189  021424  010177  003306                  MOV     R1,@STCS        ;LOAD PARITY, ENB, AND COUNT
2190  021430  005001                          CLR     R1
2191  021432  052737  177411  001124          BIS     #TMO+177401,$GDDAT
2192  021440  012777  000004  003264          MOV     #IMSKE,@ACR     ;SELECT IMSK
2193  021446  105377  003260          3$:     DECB    @ACR
2194  021452  017737  003260  001126          MOV     @STCS,$BDDAT
2195  021460  032737  000010  001126          BIT     #TMO,$BDDAT     ;WAIT FOR TMO
2196  021466  001005                          BNE     4$
2197  021470  005301                          DEC     R1              ;BUT DON'T WAIT TOO LONG
2198  021472  001365                          BNE     3$
2199  021474  104044                          ERROR+  44              ;TMO DID NOT SET ON A ST XFER
2200  021476  004737  023526                  JSR     PC,DMPREG
2201  021502  012777  000003  003222  4$:     MOV     #STCSE,@ACR     ;SELECT STCS
2202  021510  017737  003222  001126          MOV     @STCS,$BDDAT    ;READ STCS
2203  021516  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;IS STCS OK?
2204  021524  001403                          BEQ     5$
2205  021526  104044                          ERROR+  44              ;STCS STATUS ERROR ON ST XFER
2206  021530  004737  023526                  JSR     PC,DMPREG
2207  021534  005037  001124          5$:     CLR     $GDDAT          ;SET EXPECTED RTE FLAGS
2208  021540  012777  000010  003164          MOV     #EXCE,@ACR
2209  021546  017737  003164  001126          MOV     @EXC,$BDDAT     ;GET EXC REGISTER
2210  021554  023737  001124  001126          CMP     $GDDAT,$BDDAT
2211  021562  001403                          BEQ     6$
2212  021564  104044                          ERROR+  44              ;EXC NOT 0 ON A ST XFER
2213  021566  004737  023526                  JSR     PC,DMPREG
2214  021572  036437  021772  001200  6$:     BIT     STITAB(4),$TMP1 ;CHECK FOR STF
2215  021600  001403                          BEQ     61$             ;SKIP IF 0
2216  021602  056437  021772  001124          BIS     STITAB(4),$GDDAT;SET STF
2217  021610  036437  022002  001200  61$:    BIT     STBTAB(4),$TMP1 ;CHECK FOR STB
2218  021616  001403                          BEQ     62$             ;SKIP IF 0
2219  021620  056437  022002  001124          BIS     STBTAB(4),$GDDAT;SET STB
2220  021626  012777  000006  003076  62$:    MOV     #STFE,@ACR
2221  021634  017737  003076  001126          MOV     @STF,$BDDAT     ;READ STF
2222  021642  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CHECK DATA
2223  021650  001403                          BEQ     7$
2224  021652  104044                          ERROR+  44              ;STF ERROR ON ST XFER
2225  021654  004737  023526                  JSR     PC,DMPREG
2226  021660  005037  001124          7$:     CLR     $GDDAT          ;EXPECTED = 0
2227  021664  012777  000005  003040          MOV     #PGFE,@ACR      ;SELECT PGF
2228  021672  017737  003040  001126          MOV     @PGF,$BDDAT     ;GET PGF
2229  021700  001403                          BEQ     8$              ;PGF SHOULD BE 0
2230  021702  104044                          ERROR+  44              ;PGF NO 0 ON A ST XFER
2231  021704  004737  023526                  JSR     PC,DMPREG
2232  021710  013737  024756  001124  8$:     MOV     SNAP0,$GDDAT    ;GET EXPECTED DATA FOR DCF
2233  021716  012777  000007  003006          MOV     #DCFE,@ACR      ;SELECT DCF
2234  021724  017737  003006  001126          MOV     @DCF,$BDDAT     ;GET DCF REGISTER
2235  021732  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;CHECK THAT THERE IS NO CHANGE
2236  021740  001403                          BEQ     9$
2237  021742  104044                          ERROR+  44              ;DCF ERROR ON A ST XFER
2238  021744  004737  023526                  JSR     PC,DMPREG
2239  021750  105200                  9$:     INCB    R0              ;INCREMENT TEST DATA
2240  021752  001402                          BEQ     10$             ;SKIP WHEN DONE
2241  021754  000137  021204                  JMP     1$
2242  021760  000207                  10$:    RTS     PC              ;EXIT
```

```
                                .SBTTL  LINE CONTROL BIT TABLES

021762  000000          SIDTAB: SID0                    ;SID TABLE
021764  000400                  SID1
021766  001000                  SID2
021770  001400                  SID3

021772                  PGITAB:                         ;PROGRAM INTERRUPT ENABLE TABLE
021772                  STITAB:                         ;SANITY INTERRUPT ENABLE TABLE
021772                  IMITAB:                         ;INTERRUPT MASK INHIBIT TABLE
021772                  PIFTAB:                         ;PGM INTERRUPT FLAG TABLE
021772                  SIFTAB:                         ;SANITY INTERRUPT FLAG TABLE
021772                  DCFTAB:                         ;DCLO INTERRUPT FLAG TABLE
021772  000001          RTETAB: RTE0                    ;INVALID INPUT FLAG TABLE
021774  000002                  RTE1
021776  000004                  RTE2
022000  000010                  RTE3

022002                  PGBTAB:                         ;PROGRAM BOOT ENABLE TABLE
022002                  STBTAB:                         ;SANITY BOOT ENABLE TABLE
022002                  BMITAB:                         ;BOOT MASK INHIBIT TABLE
022002                  PBFTAB:                         ;PGM BOOT FLAG TABLE
022002                  SBFTAB:                         ;SANITY BOOT FLAG TABLE
022002                  BRKTAB:                         ;BREAK STATE TABLE
022002  000400          UVITAB: UI0                     ;UNEXPECTED INPUT FLAG TABLE
022004  001000                  UI1
022006  002000                  UI2
022010  004000                  UI3
```

```
  2273  022012                          ENDTST:
  2274
  2275                                  ;ON FIRST PASS, DETERMINE SANITY-TIMER COUNT RATE (256 US, 8.192 MS,
  2276                                  ;OR 16.384 MS) AND PRINT IT OUT.
  2277                                  ;ON SUBSEQUENT PASSES, CHECK FOR CONSISTENCY.
  2278                                  ;
  2279                                  ;MAINTENANCE LOOPBACK IS USED, WITH HARDWARE SELF-ID.
  2280
  2281                                  ;;*****************************************************************
   (3)                                  ;*TEST 57       DETERMINE SANITY TIMER COUNT RATE
   (3)                                  ;;*****************************************************************
   (2)  022012  000004                  TST57:  SCOPE
  2282  022014  012737  000340  177776          MOV     #340,PSW        ;RAISE PROCESSOR PRIORITY
  2283  022022  004737  023306                  JSR     PC,SETMNT       ;INIT THE INTERFACE
  2284  022026  012777  000015  002676          MOV     #MTCE,@ACR      ;SELECT MTC REGISTER
  2285  022034  012777  000013  002674          MOV     #DSBT+MLEN+MDSD,@MTC    ;SET LOOPBACK, DISABLE BOOT & DRIVERS
  2286  022042  117704  002666                  MOVB    @ACRH,R4        ;GET HARDWARE SELF-ID
  2287  022046  010437  024754                  MOV     R4,LINMBR       ;PUT IT INTO LINE NUMBER WORD
  2288  022052  006304                          ASL     R4              ;MAKE A WORD INDEX
  2289  022054  012777  000002  002650          MOV     #STTEE,@ACR     ;SELECT STTE (S.T. TRANSMIT ENABLE)
  2290  022062  016477  021772  002646          MOV     STITAB(R4),@STTE        ;SET XMIT ENABLE FOR SELF
  2291  022070  017746  002650                  MOV     @1111P,-(SP)    ;SAVE OLD INTR VECTOR INFO
  2292  022074  017746  002642                  MOV     @1111V,-(SP)
  2293  022100  012777  022234  002634          MOV     #4$,@1111V      ;SETUP NEW INTR VECTOR
  2294  022106  012777  000340  002630          MOV     #340,@1111P
  2295  022114  005000                          CLR     R0              ;CLEAR CHARACTER COUNT
  2296  022116  013701  024732                  MOV     ACR,R1          ;R1 = ACR ADDRESS
  2297  022122  013702  024736                  MOV     ADR,R2          ;R2 = ADR ADDRESS
  2298  022126  012703  000001                  MOV     #PGCSE,R3       ;R3 = PGCS ACCESS CODE
  2299  022132  012704  004000                  MOV     #PRDY,R4        ;R4 = PG RDY BIT MASK
  2300  022136  012705  000007                  MOV     #PTP+IE+GO,R5   ;R5 = VALUE FOR LOADING PGCS
  2301  022142  012711  000003                  MOV     #STCSE,(R1)     ;SELECT STCS
  2302  022146  012712  000401                  MOV     #401,(R2)       ;START SANITY TIMER FOR DOING 2 COUNTS
  2303  022152  005037  177776                  CLR     PSW             ;ALLOW INTERRUPTS
  2304
  2305  022156  010311                  1$:     MOV     R3,(R1)         ;SELECT PGCS
  2306  022160  010512                          MOV     R5,(R2)         ;XMIT NULL PG COMMAND, SET INTR ENABLE
  2307  022162  005200                          INC     R0              ;COUNT # OF CHARS TRANSMITTED
  2308  022164  001404                          BEQ     3$              ;BUT DON'T DO FOREVER.
  2309  022166  010311                  2$:     MOV     R3,(R1)         ;SELECT PGCS
  2310  022170  030412                          BIT     R4,(R2)         ;LOOK FOR PG RDY
  2311  022172  001371                          BNE     1$              ;XMIT AGAIN IF SET
  2312  022174  000774                          BR      2$              ;LOOP IF NOT
  2313
  2314                                  ;COME HERE IF NO TIMEOUT INTR SEEN AFTER 64K TRANSMITS.
  2315  022176  012737  000340  177776  3$:     MOV     #340,PSW        ;RAISE PROCESSOR TO LOCKOUT INTERRUPTS
  2316  022204  012737  177410  001124          MOV     #177400+TMO,$GDDAT ;SETUP EXPECTED DATA
  2317  022212  012711  000003                  MOV     #STCSE,(R1)     ;SELECT STCS
  2318  022216  011237  001126                  MOV     (R2),$BDDAT     ;GET CONTENTS OF STCS FOR PRINTOUT
  2319  022222  104044                          ERROR+  44              ;NO TIMEOUT SEEN AFTER 64K TRANSMITS.
  2320  022224  004737  023526                  JSR     PC,DMPREG
  2321  022230  000137  023260                  JMP     99$             ;THEN JUST EXIT
  2322
  2323                                  ;COME HERE ON INTERRUPT, CHECK THAT IT'S THE CORRECT ONE.
  2324  022234  062706  000004          4$:     ADD     #4,SP           ;POP STACK
  2325  022240  010046                          MOV     R0,-(SP)        ;SAVE COUNT
```

```
2326  022242  010311                          MOV     R3,(R1)          ;SELECT PGCS
2327  022244  012705  000062                  MOV     #50.,R5          ;WAIT FOR FINAL PG RDY
2328  022250  005305                          DEC     R5
2329  022252  001376                          BNE     .-2
2330  022254  011237  001126                  MOV     (R2),$BDDAT      ;GET PGCS CONTENTS
2331  022260  011137  001124                  MOV     (R1),$GDDAT      ;GET SELF-ID
2332  022264  042737  000017  001124          BIC     #17,$GDDAT       ;CLEAR CODE
2333  022272  052737  004016  001124          BIS     #PRDY+IE+IP+PTP,$GDDAT  ;SET UP EXPECTED VALUE
2334  022300  023737  001126  001124          CMP     $BDDAT,$GDDAT    ;VERIFY PGCS CONTENTS
2335  022306  001403                          BEQ     5$
2336  022310  104044                          ERROR+  44               ;PGCS INCORRECT AFTER INTERRUPT
2337  022312  004737  023526                  JSR     PC,DMPREG
2338  022316  005037  001124          5$:     CLR     $GDDAT           ;SETUP EXPECTED VALUE FOR PGF, DCF, AND EXC
2339  022322  012777  000005  002402          MOV     #PGFE,@ACR       ;SELECT PGF
2340  022330  017737  002402  001126          MOV     @PGF,$BDDAT      ;GET PG RECEIVE FLAGS
2341  022336  001403                          BEQ     6$               ;SKIP IF 0
2342  022340  104044                          ERROR+  44               ;PGF NOT 0 AFTER INTERRUPT
2343  022342  004737  023526                  JSR     PC,DMPREG
2344  022346  012777  000007  002356  6$:     MOV     #DCFE,@ACR       ;SELECT DCF
2345  022354  017737  002356  001126          MOV     @DCF,$BDDAT      ;GET FLAGS & TEST THEM
2346  022362  001403                          BEQ     7$               ;  -- SHOULD BE 0
2347  022364  104044                          ERROR+  44               ;DCF NOT 0 AFTER INTERRUPT
2348  022366  004737  023526                  JSR     PC,DMPREG
2349  022372  012777  000010  002332  7$:     MOV     #EXCE,@ACR       ;SELECT EXC
2350  022400  017737  002332  001126          MOV     @EXC,$BDDAT      ;GET FLAGS & TEST THEM
2351  022406  001403                          BEQ     8$               ; -- SHOULD BE 0
2352  022410  104044                          ERROR+  44               ;EXC NOT CLEAR AFTER INTERRUPT
2353  022412  004737  023526                  JSR     PC,DMPREG
2354  022416  012737  000017  001124  8$:     MOV     #17,$GDDAT       ;SETUP EXPECTED VALUE FOR STF
2355  022424  012777  000006  002300          MOV     #STFE,@ACR       ;SELECT STF
2356  022432  017737  002300  001126          MOV     @STF,$BDDAT      ;GET STF
2357  022440  023737  001126  001124          CMP     $BDDAT,$GDDAT    ;CHECK VALUE
2358  022446  001403                          BEQ     9$
2359  022450  104044                          ERROR+  44               ;STF INCORRECT AFTER INTERRUPT
2360  022452  004737  023526                  JSR     PC,DMPREG
2361
2362  022456  012637  001126          9$:     MOV     (SP)+,$BDDAT     ;GET COUNT VALUE
2363  022462  005737  025022                  TST     STCCNT           ;IS THERE A VALID COUNT?
2364  022466  001402                          BEQ     .+6
2365  022470  000137  023200                  JMP     20$              ;IF YES, GO VERIFY CONSISTENCY.
2366  022474  012737  000011  001124          MOV     #9.,$GDDAT       ;NO -- SO START CHECKING VALUE.
2367  022502  023737  001126  001124          CMP     $BDDAT,$GDDAT    ;IS COUNT .LT 9.?
2368  022510  002002                          BGE     .+6              ;GO ON IF NO
2369  022512  000137  023170                  JMP     12$              ;ERROR IF YES (COUNTER TOO FAST)
2370  022516  012737  000024  001124          MOV     #20.,$GDDAT
2371  022524  023737  001126  001124          CMP     $BDDAT,$GDDAT    ;
2372  022532  003047                          BGT     10$              ;GREATER THAN 20.?
2373  022534  013737  001126  025022          MOV     $BDDAT,STCCNT    ;NO, SO BETWEEN 9 & 21 -- VALID
2374  022542  012737  000024  025026          MOV     #20.,MAXCNT      ;SAVE MAX LIMIT
2375  022550  012737  000011  025024          MOV     #9.,MINCNT       ;AND MIN LIMIT, FOR FUTURE TESTING.
2376  022556  005737  000042                  TST     @#42             ;ARE WE IN ACT?
2377  022562  001402                          BEQ     .+6              ;BEQ IF NO.
2378  022564  000137  023260                  JMP     99$              ;IF YES, DON'T PRINT.
2379  022570  104401  022576                  TYPE    ,65$             ;;TYPE ASCIZ STRING
  (1) 022574  000424                          BR      64$              ;;GET OVER THE ASCIZ
  (1)                             ;;65$:      .ASCIZ  <200>/SANITY TIMER SET FOR 256US PER COUNT./<200>
```

```
  (1)  022646                                64$:
2380   022646  000137  023260                      JMP     99$
2381
2382   022652  012737  000500  001124  10$:      MOV     #320.,$GDDAT
2383   022660  023737  001126  001124            CMP     $BDDAT,$GDDAT   ;CHECK FOR COUNT BETWEEN 320.
2384   022666  002540                            BLT     12$
2385   022670  012737  001200  001124            MOV     #640.,$GDDAT
2386   022676  023727  001126  001200            CMP     $BDDAT,#640.    ;AND 640.
2387   022704  003045                            BGT     11$
2388   022706  013737  001126  025022            MOV     $BDDAT,STCCNT   ;SAVE COUNT
2389   022714  012737  000500  025024            MOV     #320.,MINCNT    ;AND LIMITS
2390   022722  012737  001200  025026            MOV     #640.,MAXCNT
2391   022730  005737  000042                    TST     @#42            ;ARE WE IN ACT?
2392   022734  001151                            BNE     99$             ;IF YES, DON'T PRINT.
2393   022736  104401  022744                    TYPE    ,67$            ;;TYPE ASCIZ STRING
  (1)  022742  000425                            BR      66$             ;;GET OVER THE ASCIZ
  (1)                                ;;67$:     .ASCIZ  <200>/SANITY TIMER SET FOR 8.192MS PER COUNT./<200>
  (1)  023016                                66$:
2394   023016  000520                            BR      99$
2395
2396   023020  012737  001200  001124  11$:      MOV     #640.,$GDDAT
2397   023026  023737  001126  001124            CMP     $BDDAT,$GDDAT   ;COUNT .LT 640.?
2398   023034  002455                            BLT     12$             ;ERROR IF YES.
2399   023036  012737  002400  001124            MOV     #1280.,$GDDAT
2400   023044  023727  001126  002400            CMP     $BDDAT,#1280.   ;GREATER THAN 1280.?
2401   023052  003046                            BGT     12$             ;ERROR IF YES.
2402   023054  013737  001126  025022            MOV     $BDDAT,STCCNT   ;SAVE COUNT
2403   023062  012737  001200  025024            MOV     #640.,MINCNT    ;AND LIMITS
2404   023070  012737  002400  025026            MOV     #1280.,MAXCNT
2405   023076  005737  000042                    TST     @#42            ;ARE WE IN ACT?
2406   023102  001066                            BNE     99$
2407   023104  104401  023112                    TYPE    ,69$            ;;TYPE ASCIZ STRING
  (1)  023110  000426                            BR      68$             ;;GET OVER THE ASCIZ
  (1)                                ;;69$:     .ASCIZ  <200>/SANITY TIMER SET FOR 16.384MS PER COUNT./<200>
  (1)  023166                                68$:
2408   023166  000434                            BR      99$
2409
2410                                       ;HERE ON ERROR
2411   023170  104044                    12$:    ERROR+  44              ;CAN'T RESOLVE SANITY TIMER VALUE
2412   023172  004737  023526                    JSR     PC,DMPREG       ;(COUNTS NOT WITHIN LIMITS
2413   023176  000430                            BR      99$
2414
2415                                       ;HERE ON SUBSEQUENT PASSES TO CHECK CONSISTENCY
2416   023200  013737  025026  001124  20$:      MOV     MAXCNT,$GDDAT   ;GET EXPECTED MAX LIMIT FOR TYPEOUT.
2417   023206  013737  025022  024750            MOV     STCCNT,DISPLY   ;DISPLAY COUNT AS FOUND ON FIRST PASS.
2418   023214  023737  001126  001124            CMP     $BDDAT,$GDDAT   ;SEE THAT COUNT NOT EXCEEDED.
2419   023222  003404                            BLE     21$             ;OK IF .LE MAX LIMIT.
2420   023224  104044                            ERROR+  44              ;COUNT GREATER THAN MAX OF RANGE.
2421   023226  004737  023526                    JSR     PC,DMPREG
2422   023232  000412                            BR      99$
2423   023234  013737  025024  001124  21$:      MOV     MINCNT,$GDDAT   ;GET EXPECTED MIN LIMIT.
2424   023242  023737  001126  001124            CMP     $BDDAT,$GDDAT   ;CHECK
2425   023250  002003                            BGE     99$             ;OK IF ACTUAL .GE MIN LIMIT.
2426   023252  104044                            ERROR+  44              ;COUNT LESS THAN MIN OF RANGE.
2427   023254  004737  023526                    JSR     PC,DMPREG
2428
```

```
2429  023260  012677  001456          99$:    MOV     (SP)+,@IIIV     ;RESTORE VECTOR
2430  023264  012677  001454                  MOV     (SP)+,@IIIP     ;
2431  023270  012777  100000  001434          MOV     #CLR,@ACR       ;INIT THE INTERFACE
2432
2433
2434  023276  000005                          RESET
2435  023300  000240                          NOP
2436  023302  000137  032052                  JMP     $EOP
```

```
                                        .SBTTL  TEST AND UTILITY SUBROUTINES

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

023306  005037  023446         SETMNT: CLR     MNTBTS          ;CLEAR EXTRA MNT BITS
023312  012777  100000  001412 SETMN1: MOV     #CLR,@ACR       ;RESET THE INTERFACE
023320  016400  021762                 MOV     SIDTAB(4),R0    ;GET SID VALUE
023324  062700  000016                 ADD     #TMT,R0         ;ADD LOOPBACK + BOOT DISABLE TO MNT. SID
023330  063700  023446                 ADD     MNTBTS,R0       ;ADD ANY EXTRA MNT BITS
023334  027777  001372  001370         CMP     @ACR,@ACR       ;SHORT DELAY TO ALLOW RESET TO WORK
023342  027777  001364  001362         CMP     @ACR,@ACR
023350  012777  000015  001354         MOV     #MTCE,@ACR      ;CHOOSE MTC
023356  010077  001354                 MOV     R0,@MTC ;LOAD MTC
023362  012777  000001  001342         MOV     #PGCSE,@ACR     ;SELECT PGCS
023370  017737  001342  024750         MOV     @PGCS,DISPLY    ;READ PGCS
023376  032737  004000  024750         BIT     #PRDY,DISPLY    ;CHECK FOR RDY
023404  001003                 1$:     BNE     1$              ;SKIP IF SET
023406  104044                         ERROR+  44              ;DEVICE CLR DID NOT SET PGCS RDY
023410  004737  023526                 JSR     PC,DMPREG
023414                         SETMNX: MOV     #PGFE,@ACR      ;SELECT PGF
023414  012777  000005  001310         MOV     #7400,@PGF      ;CLEAR BOOT FLAGS
023422  012777  007400  001306         MOV     #7400,@STF      ;CLEAR BOOT FLAGS
023430  012777  007400  001300         MOV     #17,@DCF        ;CLEAR DCF FLAGS
023436  012777  000017  001272         RTS     PC              ;EXIT
023444  000207

023446  000000         MNTBTS: 0

023450  012746  023414 CLEARX: MOV     #SETMNX,-(SP)   ;GET INTERMEDIATE RETURN FOR FLAG CLEARING

                                ;ROUTINE TO INIT THE IIST, THEN DELAY A SHORT TIME

023454  012777  100000  001250 CLEAR:  MOV     #100000,@ACR    ;RESET THE INTERFACE
023462  012746  000030                 MOV     #30,-(SP)       ;GET A LOOP COUNTER
023466  005316         1$:     DEC     (SP)
023470  001376                 BNE     1$
023472  005726                 TST     (SP)+
023474  000207                 RTS     PC

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

023476  012737  004000  023446 SSTMNT: MOV     #MTYP,MNTBTS    ;SET TYP TO A 1
023504  000702                 BR      SETMN1          ;EXIT THROUGH SETMNT
```

```
2484
2485                                    ;PARITY CALCULATION SUBROUTINE
2486                                    ;COUNTS THE NUMBER OF 1 BITS IN R0 IN R1
2487
2488  023506  010046                    PARCAL: MOV     R0,-(SP)
2489  023510  005001                            CLR     R1
2490  023512  006300                    1$:     ASL     R0
2491  023514  005501                            ADC     R1
2492  023516  005700                            TST     R0
2493  023520  001374                            BNE     1$
2494  023522  012600                            MOV     (SP)+,R0
2495  023524  000207                            RTS     PC
```

```
                                        ;DUMP ALL REGISTERS ROUTINE

023526  032777  000001  155404  DMPREG: BIT     #BIT0,@SWR      ;TEST BIT0
023534  001001                          BNE     1$              ;SKIP IF SET
023536  000207                          RTS     PC              ;EXIT IF 0
023540  104401  042002          1$:     TYPE    ,HDRMES
023544  012737  000011  023642          MOV     #9.,TCR         ;DO 9 TIMES
023552  017746  001154                  MOV     @ACR,-(SP)      ;;PUSH @ACR ON STACK
023556  005077  001150                  CLR     @ACR            ;INIT ADDRESS
023562                          2$:
(1) 023562  017746  001150              MOV     @ADR,-(SP)      ;;SAVE @ADR FOR TYPEOUT
(1) 023566  104402                      TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
023570  104401  042112                  TYPE    ,TWOSP
023574  005337  023642                  DEC     TCR             ;DONE 9?
023600  001370                          BNE     2$              ;DO 9
023602  104401  042131                  TYPE    ,CARET
023606  012677  001120                  MOV     (SP)+,@ACR      ;;POP STACK INTO @ACR
023612  005777  155322                  TST     @SWR            ;CHECK FOR ERR HLT
023616  100002                          BPL     3$              ;SKIP IF 0
023620  000000                          HALT
023622  104407                          CKSWR
023624  032777  001000  155306  3$:     BIT     #BIT9,@SWR      ;CHECK FOR LOOP ON ERROR
023632  001402                          BEQ     4$              ;SKIP IF CLEAR
023634  013716  001110                  MOV     $LPERR,(SP)     ;IF LOOP ON ERROR, RETURN TO PRESET LOCATION
023640  000207                  4$:     RTS     PC              ;EXIT

023642  000000                  TCR:    0
```

```
                                ;ROUTINE TO GET DEVICE PARAMETERS
                                GETPAR:
(1) 023644 104401 023652                TYPE    ,65$                    ;;TYPE ASCIZ STRING
(1) 023650 000440                       BR      64$                     ;;GET OVER THE ASCIZ
(1)                             ;;65$:  .ASCIZ  <200>/DO YOU WANT TO CHANGE DEVICE PARAMETERS (CSR, ETC.)? [Y OR N]/
(1) 023752                      64$:
2527 023752 104410                      RDCHR
2528 023754 022627 000131               CMP     (SP)+,#'Y               ;SEE WHAT ANSWER IS.
2529 023760 001402                      BEQ     .+6                     ;IF Y(ES), ASK FOR PARAMETERS.
2530 023762 000137 024504               JMP     3$                      ;IF ANYTHING ELSE, ASSUME NO.
2531 023766 104401 023774               TYPE    ,67$                    ;;TYPE ASCIZ STRING
(1) 023772 000444                       BR      66$                     ;;GET OVER THE ASCIZ
(1)                             ;;67$:  .ASCIZ  <200>/IN THE FOLLOWING THREE PRINTOUTS, TYPE THE NEW DATA OR 0 IF NO CHA
(1) 024104                      66$:
2532 024104 104401 041706       10$:    TYPE    ,ASKDVA
2533 024110 013746 024732               MOV     ACR,-(SP)               ;;SAVE ACR FOR TYPEOUT
(1) 024114 104402                       TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2534 024116 104401 042110               TYPE    ,SPACE
2535 024122 104412                      RDOCT
2536 024124 012600                      MOV     (SP)+,R0                ;GET VALUE TYPED IN.
2537 024126 001402                      BEQ     11$                     ;IF ZERO, DON'T CHANGE.
2538 024130 010037 024732               MOV     R0,ACR                  ;STORE NEW VALUE
2539 024134 023727 024732 160000 11$:   CMP     ACR,#160000             ;CHECK FOR LEGALITY
2540 024142 103760                      BLO     10$                     ;LOOP IF BAD
2541 024144 032737 000003 024732        BIT     #3,ACR                  ;CHECK FOR MOD. 4
2542 024152 001354                      BNE     10$                     ;LOOP IF BAD
2543 024154 013737 024732 024734        MOV     ACR,ACRH                ;LOAD OTHER ADDRESS SLOTS
2544 024162 005237 024734               INC     ACRH
2545 024166 013737 024734 024736        MOV     ACRH,ADR
2546 024174 005237 024736               INC     ADR
2547 024200 013737 024736 024740        MOV     ADR,ADRH
2548 024206 005237 024740               INC     ADRH
2549 024212 104401 041726       1$:     TYPE    ,ASKPIV
2550 024216 013746 024742               MOV     IIIV,-(SP)              ;;SAVE IIIV FOR TYPEOUT
(1)                                                                     ;;1
(1) 024222 104403                       TYPOS                           ;;GO TYPE--OCTAL ASCII
(1) 024224    006                       .BYTE   6                       ;;TYPE 6 DIGITS
(1) 024225    000                       .BYTE   0                       ;;SUPPRESS LEADING ZEROS
2551 024226 104401 042110               TYPE    ,SPACE
2552 024232 104412                      RDOCT
2553 024234 012600                      MOV     (SP)+,R0                ;GET VALUE TYPED IN.
2554 024236 001402                      BEQ     12$                     ;IF 0, DON'T CHANGE.
2555 024240 010037 024742               MOV     R0,IIIV                 ;LOAD NEW VALUE
2556 024244 023727 024742 002000 12$:   CMP     IIIV,#2000              ;CHECK FOR LEGALITY
2557 024252 103357                      BHIS    1$                      ;LOOP IF BAD
2558 024254 032737 000003 024742        BIT     #3,IIIV                 ;CHECK FOR MOD. 4
2559 024262 001353                      BNE     1$                      ;LOOP IF BAD
2560 024264 013737 024742 024744        MOV     IIIV,IIIP               ;LOAD PIL
2561 024272 062737 000002 024744        ADD     #2,IIIP                 ;WITH IIIV+2
2562 024300 104401 041761       2$:     TYPE    ,ASKBRL
2563 024304 013746 024746               MOV     IIIL,-(SP)              ;;SAVE IIIL FOR TYPEOUT
(1)                                                                     ;;1
(1) 024310 104403                       TYPOS                           ;;GO TYPE--OCTAL ASCII
(1) 024312    006                       .BYTE   6                       ;;TYPE 6 DIGITS
```

```
  (1)  024313  000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
2564   024314  104401  042110              TYPE    ,SPACE
2565   024320  104412                      RDOCT
2566   024322  012600                      MOV     (SP)+,R0        ;GET VALUE TYPED IN.
2567   024324  001402                      BEQ     21$             ;IF ZERO, DON'T CHANGE.
2568   024326  010037  024746              MOV     R0,IIIL         ;GET BR LEVEL
2569   024332  023727  024746  000004  21$: CMP     IIIL,#4         ;CHECK FOR LEGALITY
2570   024340  103757                      BLO     2$              ;LOOP IF BAD
2571   024342  023727  024746  000007      CMP     IIIL,#7         ;CHECK
2572   024350  101353                      BHI     2$              ;LOOP IF BAD
2573   024352  000137  024504              JMP     3$              ;JUMP OVER HELP MSG.
2574   024356                          30$:
  (1)  024356  104401  024364              TYPE    ,69$            ;;TYPE ASCIZ STRING
  (1)  024362  000423                      BR      68$             ;;GET OVER THE ASCIZ
  (1)                                 ;;69$: .ASCIZ  <200>/TEST CONFIGURATION-SELECT CODES ARE:/
  (1)  024432                          68$:
2575   024432  012737  002762  001176      MOV     #MTAB,$TMP0     ;GET START OF MESSAGE POINTER TABLE.
2576   024440  104401  001225          31$: TYPE    ,$CRLF          ;OUTPUT NEWLINE
2577   024444  017737  154526  024454      MOV     @$TMP0,32$      ;GET ADDRESS OF TEXT LINE TO PRINT.
2578   024452  104401                      TYPE
2579   024454  000000                  32$: .WORD   0
2580   024456  062737  000002  001176      ADD     #2,$TMP0        ;BUMP POINTER
2581   024464  023727  001176  003004      CMP     $TMP0,#CNFMAX   ;AT END?
2582   024472  103762                      BLO     31$             ;IF NOT, GO FOR MORE.
2583   024474  104401  001225              TYPE    ,$CRLF          ;EXTRA CRLF
2584   024500  104401  001225              TYPE    ,$CRLF  ;NEW LINE
2585   024504                          3$:
  (1)  024504  104401  024512              TYPE    ,71$            ;;TYPE ASCIZ STRING
  (1)  024510  000437                      BR      70$             ;;GET OVER THE ASCIZ
  (1)                                 ;;71$: .ASCIZ  <200>/ENTER TEST CONFIGURATION SELECTION (0-10) OR 111 FOR HELP: /
  (1)  024610                          70$:
2586   024610  104412                      RDOCT
2587   024612  012637  025020              MOV     (SP)+,CONFIG    ;GET VALUE
2588   024616  023737  025020  003004      CMP     CONFIG,CNFMAX   ;SEE IF VALID
2589   024624  101402                      BLOS    .+6
2590   024626  000137  024356              JMP     30$             ;IF NOT, DO HELP MSG.
2591   024632  000207                      RTS     PC              ;EXIT
```

```
                        ;ROUTINE TO REPORT THE SELF ID OF THE IIST
                        
024634  005737  001236  GETSID: TST     $PASS           ;PASS 1?
024640  001401                 BEQ     1$              ;SKIP IF YES
024642  000207                 RTS     PC              ;EXIT IF NOT
024644  005737  000042  1$:    TST     @#42            ;ARE WE IN ACT?
024650  001002                 BNE     2$              ;IF YES, DON'T PRINT
024652  104401  042115         TYPE    ,DMPSID
024656  012777  100000  000046 2$:    MOV     #CLR,@ACR       ;CLEAR INTERFACE
024664  017700  000042         MOV     @ACR,R0         ;GET SID
024670  000300                 SWAB    R0              ;INTO B2-0
024672  042700  177770         BIC     #177770,R0      ;CLEAR B15-3
024676  010037  024730         MOV     R0,SIDNUM       ;SAVE SELF ID NUM.
024702  005737  000042         TST     @#42            ;IN ACT
024706  001007                 BNE     3$              ;DON'T PRINT IF YES.
024710  010046                 MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
 (1)                           ;;1
 (1)    024712  104403         TYPOS                   ;;GO TYPE--OCTAL ASCII
 (1)    024714     006         .BYTE   6               ;;TYPE 6 DIGITS
 (1)    024715     000         .BYTE   0               ;;SUPPRESS LEADING ZEROS
024716  104401  042131         TYPE    ,CARET
024722  104401  042131         TYPE    ,CARET
024726  000207         3$:     RTS     PC              ;EXIT

024730  000000         SIDNUM: 0
```

```
                                        .SBTTL  DEVICE ADDRESS PARAMETER BLOCK

024732  177500          ACR:    177500          ;ACCESS CONTROL REGISTER
024734  177501          ACRH:   177501          ;HIGH BYTE OF ACR

024736                  ADR:                    ;ACCESS DATA REGISTER
024736                  PGTE:                   ;PGM-GEN. XMIT ENABLES REGISTER
024736                  PGCS:                   ;PGM-GEN. CSR
024736                  STTE:                   ;SANITY TIMER XMIT ENABLES REGISTER
024736                  STCS:                   ;SANITY TIMER CSR
024736                  IMSK:                   ;INPUT MASKS REGISTER
024736                  PGF:                    ;PGM-GEN. I/B FALAGS REGISTER
024736                  STF:                    ;SANITY TIMER FLAGS REGISTER
024736                  DCF:                    ;DCLO/DISCONNECT FLAGS REGISTER
024736                  EXC:                    ;EXCEPTIONS REGISTER
024736  177502          MTC:    177502          ;MAINTENANCE REGISTER
024740                  ADRH:                   ;ADR HIGH BYTE
024740                  PGCSH:                  ;PGCS HIGH BYTE
024740                  STCSH:                  ;STCS HIGH BYTE
024740                  IMSKH:                  ;IMSK HIGH BYTE
024740                  PGFH:                   ;PGF HIGH BYTE
024740                  DCFH:                   ;DCF HIGH BYTE
024740                  EXCH:                   ;EXC HIGH BYTE
024740  177503          MTCH:   177503          ;MTC HIGH BYTE
024742  000260          IIIV:   260             ;IIST INTERRUPT VECTOR ADDRESS
024744  000262          IIIP:   262             ;IIST INTERRUPT PRIORITY ADDRESS
024746  000006          IIIL:   6               ;IIST INTERRUPT LEVEL


                        ;CONSTANTS AND VARIABLES

024750  000000          DISPLY: 0
024752  000000          TIMPER: 0
024754  000000          LINMBR: 0
024756  000020          SNAPO:  .BLKW   16.

025016  000000          PARFLG: .WORD   0

025020  000000          CONFIG: .WORD   0

025022  000000          STCCNT: .WORD   0       ;COUNT OF # PG CHARACTERS TRANSMITTED DURING
                                                ;TWO COUNTS OF THE SANITY TIMER.
025024  000000          MINCNT: .WORD   0       ;MINIMUM ALLOWED VALUE OF STCCNT.
025026  000000          MAXCNT: .WORD   0       ;MAXIMUM ALLOWED VALUE OF STCCNT.
```

```
                                .SBTTL  MAINTENANCE-MODE SELF-BOOT TEST

                        ;MAINTENANCE-MODE BOOT TEST.
                        ;USES MAINTENANCE LOOPBACK TO GENERATE BOOT TO THIS CPU.
                        ;RECEIVE CHANNEL AND TYPE (PG OR ST) SELECTED VIA OPERATOR INPUT.
                        ;  (0-3) FOR PG BOOTS, (10-13) FOR ST BOOTS.
                        ;THIS TEST VERIFIES OPERATION OF THE BOOT-PULSE COMBINING LOGIC,
                        ;THE BOOT ONE-SHOT, THE DRIVER, AND THE CABLE.

025030  000004          MMBOOT: SCOPE
025032  004737  024634          JSR     PC,GETSID       ;GET SELF-ID
025036  005227  177777          INC     #-1             ;FIRST TIME?
025042  001402                  BEQ     .+6
025044  000137  025476          JMP     1$              ;IF NOT, SKIP EXPLANATION.

025050  104401  025056          TYPE    ,65$            ;;TYPE ASCIZ STRING
025054  000434                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <CRLF>/AFTER SELECTING THE OPERATING CODE, FOLLOWED BY (CR),/
025146                  64$:
025146  104401  025154          TYPE    ,67$            ;;TYPE ASCIZ STRING
025152  000431                  BR      66$             ;;GET OVER THE ASCIZ
                        ;;67$:  .ASCIZ  <CRLF>/  THE OPERATOR MUST VERIFY THAT THE PROPER BOOT/
025236                  66$:
025236  104401  025244          TYPE    ,69$            ;;TYPE ASCIZ STRING
025242  000426                  BR      68$             ;;GET OVER THE ASCIZ
                        ;;69$:  .ASCIZ  <CRLF>/  ACTION TOOK PLACE.  OPERATING CODES ARE:/
025320                  68$:
025320  104401  025326          TYPE    ,71$            ;;TYPE ASCIZ STRING
025324  000431                  BR      70$             ;;GET OVER THE ASCIZ
                        ;;71$:  .ASCIZ  <CRLF>/     0-3 SELECTS PROGRAMMED BOOT, RECEIVERS 0-3/
025410                  70$:
025410  104401  025416          TYPE    ,73$            ;;TYPE ASCIZ STRING
025414  000430                  BR      72$             ;;GET OVER THE ASCIZ
                        ;;73$:  .ASCIZ  <CRLF>/     10-13 SELECTS SANITY BOOT, RECEIVERS 0-3/<CRLF>
025476                  72$:

025476  005777  153444  1$:     TST     @$TKB           ;FLUSH KEYBOARD
025502  104401  025510          TYPE    ,75$            ;;TYPE ASCIZ STRING
025506  000412                  BR      74$             ;;GET OVER THE ASCIZ
                        ;;75$:  .ASCIZ  <CRLF>/ENTER BOOT CODE: /
025534                  74$:
025534  104412                  RDOCT                   ;GET OCTAL INPUT
025536  011637  024754          MOV     (SP),LINMBR     ;SAVE VALUE
025542  012600                  MOV     (SP)+,R0        ;AND A COPY
025544  032700  177764          BIT     #177764,R0      ;IS IT A VALID CODE?
025550  001427                  BEQ     2$              ;IF YES, GO ON.
025552  104401  025560          TYPE    ,77$            ;;TYPE ASCIZ STRING
025556  000423                  BR      76$             ;;GET OVER THE ASCIZ
                        ;;77$:  .ASCIZ  <CRLF>/ILLEGAL CODE; MUST BE 0-3 OR 10-13/<CRLF>
025626                  76$:
025626  000723                  BR      1$              ;GO BACK FOR ANOTHER

025630  012737  025630  001110  2$:     MOV     #2$,$LPERR      ;SET UP FOR ERROR LOOP
025636  004737  023306          JSR     PC,SETMNT       ;INIT, THEN SETUP MAINT. MODE
025642  012777  000015  177062          MOV     #MICE,@ACR      ;SELECT MTC
025650  012777  000003  177060          MOV     #MLEN+MDSD,@MTC ;SET PROPER MODE.
```

```
 2696  025656  013700  024754                 MOV     LINMBR,R0       ;GET CODE
 2697  025662  010037  024750                 MOV     R0,DISPLY       ;MAKE COPY
 2698  025666  042737  000010  024754         BIC     #10,LINMBR
 2699  025674  042700  000010                 BIC     #10,R0          ;MASK OUT TYPE BIT
 2700  025700  006300                         ASL     R0              ;MAKE WORD INDEX
 2701  025702  016001  022002                 MOV     PGBTAB(R0),R1   ;GET SELECTED BIT
 2702  025706  005101                         COM     R1              ;INVERT TO CONFIGURE MASKS.
 2703  025710  012777  000004  177014         MOV     #IMSKE,@ACR     ;SELECT IMASK REG.
 2704  025716  010177  177014                 MOV     R1,@IMSK        ;SET ALL MASKS EXCEPT ONE.
 2705  025722  010137  001176                 MOV     R1,$TMP0        ;$TMP0 IS MASK VALUE
 2706  025726  016037  022002  001124         MOV     PGBTAB(R0),$GDDAT       ;BOOT ENABLE
 2707  025734  013701  024730                 MOV     SIDNUM,R1       ;GET SELF-ID.
 2708  025740  006301                         ASL     R1              ;MAKE WORD INDEX
 2709  025742  032737  000010  024750         BIT     #10,    DISPLY  ;WHAT TYPE OF BOOT?
 2710  025750  001036                         BNE     5$              ;BRANCH IF ST
 2711
 2712                                 ;DO PROGRAMMED BOOT:
 2713  025752  012777  000000  176752         MOV     #PGTEE,@ACR     ;SELECT PGTE REGISTER.
 2714  025760  016177  022002  176750         MOV     PGBTAB(R1),@PGTE        ;ENABLE BOOT TO SELF.
 2715  025766  012777  000001  176742         MOV     #GO,@PGCS       ;INITIATE BOOT SEQUENCE.
 2716  025774  005002                         CLR     R2
 2717  025776  062702  000001                 ADD     #1,R2           ;STALL
 2718  026002  001375                         BNE     .-4
 2719  026004  012777  000010  176720         MOV     #EXCE,@ACR      ;SELECT EXC
 2720  026012  017737  176720  001200         MOV     @EXC,$TMP1      ;GET EXC INTO $TMP1
 2721  026020  012777  000005  176704         MOV     #PGFE,@ACR      ;SELECT PGF
 2722  026026  017737  176704  001126         MOV     @PGF,$BDDAT     ;GET IT
 2723  026034  104044                         ERROR   +44             ;NO BOOT OCCURRED
 2724  026036  004737  023526                 JSR     PC,DMPREG
 2725  026042  000137  025476                 JMP     1$              ;TRY SOME MORE?
 2726
 2727                                 ;DO SANITY-TIMER BOOT (AND HALT):
 2728  026046  012777  000002  176656  5$:    MOV     #STTEE,@ACR     ;SELECT ST TRANSMIT ENABLES
 2729  026054  016177  022002  176654         MOV     STBTAB(R1),@STTE        ;SET TO XMIT TO SELF
 2730  026062  012777  000005  176646         MOV     #LKE+ENB,@STCS  ;ENABLE CLOCK AND HALT
 2731  026070  005002                         CLR     R2
 2732  026072  005777  176634          6$:    TST     @ACR            ;STALL
 2733  026076  005777  176630                 TST     @ACR
 2734  026102  005777  176624                 TST     @ACR
 2735  026106  005777  176620                 TST     @ACR
 2736  026112  005202                         INC     R2
 2737  026114  001366                         BNE     6$
 2738  026116  012777  000010  176606         MOV     #EXCE,@ACR      ;SELECT EXC
 2739  026124  017737  176606  001200         MOV     @EXC,$TMP1      ;GET IT
 2740  026132  012777  000006  176572         MOV     #STFE,@ACR      ;SELECT STF
 2741  026140  017737  176572  001126         MOV     @STF,$BDDAT     ;SELECT STF
 2742  026146  104044                         ERROR   +44             ;NO BOOT (OR EVEN HALT) OCCURRED
 2743  026150  004737  023526                 JSR     PC,DMPREG
 2744  026154  000137  025476                 JMP     1$              ;TRY FOR ANOTHER?
```

```
                                        .SBTTL  MULTIPROCESSOR BOOT TEST

                                ;ALLOWS OPERATOR TO INITIATE BOOTS TO OTHER CPU'S.
                                ;IF THE PROGRAM IN THE BOOTSTRAP ROM SETS THE DIP11-A
                                ;INTERRUPT ENABLE BIT, THIS TEST SHOULD BE STARTED IN EACH
                                ;CPU.  THE BOOTED CPU CAN THEN BE INTERRUPTED TO REPORT THE FACT.

026160  000004                  MPDIBT: SCOPE
026162  004737  024634                  JSR     PC,GETSID               ;GET SELF-ID
026166  012737  000340  177776  MPDBT1: MOV     #PR7,PSW                ;LOCK OUT INTERRUPTS
026174  004737  023450                  JSR     PC,CLEARX               ;INIT THE INTERFACE & FLAGS
026200  012777  026714  176534          MOV     #MPDINT,@IIIV           ;SET UP IIST VECTOR
026206  012777  000340  176532          MOV     #PR7,@IIIL              ;SET TO PRI 7 ON INTERRUPT

026214  012777  000001  176510  1$:     MOV     #PGCSE,@ACR     ;SELECT PGCS REGISTER
026222  012777  000004  176506          MOV     #IE,@PGCS       ;SET INTR ENB TO CATCH ERRORS
026230  012706  001100                  MOV     #STACK,SP       ;SETUP STACK POINTER
026234  005037  177776                  CLR     PSW             ;ALLOW INTERRUPTS
026240  005777  152702                  TST     @$TKB           ;FLUSH KBD BUFFER OF OLD STUFF
026244  104401  026252                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 (1)    026250  000415                  BR      64$             ;;GET OVER THE ASCIZ
 (1)                            ;;65$:  .ASCIZ  <CRLF>/ENTER # OF CPU TO BOOT: /
 (1)    026304                  64$:

026304  005037  001124  10$:     CLR     $GDDAT          ;EXPECT PGF AND STF TO BE CLEAR
026310  012777  000005  176414          MOV     #PGFE,@ACR      ;SELECT PGF REGISTER
026316  017737  176414  001126          MOV     @PGF,$BDDAT     ;GET PGF AND TEST IT
026324  001411                          BEQ     11$             ;OK IF 0 (WE SHOULDN'T SEE FLAGS HERE)
026326  104044                          ERROR   +44             ;ERROR -- PGF FLAGS PRESENT BUT NO BOOTING
026330  004737  023526                  JSR     PC,DMPREG
026334  012777  000005  176370          MOV     #PGFE,@ACR      ;SELECT PGF AGAIN
026342  012777  007417  176366          MOV     #7417,@PGF      ;CLEAR THE FLAGS
026350  012777  000006  176354  11$:     MOV     #STFE,@ACR      ;SELECT STF REGISTER
026356  017737  176354  001126          MOV     @STF,$BDDAT     ;GET STF AND TEST IT
026364  001411                          BEQ     12$             ;OK IF 0
026366  104044                          ERROR   +44             ;ERROR -- STF FLAGS PRESENT
026370  004737  023526                  JSR     PC,DMPREG
026374  012777  000006  176330          MOV     #STFE,@ACR      ;SELECT STF AGAIN
026402  012777  007417  176326          MOV     #7417,@STF      ;CLEAR THE FLAGS
026410  105777  152530          12$:    TSTB    @$TKS           ;LOOK FOR KBD INPUT
026414  100333                          BPL     10$             ;LOOP BACK IF NONE.

026416  104412                          RDOCT                   ;GET OCTAL INPUT WHEN PRESENT
026420  011637  024750                  MOV     (SP),DISPLY     ;SAVE THE # OF CPU TO BE BOOTED
026424  022627  000003                  CMP     (SP)+,#3        ;SEE IF IT'S LEGAL (0-3)
026430  101427                          BLOS    2$              ;BR IF YES - IF NOT, SAY SO
026432  104401  026440                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 (1)    026436  000423                  BR      66$             ;;GET OVER THE ASCIZ
 (1)                            ;;67$:  .ASCIZ  <CRLF>/ILLEGAL CODE; MUST BE IN RANGE 0-3/<CRLF>
 (1)    026506                  66$:
026506  000642                          BR      1$              ;GO ASK FOR IT AGAIN

                                ;COME HERE ON VALID INPUT FROM KEYBOARD.

026510  013700  024750          2$:     MOV     DISPLY,R0       ;GET CPU SELECT CODE
026514  006300                          ASL     R0              ;MAKE IT A WORD INDEX
```

```
2796  026516  012777  000000  176206          MOV     #PGTEE,@ACR     ;SELECT PG XMIT ENABLES REGISTER
2797  026524  016077  022002  176204          MOV     PGBTAB(R0),@PGTE  ;SELECT CPU TO BOOT
2798  026532  052777  000001  176176          BIS     #GO,@PGCS       ;SEND THE BOOT
2799  026540  012702  000004                  MOV     #4,R2           ;INITIALIZE OUTER COUNTER FOR STALL
2800  026544  005003                  3$:     CLR     R3              ;ZERO THE INNER COUNTER
2801  026546  005777  176160          4$:     TST     @ACR            ;STALL TO ALLOW BOOTING TO OCCUR
2802  026552  005777  176154                  TST     @ACR            ;AND INIT TO FINISH ON TARGET CPU.
2803  026556  005777  176150                  TST     @ACR            ;EACH "TST @ACR" SHOULD BURN UP AT
2804  026562  005777  176144                  TST     @ACR            ;LEAST 2 MICROSECONDS
2805  026566  005777  176140                  TST     @ACR            ;TO ALLOW AT LEAST 4 SECONDS OF STALL.
2806  026572  005777  176134                  TST     @ACR
2807  026576  005203                          INC     R3              ;BUMP INNER COUNTER.
2808  026600  001362                          BNE     4$              ;LOOP 'TIL IT GOES TO ZERO.
2809  026602  005302                          DEC     R2              ;BUMP OUTER COUNTER
2810  026604  001357                          BNE     3$              ;LOOP 'TIL IT COUNTS TO ZERO.
2811  026606  013700  024750                  MOV     DISPLY,R0       ;GET TARGET CPU # AGAIN
2812  026612  006300                          ASL     R0              ;MAKE WORD INDEX
2813  026614  012777  000000  176110          MOV     #PGTEE,@ACR     ;SELECT PGTE AGAIN
2814  026622  016077  021772  176106          MOV     PGITAB(R0),@PGTE        ;SELECT CPU TO BE INTERRUPTED
2815  026630  052777  000001  176100          BIS     #GO,@PGCS       ;SEND THE INTERRUPT
2816  026636  104401  026644                  TYPE    ,69$            ;;TYPE ASCIZ STRING
 (1)  026642  000416                          BR      68$             ;;GET OVER THE ASCIZ
 (1)                          ;;69$:  .ASCIZ  <CRLF>/BOOT & INTR SENT TO CPU # /
 (1)  026700                  68$:
2817  026700  013746  024750                  MOV     DISPLY,-(SP)    ;PRINT THE CPU.
2818  026704  104403                          TYPOS
2819  026706     001     000                  .BYTE   1,0
2820  026710  000137  026214                  JMP     1$              ;GO FOR MORE.
2821
2822                                ;COME HERE ON INTERRUPT
2823
2824  026714  012706  001100          MPDINT: MOV     #STACK,SP       ;REESTABLISH STACK, ASSUMING BOOT CHANGED IT.
2825  026720  012777  000010  176004          MOV     #EXCE,@ACR      ;SELECT EXC (EXCEPTIONS) REGISTER
2826  026726  005037  001124                  CLR     $GDDAT          ;EXPECT NO UIF OR RTE FLAGS.
2827  026732  017737  176000  001126          MOV     @EXC,$BDDAT     ;GET AND TEST EXC
2828  026740  001411                          BEQ     1$              ;OK, BRANCH IF 0.
2829  026742  104044                          ERROR   +44             ;ERROR - EXC IS NONZERO.
2830  026744  004737  023526                  JSR     PC,DMPREG
2831  026750  012777  000010  175754          MOV     #EXCE,@ACR      ;SELECT EXC AGAIN
2832  026756  012777  007417  175752          MOV     #7417,@EXC      ;CLEAR THE FLAGS
2833  026764  012777  000007  175740  1$:     MOV     #DCFE,@ACR      ;SELECT DCF REGISTER.
2834  026772  017737  175740  001126          MOV     @DCF,$BDDAT     ;GET DCF
2835  027000  013737  001126  001124          MOV     $BDDAT,$GDDAT   ;COPY IT
2836  027006  042737  000017  001124          BIC     #17,$GDDAT      ;EXPECT NO DCF FLAGS, BUT BRKS OK.
2837  027014  023737  001126  001124          CMP     $BDDAT,$GDDAT   ;CHECK THE DATA
2838  027022  001411                          BEQ     2$              ;BRANCH IF OK
2839  027024  104044                          ERROR   +44             ;ERROR - DCF FLAGS NONZERO
2840  027026  004737  023526                  JSR     PC,DMPREG
2841  027032  012777  000007  175672          MOV     #DCFE,@ACR      ;SELECT DCF AGAIN
2842  027040  012777  000017  175670          MOV     #17,@DCF        ;CLEAR THE FLAGS.
2843
2844  027046  012777  000006  175656  2$:     MOV     #STFE,@ACR      ;SELECT SANITY-TIMER FLAGS REGISTER
2845  027054  005037  001124                  CLR     $GDDAT          ;EXPECT NO ST FLAGS
2846  027060  017737  175652  001126          MOV     @STF,$BDDAT     ;GET AND TEST STF CONTENTS
2847  027066  001411                          BEQ     3$              ;BR IF ZERO
2848  027070  104044                          ERROR   +44             ;ERROR -- STF IS NOT ZERO
```

```
2849  027072  004737  023526                    JSR     PC,DMPREG
2850  027076  012777  000006  175626            MOV     #STFE,@ACR      ;SELECT STF AGAIN
2851  027104  012777  007417  175624            MOV     #7417,@STF      ;CLEAR ALL STF FLAGS
2852
2853  027112  005037  001124          3$:       CLR     $GDDAT          ;EXPECT NO FLAGS IN PGTE (BOOTING SHOULD CLEAR)
2854  027116  012777  000000  175606            MOV     #PGTEE,@ACR     ;SELECT PGTE REGISTER
2855  027124  017737  175606  001126            MOV     @PGTE,$BDDAT    ;GET AND TEST PGTE
2856  027132  001403                            BEQ     4$              ;BR IF 0
2857  027134  104044                            ERROR   +44             ;ERROR -- PGTE NOT ZERO
2858  027136  004737  023526                    JSR     PC,DMPREG
2859
2860  027142  012777  000002  175562  4$:       MOV     #STTEE,@ACR     ;SELECT STTE REGISTER
2861  027150  017737  175562  001126            MOV     @STTE,$BDDAT    ;GET STTE AND TEST FOR ZERO
2862  027156  001403                            BEQ     5$              ;OK OF ZERO
2863  027160  104044                            ERROR   +44             ;ERROR - STTE NOT ZERO
2864  027162  004737  023526                    JSR     PC,DMPREG
2865
2866  027166  012777  000003  175536  5$:       MOV     #STCSE,@ACR     ;SELECT STCS REGISTER
2867  027174  017737  175536  001126            MOV     @STCS,$BDDAT    ;GET STCS AND TEST FOR 0
2868  027202  001403                            BEQ     6$              ;OK IF ZERO
2869  027204  104044                            ERROR   +44             ;ERROR - STCS NOT ZERO
2870  027206  004737  023526                    JSR     PC,DMPREG
2871
2872  027212  012737  004014  001124  6$:       MOV     #IP+IE+PRDY,$GDDAT      ;EXPECT BITS IN PGCS
2873  027220  013700  024730                    MOV     SIDNUM,R0       ;GET SELF-ID
2874  027224  006300                            ASL     R0              ;MAKE WORD INDEX
2875  027226  056037  021762  001124            BIS     SIDTAB(R0),$GDDAT  ;INSERT SELF-ID INTO EXPECTED DATA
2876  027234  012777  000001  175470            MOV     #PGCSE,@ACR     ;SELECT PGCS REGISTER
2877  027242  017737  175470  001126            MOV     @PGCS,$BDDAT    ;GET PGCS
2878  027250  023737  001126  001124            CMP     $BDDAT,$GDDAT   ;CHECK THE RESULT
2879  027256  001403                            BEQ     7$              ;BRANCH IF OK
2880  027260  104044                            ERROR   +44             ;ERROR -- PGCS INCORRECT
2881  027262  004737  023526                    JSR     PC,DMPREG
2882
2883  027266  012777  000001  175436  7$:       MOV     #PGCSE,@ACR     ;SELECT PGCS REGISTER
2884  027274  012777  100000  175434            MOV     #100000,@PGCS   ;CLEAR ERR AND IE
2885  027302  012777  000005  175422            MOV     #PGFE,@ACR      ;SELECT PGF REGISTER
2886  027310  017737  175422  024750            MOV     @PGF,DISPLY     ;GET PGF
2887  027316  005037  001126                    CLR     $BDDAT
2888  027322  005037  001124                    CLR     $GDDAT
2889  027326  113737  024751  001124            MOVB    DISPLY+1,$GDDAT ;GET PG BOOT FLAGS
2890  027334  113737  024750  001126            MOVB    DISPLY,$BDDAT   ;GET PG INTR FLAGS
2891  027342  023737  001126  001124            CMP     $BDDAT,$GDDAT   ;PG BOOT & INTR FLAGS SHOULD BE EQUAL
2892  027350  001403                            BEQ     8$              ;BRANCH IF THEY ARE.
2893  027352  104044                            ERROR   +44             ;ERROR -- PG BOOT & INTR FLAGS NOT SAME
2894  027354  004737  023526                    JSR     PC,DMPREG
2895
2896  027360  013700  001124          8$:       MOV     $GDDAT,R0       ;GET BOOT FLAGS
2897  027364  001004                            BNE     9$              ;BRANCH IF NONZERO
2898  027366  104044                            ERROR   +44             ;ERROR -- NO BOOT FLAGS
2899  027370  004737  023526                    JSR     PC,DMPREG
2900  027374  000427                            BR      12$             ;CAN' REPORT BOOT
2901
2902  027376  005046          9$:       CLR     -(SP)           ;CLEAR COUNTER
2903  027400  000241          10$:      CLC                     ;CLEAR C-BIT IN PREP. FOR ROTATE
2904  027402  006000                            ROR     R0              ;SHIFT FLAGS INTO C-BIT
```

```
 2905  027404  103402                        BCS     11$             ;FOUND IT IF C SET
 2906  027406  005216                        INC     (SP)            ;IF NOT, GO AGAIN
 2907  027410  000773                        BR      10$
 2908  027412                          11$:
  (1)  027412  104401  027420                TYPE    ,65$            ;;TYPE ASCIZ STRING
  (1)  027416  000412                        BR      64$             ;;GET OVER THE ASCIZ
  (1)                             ;;65$:     .ASCIZ  <CRLF>/BOOTED FROM CPU #/
  (1)  027444                     64$:
 2909  027444  104403                        TYPOS                   ;OUTPUT THE CPU #
 2910  027446     001     000                .BYTE   1,0
 2911  027450  104401  001225                TYPE    ,$CRLF          ;AND A NEWLINE
 2912  027454  012777  000005  175250  12$:  MOV     #PGFE,@ACR      ;SELECT PGF
 2913  027462  012777  007417  175246        MOV     #7417,@PGF      ;CLEAR FLAGS
 2914  027470  012706  001100                MOV     #STACK,SP
 2915  027474  000137  026166                JMP     MPDBT1
```

```
                        .SBTTL  SANITY-TIMER TIMEOUT HALT/FREEZE TEST

                        ;ALLOWS OPERATOR TO START THE SANITY TIMER AND VERIFY IF HALT OCCURS.

027500 000004           STHLT:  SCOPE
027502 004737 024634            JSR     PC,GETSID       ;GET SELF-ID
027506 004737 023450    2$:     JSR     PC,CLEARX       ;INIT
027512 005777 151430            TST     @$TKB           ;FLUSH KEYBOARD OF OLD STUFF.
027516 104401 027524            TYPE    ,65$            ;;TYPE ASCIZ STRING
027522 000434                   BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <CRLF>/TYPE ANY CHARACTER WHEN READY TO INITIATE HALT ACTION;/
027614                  64$:
027614 104401 027622            TYPE    ,67$            ;;TYPE ASCIZ STRING
027620 000425                   BR      66$             ;;GET OVER THE ASCIZ
                        ;;67$:  .ASCIZ  <CRLF>#  THEN VERIFY THAT CPU IS HALTED/FROZEN.#
027674                  66$:
027674 104410                   RDCHR                   ;GET INPUT
027676 012706 001100            MOV     #STACK,SP
027702 012777 000003 175022     MOV     #STCSE,@ACR     ;SELECT STCS REGISTER
027710 012777 000005 175020     MOV     #LKE+ENB,@STCS  ;SET LOCK-ENABLE AND START CLOCK
027716 005000                   CLR     R0              ;INIT STALL COUNTER
027720 005777 175006    1$:     TST     @ACR
027724 005777 175002            TST     @ACR
027730 005777 174776            TST     @ACR
027734 005200                   INC     R0              ;STALL
027736 001370                   BNE     1$              ;LOOP

                        ;ERROR IF WE FALL THROUGH -- NO HALT OCCURRED
                        ;  OR MAYBE OPERATOR JUST HIT "CONTINUE"
027740 104044                   ERROR   +44
027742 004737 023526            JSR     PC,DMPREG
027746 000657                   BR      2$
```

```
                                        .SBTTL  MANUAL CONTROL-PANEL TEST

                                ;TEST MANUAL CONTROL PANEL (INHIBIT INPUTS) VIA DIALOG.
                                ;FOR TIME BEING, WE JUST PRINT OUT DCF REGISTER WHEN IT CHANGES.

027750  000004                  MCP:    SCOPE
027752  004737  023450                  JSR     PC,CLEARX       ;INIT THE INTERFACE
027756  004737  024634                  JSR     PC,GETSID       ;SET SELF-ID
027762  104401  027770                  TYPE    ,65$            ;;TYPE ASCIZ STRING
027766  000415                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/INITIAL DCF CONTENTS = /
030022                          64$:
030022  012777  000007  174702          MOV     #DCFE,@ACR      ;SELECT DCF
030030  017737  174702  024750          MOV     @DCF,DISPLY     ;GET INITIAL CONTENTS OF DCF
030036  013746  024750                  MOV     DISPLY,-(SP)
030042  104402                          TYPOC                   ;TYPE IT OUT.

030044  012777  000007  174660  1$:     MOV     #DCFE,@ACR      ;SELECT DCF
030052  017737  174660  001124          MOV     @DCF,$GDDAT     ;GET IT
030060  023737  001124  024750          CMP     $GDDAT,DISPLY   ;COMPARE WITH OLD VALUE
030066  001766                          BEQ     1$              ;JUST LOOK AGAIN IF EQUAL

030070  104401  030076                  TYPE    ,67$            ;;TYPE ASCIZ STRING
030074  000411                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <CRLF>/DCF CHANGED TO /
030120                          66$:
030120  013746  001124                  MOV     $GDDAT,-(SP)
030124  104402                          TYPOC                   ;PRINT IT
030126  012777  000007  174576          MOV     #DCFE,@ACR      ;SELECT DCF
030134  012777  000017  174574          MOV     #17,@DCF        ;CLEAR FLAGS
030142  013737  001124  024750          MOV     $GDDAT,DISPLY   ;MAKE NEW OLD
030150  042737  000017  024750          BIC     #17,DISPLY
030156  000732                          BR      1$
```

```
                                        .SBTTL  SCOPE LOOPS

                                ;ROUTINES FOR CYCLING THE LOGIC REPEATEDLY.

030160  000004                 SLOOPS: SCOPE
030162  004737  024634                 JSR     PC,GETSID       ;GET SELF-ID INTO SIDNUM
030166  005227  177777                 INC     #-1             ;FIRST TIME SINCE LOADING?
030172  001402                         BEQ     1$              ;IF YES, PRINT EXPLANATIONS.
030174  000137  031232                 JMP     SLOOP1          ;IF NO, JUST GO ASK FOR WHAT TO DO.

030200  005777  150742         1$:     TST     @$TKB           ;FLUSH KEYBOARD OF STUFF HANGING AROUND.
030204  104401  030214                 TYPE    ,SLMSG          ;TYPE MESSAGE.
030210  000137  031232                 JMP     SLOOP1          ;NOW GO SEE WHAT TO DO

030214  052200  042510  051440  SLMSG: .ASCII  <CRLF>/THE SCOPE LOOPS ARE USED FOR CYCLING THE/
030222  047503  042520  046040
030230  047517  051520  040440
030236  042522  052440  042523
030244  020104  047506  020122
030252  054503  046103  047111
030260  020107  044124     105
030265     040  047514  044507          .ASCII  / LOGIC WHILE MONITORING WITH AN/<200>/ OSCILLOSCOPE./
030272  020103  044127  046111
030300  020105  047515  044516
030306  047524  044522  043516
030314  053440  052111  020110
030322  047101  020200  051517
030330  044503  046114  051517
030336  047503  042520     056
030343     200  042523  042514          .ASCII  <CRLF>/SELECT THE LOOP FROM THE FOLLOWING MENU BY TYPING/
030350  052103  052040  042510
030356  046040  047517  020120
030364  051106  046517  052040
030372  042510  043040  046117
030400  047514  044527  043516
030406  046440  047105  020125
030414  054502  052040  050131
030422  047111     107
030425     200  044440  051524          .ASCII  <200>/ ITS OCTAL CODE FOLLOWED BY CARRIAGE RETURN./
030432  047440  052103  046101
030440  041440  042117  020105
030446  047506  046114  053517
030454  042105  041040  020131
030462  040503  051122  040511
030470  042507  051040  052105
030476  051125  027116
030502  052200  042510  043040          .ASCII  <CRLF>/THE FOLLOWING OPERATIONS AND THEIR RESPECTIVE CODES ARE AVAILABLE
030510  046117  047514  044527
030516  043516  047440  042520
030524  040522  044524  047117
030532  020123  047101  020104
030540  044124  044505  020122
030546  042522  050123  041505
030554  044524  042526  041440
030562  042117  051505  040440
030570  042522  040440  040526
```

```
        030576  046111  041101  042514
        030604     072
 2991   030605     200  020200  030040          .ASCII  <CRLF><CRLF>/  0  GENERATE BOOT PULSES (AT J2 PIN B AND J3 PIN 2)/
        030612  020040  042507  042516
        030620  040522  042524  041040
        030626  047517  020124  052520
        030634  051514  051505  024040
        030642  052101  045040  020062
        030650  044520  020116  020102
        030656  047101  020104  031512
        030664  050040  047111  031040
        030672     051
 2992   030673     200  020040  020061          .ASCII  <CRLF>/  1  TRANSMIT PG INTERRUPTS TO SELF ON IIST BUS/
        030700  052040  040522  051516
        030706  044515  020124  043520
        030714  044440  052116  051105
        030722  052522  052120  020123
        030730  047524  051440  046105
        030736  020106  047117  044440
        030744  051511  020124  052502
        030752     123
 2993   030753     200  020040  020062          .ASCII  <CRLF>/  2  TRANSMIT ST INTERRUPTS TO SELF ON IIST BUS/
        030760  052040  040522  051516
        030766  044515  020124  052123
        030774  044440  052116  051105
        031002  052522  052120  020123
        031010  047524  051440  046105
        031016  020106  047117  044440
        031024  051511  020124  052502
        031032     123
 2994   031033     200  020040  020063          .ASCII  <CRLF>/  3  TRANSMIT PG INTERRUPTS TO ALL IIST'S/
        031040  052040  040522  051516
        031046  044515  020124  043520
        031054  044440  052116  051105
        031062  052522  052120  020123
        031070  047524  040440  046114
        031076  044440  051511  023524
        031104     123
 2995   031105     200  020040  020064          .ASCII  <CRLF>/  4  TRANSMIT PG BOOTS TO ALL IIST'S/
        031112  052040  040522  051516
        031120  044515  020124  043520
        031126  041040  047517  051524
        031134  052040  020117  046101
        031142  020114  044511  052123
        031150  051447
 2996   031152  100200  047524  043440          .ASCIZ  <CRLF><CRLF>/TO GET OUT OF THE LOOP, TYPE ANY CHARACTER./<CRLF>
        031160  052105  047440  052125
        031166  047440  020106  044124
        031174  020105  047514  050117
        031202  020054  054524  042520
        031210  040440  054516  041440
        031216  040510  040522  052103
        031224  051105  100056     000
 2997           031232                          .EVEN
 2998
 2999   031232  005777  147710          SLOOP1: TST     @$TKB           ;FLUSH KEYBOARD
```

```
 3000  031236  104401  031244                 TYPE    ,65$            ;;TYPE ASCIZ STRING
  (1)  031242  000423                         BR      64$             ;;GET OVER THE ASCIZ
  (1)                                 ;;65$:  .ASCIZ  <CRLF>/ENTER CODE FOR DESIRED LOOP (0-4): /
  (1)  031312                         64$:
 3001  031312  104412                         RDOCT                   ;GET INPUT
 3002  031314  012600                         MOV     (SP)+,R0        ;GET VALUE
 3003  031316  020027  000004                 CMP     R0,#4           ;SEE IF LEGAL
 3004  031322  101426                         BLOS    1$              ;IF 0-4, THEN OK, GO DO IT.
 3005  031324  104401  031332                 TYPE    ,67$            ;;TYPE ASCIZ STRING
  (1)  031330  000422                         BR      66$             ;;GET OVER THE ASCIZ
  (1)                                 ;;67$:  .ASCIZ  /ILLEGAL CODE; MUST BE IN RANGE 0-4/
  (1)  031376                         66$:
 3006  031376  000715                         BR      SLOOP1          ;GO ASK AGAIN.
 3007
 3008  031400  006300                 1$:     ASL     R0              ;MAKE WORD INDEX.
 3009  031402  000170  031406                 JMP     @SLDISP(R0)     ;JUMP VIA DISPATCH TABLE.
 3010
 3011                                 ;DISPATCH TABLE
 3012  031406  031420                 SLDISP: SCBOOT  ;0 BOOT PULSES
 3013  031410  032000                         SCPISF  ;1 PG INTERRUPTS TO SELF
 3014  031412  031554                         SCSTSF  ;2 ST INTERRUPTS TO SELF
 3015  031414  032026                         SCPIAL  ;3 PG INTERRUPTS TO ALL
 3016  031416  032040                         SCPBAL  ;4 PG BOOTS TO ALL
 3017
 3018
 3019                                 ;BOOT PULSES (ASSUMES BOOT CABLE IS DISCONNECTED)
 3020
 3021  031420  004737  023450         SCBOOT: JSR     PC,CLEARX       ;INIT THE INTERFACE
 3022  031424  012777  000015  173300         MOV     #MTCE,@ACR      ;SELECT MTC
 3023  031432  012777  000002  173276         MOV     #MLEN,@MTC      ;SET MAINT. LOOPBACK
 3024  031440  005000                         CLR     R0              ;R0 CYCLES THE MASKS SO EACH RECEIVER OPERATES IN TURN.
 3025
 3026  031442  010001                 1$:     MOV     R0,R1           ;GET MASK CYCLE CONTROL
 3027  031444  006301                         ASL     R1              ;MAKE WORD INDEX
 3028  031446  042701  177770                 BIC     #177770,R1      ;ZAP JUNK
 3029  031452  016102  022002                 MOV     PGBTAB(R1),R2   ;GET BIT TO ENABLE
 3030  031456  005102                         COM     R2              ;SET ALL OTHER MASKS
 3031  031460  012777  000004  173244         MOV     #IMSKE,@ACR     ;SELECT IMSK REGISTER
 3032  031466  010277  173244                 MOV     R2,@IMSK        ;LOAD MASKS, LEAVING ONE CLEAR
 3033  031472  013701  024730                 MOV     SIDNUM,R1       ;GET SELF-ID
 3034  031476  006301                         ASL     R1              ;MAKE WORD INDEX
 3035  031500  012777  000000  173224         MOV     #PGTEE,@ACR     ;SELECT PGTE
 3036  031506  016177  022002  173222         MOV     PGBTAB(R1),@PGTE ;SET TO XMIT TO SELF
 3037  031514  012777  000001  173214         MOV     #GO,@PGCS       ;TRANSMIT
 3038  031522  012701  000040                 MOV     #40,R1          ;INIT STALL COUNTER
 3039  031526  013702  024732                 MOV     ACR,R2          ;GET ACR ADDRESS
 3040  031532  005712                 2$:     TST     (R2)            ;STALL
 3041  031534  005301                         DEC     R1
 3042  031536  001375                         BNE     2$
 3043  031540  005200                         INC     R0              ;WHEN DONE, CYCLE MASK
 3044  031542  105777  147376                 TSTB    @$TKS           ;SEE IF KBD ACTIVE
 3045  031546  100335                         BPL     1$      ;IF NO, KEEP LOOPING
 3046  031550  000137  031232                 JMP     SLOOP1          ;IF YES, EXIT LOOP
 3047
 3048
 3049                                 ;SEND REAL SANITY-TIMER INTERRUPTS TO SELF:
```

```
3050  031554  004737  023450          SCSTSF: JSR     PC,CLEARX       ;INIT THE INTERFACE
3051  031560  013700  024736          1$:     MOV     ADR,R0          ;GET ADR ADDRESS
3052  031564  114001                          MOVB    -(R0),R1        ;BUMP ADDRESS AND GET SELF-ID
3053  031566  006301                          ASL     R1              ;MAKE WORD INDEX
3054  031570  112740  000002                  MOVB    #STTEE,-(R0)    ;SELECT STTE REGISTER
3055  031574  016160  021772  000002          MOV     STITAB(R1),2(R0) ;SET ENABLE TO XMIT TO SELF
3056  031602  012760  000011  000002          MOV     #TMO+ENB,2(R0)  ;CLEAR TMO, ENABLE CLOCK FOR 1 COUNT.
3057  031610  012701  023420                  MOV     #10000.,R1      ;SET DELAY COUNTER
3058
3059  031614  105777  147324          2$:     TSTB    @$TKS           ;KEYBOARD ACTIVE?
3060  031620  100002                          BPL     .+6             ;IF NOT, SKIP
3061  031622  000137  031232                  JMP     SLOOP1          ;IF YES, EXIT LOOP
3062  031626  012710  000003                  MOV     #STCSE,(R0)     ;SELECT STCS
3063  031632  032760  000010  000002          BIT     #TMO,2(R0)      ;TMO SET?
3064  031640  001003                          BNE     3$              ;SKIP IF YES
3065  031642  005301                          DEC     R1              ;IF NOT, JUST WAIT
3066  031644  001363                          BNE     2$              ;BUT NOT TOO LONG
3067  031646  000742                          BR      SCSTSF          ;NO TIMEOUT -- GO TO INIT
3068
3069  031650  012701  000040          3$:     MOV     #40,R1          ;SET TO STALL FOR XMIT
3070
3071  031654  005301                  4$:     DEC     R1              ;STALL
3072  031656  001376                          BNE     4$
3073  031660  000737                          BR      1$              ;DO IT AGAIN
3074
3075
3076                                  ;GENERAL TRANSMIT ROUTINE:
3077  031662  000000                  XMITS:  .WORD   0       ;BIT TO TRANSMIT
3078
3079  031664  004737  023450          SCPXMT: JSR     PC,CLEARX       ;INIT THE INTERFACE
3080  031670  012777  000000  173034          MOV     #PGTEE,@ACR     ;SELECT THE ACR
3081  031676  013700  031662                  MOV     XMITS,R0        ;GET BITS TO TRANSMIT
3082  031702  004737  023506                  JSR     PC,PARCAL       ;CALCULATE PARITY
3083  031706  006301                          ASL     R1              ;POSITION PARITY BIT
3084  031710  052701  177774                  BIS     #177774,R1
3085  031714  005101                          COM     R1              ;PARITY & GO
3086  031716  012777  000000  173006  1$:     MOV     #PGTEE,@ACR     ;SELECT PGTE
3087  031724  013777  031662  173004          MOV     XMITS,@PGTE     ;LOAD XMITS
3088  031732  010177  173000                  MOV     R1,@PGCS        ;SEND
3089  031736  012702  000020                  MOV     #20,R2          ;LOOP COUNTER IN CASE OF HANG
3090  031742  105777  147176          2$:     TSTB    @$TKS           ;KEYBOARD ACTIVE?
3091  031746  100002                          BPL     3$              ;CONTINUE OF NOT
3092  031750  000137  031232                  JMP     SLOOP1          ;EXIT LOOP IF YES
3093  031754  012777  000001  172750  3$:     MOV     #PGCSE,@ACR
3094  031762  032777  004000  172746          BIT     #PRDY,@PGCS     ;READY?
3095  031770  001352                          BNE     1$              ;IF YES, DO AGAIN
3096  031772  005302                          DEC     R2              ;IF NOT, COUNT DOWN
3097  031774  001362                          BNE     2$              ;LOOP
3098  031776  000732                          BR      SCPXMT          ;NO PRDY
3099
3100
3101                                  ;PG INTERRUPTS TO SELF:
3102  032000  017701  172726          SCPISF: MOV     @ACR,R1         ;GET SELF-ID
3103  032004  000301                          SWAB    R1
3104  032006  006301                          ASL     R1
3105  032010  042701  177770                  BIC     #177770,R1
```

```
  3106  032014  016137  021772  031662          MOV     PGITAB(R1),XMITS ;GET BIT TO TRANSMIT
  3107  032022  000137  031664                  JMP     SCPXMT           ;GO DO IT
  3108
  3109                                  ;PG INTERRUPTS TO ALL:
  3110  032026  012737  000017  031662  SCPIAL: MOV     #17,XMITS        ;SET ALL INTR XMIT ENABLES
  3111  032034  000137  031664                  JMP     SCPXMT
  3112
  3113                                  ;PG BOOTS TO ALL:
  3114  032040  012737  007400  031662  SCPBAL: MOV     #7400,XMITS      ;SET ALL BOOT XMIT ENABLE
  3115  032046  000137  031664                  JMP     SCPXMT           ;GO TO MAIN LOOP
```

```
3117                                    .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)  032052                            $EOP:
 (1)  032052  000004                            SCOPE
 (1)  032054  005037  001102                    CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 (1)  032060  005037  001214                    CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 (1)  032064  005237  001236                    INC     $PASS           ;;INCREMENT THE PASS NUMBER
 (1)  032070  042737  100000  001236            BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 (1)  032076  005327                            DEC     (PC)+           ;;LOOP?
 (1)  032100  000001                    $EOPCT: .WORD   1
 (1)  032102  003022                            BGT     $DOAGN          ;;YES
 (1)  032104  012737                            MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 (1)  032106  000001                    $ENDCT: .WORD   1
 (1)  032110  032100                            $EOPCT
 (1)  032112  104401  032157                    TYPE    ,$ENDMG         ;;TYPE "END PASS #"
 (2)  032116  013746  001236                    MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 (2)  032122  104405                            TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 (1)  032124  104401  032154                    TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 (1)  032130  013700  000042            $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 (1)  032134  001405                            BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 (1)  032136  000005                            RESET                   ;;CLEAR THE WORLD
 (1)  032140  004710                    $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 (1)  032142  000240                            NOP                     ;;SAVE ROOM
 (1)  032144  000240                            NOP                     ;;FOR
 (1)  032146  000240                            NOP                     ;;ACT11
 (1)  032150                            $DOAGN:
 (1)  032150  000137                            JMP     @(PC)+          ;;RETURN
 (1)  032152  003562                    $RTNAD: .WORD   TST1
 (1)  032154     377     377     000    $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 (1)  032157     015  042412  042116    $ENDMG: .ASCIZ  <15><12>/END PASS #/
 (1)  032164  050040  051501  020123
 (1)  032172  000043
3118
3119                                    .SBTTL  SCOPE HANDLER ROUTINE
 (1)
 (2)                                    ;;**********************************************************************
 (1)                                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
 (1)                                    ;*AND LOAD THE TEST NUMBER($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)  032174                            $SCOPE:
 (1)  032174  104407                            CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
 (1)                                    ;;GO TO ERROR ROUTINE IF RETURN PC LESS THAN 1002
```

```
(1)                                     ;;OTHERWISE CONTINUE
(1)  032176  021627  001002                     CMP     (SP),#1002      ;;UNEXPECTED TRAP OR INTERRRUPT
(1)  032202  101002                             BHI     1$              ;;ARE TRAPPED HERE VIA IOT
(1)  032204  000137  032446                     JMP     $ERROR          ;;GO PROCESS UNEXPECTED TRAP
(1)  032210  032777  040000  146722  1$:       BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(1)  032216  001104                             BNE     $OVER           ;;YES IF SW14=1
(1)                                     ;#####START OF CODE FOR THE XOR TESTER#####
(1)  032220  000416                     $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                                                     ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(1)  032222  013746  000004                     MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  032226  012737  032246  000004             MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  032234  005737  177060                     TST     @#177060        ;;TIME OUT ON XOR?
(1)  032240  012637  000004                     MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  032244  000453                             BR      $SVLAD          ;;GO TO THE NEXT TEST
(1)  032246  022626                     5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(1)  032250  012637  000004                     MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  032254  000413                             BR      7$              ;;LOOP ON THE PRESENT TEST
(1)  032256                             6$:;#####END OF CODE FOR THE XOR TESTER#####
(1)  032256  105737  001103             2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(1)  032262  001421                             BEQ     3$              ;;BR IF NO
(1)  032264  123737  001115  001103             CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
(1)  032272  101015                             BHI     3$              ;;BR IF NO
(1)  032274  032777  001000  146636             BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
(1)  032302  001404                             BEQ     4$              ;;BR IF NO
(1)  032304  013737  001110  001106  7$:       MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
(1)  032312  000446                             BR      $OVER
(1)  032314  105037  001103             4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  032320  005037  001214                     CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  032324  000415                             BR      1$              ;;ESCAPE TO THE NEXT TEST
(1)  032326  032777  004000  146604  3$:       BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(1)  032334  001011                             BNE     1$              ;;BR IF YES
(1)  032336  005737  001236                     TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  032342  001406                             BEQ     1$              ;;      INHIBIT ITERATIONS
(1)  032344  005237  001104                     INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  032350  023737  001214  001104             CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  032356  002024                             BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(1)  032360  012737  000001  001104  1$:       MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  032366  013737  032444  001214             MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  032374  105237  001102             $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(1)  032400  113737  001102  001234             MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
(1)  032406  011637  001106                     MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  032412  011637  001110                     MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(1)  032416  005037  001216                     CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  032422  112737  000001  001115             MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)  032430  013777  001102  146504  $OVER:    MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1)  032436  013716  001106                     MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  032442  000002                             RTI                     ;;FIXES PS
(1)  032444  000001                     $MXCNT: 1                       ;;MAX. NUMBER OF ITERATIONS
3120                                    .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)  032446                             $ERROR:
(1)  032446  104407                             CKSWR                           ;;TEST FOR CHANGE IN SOFT-SWR
(1)  032450  105237  001103             7$:     INCB    $ERFLG                  ;;SET THE ERROR FLAG
(1)  032454  001775                             BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
(1)  032456  013777  001102  146456             MOV     $TSTNM,@DISPLAY         ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  032464  032777  002000  146446             BIT     #BIT10,@SWR             ;;BELL ON ERROR?
(1)  032472  001402                             BEQ     1$                      ;;NO - SKIP
(1)  032474  104401  001220                     TYPE    ,$BELL                  ;;RING BELL
(1)  032500  005237  001112             1$:     INC     $ERTTL                  ;;COUNT THE NUMBER OF ERRORS
(1)  032504  011637  001116                     MOV     (SP),$ERRPC             ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  032510  162737  000002  001116             SUB     #2,$ERRPC
(1)  032516  117737  146374  001114             MOVB    @$ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  032524  032777  020000  146406             BIT     #BIT13,@SWR             ;;SKIP TYPEOUT IF SET
(1)  032532  001055                             BNE     20$                     ;;SKIP TYPEOUTS
(1)  032534  021627  001002                     CMP     (SP),#1002              ;;IF RETURN PC LESS THAN 1002
(1)  032540  101046                             BHI     12$                     ;;ERROR IS ILLEGAL TRAP
(1)                                     ;;PROCESS UNEXPECTED TRAP OR INTERRUPT
(1)  032542  016637  000004  001116             MOV     4(SP),$ERRPC            ;;GET PC AT TIME OF FALSE TRAP
(1)  032550  162737  000002  001116             SUB     #2,$ERRPC               ;;ADJUST PC
(1)  032556  104401  032622                     TYPE    ,10$                    ;;TYPE HEADER
(2)  032562  013746  001116                     MOV     $ERRPC,-(SP)            ;;SAVE $ERRPC FOR TYPEOUT
(2)  032566  104402                             TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  032570  104401  032630                     TYPE    ,11$
(1)  032574  162716  000004                     SUB     #4,(SP)                 ;;GET FALSE TRAP VECTOR ADDR
(1)  032600  011637  001116                     MOV     (SP),$ERRPC
(2)  032604  013746  001116                     MOV     $ERRPC,-(SP)            ;;SAVE $ERRPC FOR TYPEOUT
(2)  032610  104402                             TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  032612  104401  001225                     TYPE    ,$CRLF
(1)  032616  022626                             CMP     (SP)+,(SP)+             ;;POP FALSE TRAP VECTOR PC&ADDR
(1)  032620  000422                             BR      20$
(1)  032622  050200  036503  000040  10$:      .ASCIZ  <200>'PC= '
(1)  032630  020040  047125  054105  11$:      .ASCIZ  '  UNEXPECTED TRAP TO '
(1)  032636  042520  052103  042105
(1)  032644  052040  040522  020120
(1)  032652  047524  000040
(1)                                             .EVEN
(1)  032656                             12$:
(1)  032656  004737  032770                     JSR     PC,$ERRTYP              ;;GO TO USER ERROR ROUTINE
(1)  032662  104401  001225                     TYPE    ,$CRLF
(1)  032666                             20$:
(1)  032666  122737  000001  001250             CMPB    #APTENV,$ENV            ;;RUNNING IN APT MODE
(1)  032674  001007                             BNE     2$                      ;;NO,SKIP APT ERROR REPORT
(1)  032676  113737  001114  032710             MOVB    $ITEMB,21$              ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  032704  004737  035732                     JSR     PC,$ATY4                ;;REPORT FATAL ERROR TO APT
(1)  032710     000                     21$:    .BYTE   0
(1)  032711     000                             .BYTE   0
(1)  032712  000777                     22$:    BR      22$                     ;;APT ERROR LOOP
(1)  032714  005777  146220             2$:     TST     @SWR                    ;;HALT ON ERROR
(1)  032720  100002                             BPL     3$                      ;;SKIP IF CONTINUE
(1)  032722  000000                             HALT                            ;;HALT ON ERROR!
```

```
(1)  032724  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
(1)  032726  032777  001000  146204  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
(1)  032734  001402                          BEQ     4$              ;;BR IF NO
(1)  032736  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(1)  032742  005737  001216          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(1)  032746  001402                          BEQ     5$              ;;BR IF NONE
(1)  032750  013716  001216                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  032754                          5$:
(1)  032754  022737  032140  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
(1)  032762  001001                          BNE     6$              ;;BRANCH IF NO
(1)  032764  000000                          HALT                    ;;YES
(1)  032766                          6$:
(1)  032766  000002                          RTI                     ;;RETURN
3121                                 .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
(1)
(2)                                  ;;**************************************************************
(1)                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($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)  032770                          $ERRTYP:
(1)  032770  104401  001225                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  032774  010046                          MOV     R0,-(SP)        ;;SAVE R0
(1)  032776  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  033000  153700  001114                  BISB    @#$ITEMB,R0
(1)  033004  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                  ;;TYPE THE PC OF THE ERROR
(2)  033006  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                  ;;ERROR ADDRESS
(2)  033012  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  033014  000445                          BR      10$             ;;GET OUT
(1)  033016  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(1)  033020  006300                          ASL     R0              ;;       WORK FOR THE ERROR TABLE
(1)  033022  006300                          ASL     R0
(1)  033024  006300                          ASL     R0
(1)  033026  062700  001354                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(1)  033032  012037  033042                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(1)  033036  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(1)  033040  104401                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
(1)  033042  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  033044  104401  001225                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  033050  012037  033060          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
(1)  033054  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
(1)  033056  104401                          TYPE                    ;;TYPE THE "DATA HEADER"
(1)  033060  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
(1)  033062  104401  001225                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  033066  010146                  5$:     MOV     R1,-(SP)        ;;SAVE R1
(1)  033070  012001                          MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
(1)  033072  001415                          BEQ     9$              ;;BR IF NO DATA TO BE TYPED
(1)  033074  012000                          MOV     (R0)+,R0        ;;PICKUP "DATA FORMAT" POINTER
(1)  033076  105720                  6$:     TSTB    (R0)+           ;;"OCTAL" OR "DECIMAL"
(1)  033100  001003                          BNE     7$              ;;BR IF DECIMAL
(2)  033102  013146                          MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
(2)  033104  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  033106  000402                          BR      8$
(1)  033110                          7$:
```

```
(2)  033110  013146                        MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
(2)  033112  104405                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  033114  005711                8$:     TST     (R1)            ;;IS THERE ANOTHER NUMBER?
(1)  033116  001403                        BEQ     9$              ;;BR IF NO
(1)  033120  104401  033140               TYPE    ,11$            ;;TYPE TWO(2) SPACES
(1)  033124  000764                        BR      6$              ;;LOOP
(1)
(1)  033126  012601                9$:     MOV     (SP)+,R1        ;;RESTORE R1
(1)  033130  012600                10$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  033132  104401  001225               TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  033136  000207                        RTS     PC              ;;RETURN
(1)  033140  020040      000       11$:    .ASCIZ  /  /            ;;TWO(2) SPACES
(1)          033144                        .EVEN
3122                               .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)  033144  010046                $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
(1)  033146  010146                        MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
(1)  033150  010246                        MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
(1)  033152  010346                        MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
(1)  033154  013746  000004               MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
(1)  033160  013746  000006               MOV     @#ERRVEC+2,-(SP)
(1)  033164  010600                        MOV     SP,R0           ;;SAVE THE STACK POINTER
(1)                                ;;SET THE ERRVEC PS TO THE PRESENT PS
(2)  033166  104400                        TRAP                    ;;PUSH OLD PSW AND PC ON STACK
(2)  033170  012637  000006               MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
(1)  033174  012701  003776               MOV     #3776,R1        ;;SETUP ADDRESS
(1)  033200  105727                        TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
(1)  033202  000200                $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
(1)  033204  100062                        BPL     $CORE           ;;BR IF NO
(1)  033206  012737  033344  000004        MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
(1)  033214  005737  177572               TST     @#SR0           ;;KT11 ARE YOU THERE?
(1)  033220  052737  100000  033202        BIS     #100000,$KT11   ;;YES--SET KT11 KEY
(1)  033226  005046                        CLR     -(SP)           ;;INITIALIZE FOR "PAR" LOADING
(1)  033230  012702  172340               MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST "PAR"
(1)  033234  012703  000010               MOV     #^D8,R3         ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
(1)  033240  012762  077406  177740  1$:  MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
(1)  033246  011622                        MOV     (SP),(R2)+      ;;LOAD "PAR"
(1)  033250  062716  000200               ADD     #200,(SP)       ;;UPDATE FOR NEXT "PAR"
(1)  033254  077307                        SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
(1)  033256  012742  177600               MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
(1)  033262  005042                        CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
(1)  033264  012737  033302  000004        MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
```

```
    (1)  033272  012737  000020  172516          MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
    (1)  033300  000401                          BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
    (1)  033302  022626                  2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
    (1)  033304  005237  177572          3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
    (1)  033310  012737  033334  000004          MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
    (1)  033316  005737  143776          4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
    (1)  033322  062712  000040                  ADD     #40,(R2)        ;;MAKE A 1K STEP
    (1)  033326  023712  172356                  CMP     @#KIPAR7,(R2)   ;;LAST ONE?
    (1)  033332  101371                          BHI     4$              ;;NO--TRY IT
    (1)  033334  011202                  $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
    (1)  033336  005037  177572                  CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
    (1)  033342  000421                          BR      $SIZEX
    (1)  033344  042737  100000  033202  $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
    (1)  033352  012737  033402  000004  $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
    (1)  033360  005002                          CLR     R2              ;;SET UP BANK
    (1)  033362  062701  004000          1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
    (1)  033366  062702  000040                  ADD     #40,R2          ;;1K STEP
    (1)  033372  005711                          TST     (R1)            ;;TRAP ON TIME OUT
    (1)  033374  022701  177776                  CMP     #177776,R1      ;;LAST ONE
    (1)  033400  001370                          BNE     1$              ;;NO--TRY AGAIN
    (1)  033402  162701  004000          $CROUT: SUB     #4000,R1
    (1)  033406  162702  000040          $SIZEX: SUB     #40,R2          ;;DROP BACK
    (1)  033412  010006                          MOV     R0,SP           ;;RESTORE THE STACK
    (1)  033414  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
    (1)  033420  012637  000004                  MOV     (SP)+,@#ERRVEC
    (1)  033424  010137  033446                  MOV     R1,$LSTAD       ;;LAST ADDRESS
    (1)  033430  010237  033450                  MOV     R2,$LSTBK       ;;LAST BANK
    (1)  033434  012603                          MOV     (SP)+,R3        ;;RESTORE R3
    (1)  033436  012602                          MOV     (SP)+,R2        ;;RESTORE R2
    (1)  033440  012601                          MOV     (SP)+,R1        ;;RESTORE R1
    (1)  033442  012600                          MOV     (SP)+,R0        ;;RESTORE R0
    (1)  033444  000207                          RTS     PC
    (1)  033446  000000          $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
    (1)  033450  000000          $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
3123                             .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)  033452  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
    (1)  033454  016666  000004  000002  MOV     4(SP),2(SP)     ;;INPUT NUMBER
    (3)  033462  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
    (3)  033464  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
    (3)  033466  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
    (1)  033470  104411          1$:     RDLIN                   ;;READ AN ASCIZ LINE
    (1)  033472  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
    (1)  033474  010037  033600          MOV     R0,5$           ;;AND SAVE IT
```

```
(1)  033500  005001                        CLR     R1              ;;CLEAR DATA WORD
(1)  033502  005002                        CLR     R2
(1)  033504  112046                2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
(1)  033506  001420                        BEQ     3$              ;;IF ZERO GET OUT
(1)  033510  122716  000060                CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
(1)  033514  003026                        BGT     4$              ;;IS AN OCTAL DIGIT
(1)  033516  122716  000067                CMPB    #'7,(SP)
(1)  033522  002423                        BLT     4$
(1)  033524  006301                        ASL     R1              ;;*2
(1)  033526  006102                        ROL     R2
(1)  033530  006301                        ASL     R1              ;;*4
(1)  033532  006102                        ROL     R2
(1)  033534  006301                        ASL     R1              ;;*8
(1)  033536  006102                        ROL     R2
(1)  033540  042716  177770                BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
(1)  033544  062601                        ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
(1)  033546  000756                        BR      2$              ;;LOOP
(1)  033550  005726                3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
(1)  033552  010166  000012                MOV     R1,12(SP)       ;;SAVE THE RESULT
(1)  033556  010237  033610                MOV     R2,$HIOCT
(3)  033562  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  033564  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  033566  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  033570  000002                        RTI                     ;;RETURN
(1)  033572  005726                4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
(1)  033574  105010                        CLRB    (R0)            ;;SET A TERMINATOR
(1)  033576  104401                        TYPE                    ;;TYPE UP THRU THE BAD CHAR.
(1)  033600  000000                5$:     .WORD   0
(1)  033602  104401  001224                TYPE    ,$QUES          ;;"?" "CR" & "LF"
(1)  033606  000730                        BR      1$              ;;TRY AGAIN
(1)  033610  000000                $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
3124                                       .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)  033612  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
(1)  033620  001074                        BNE     15$             ;;BRANCH IF NO
(1)  033622  105777  145316                TSTB    @$TKS           ;;CHAR THERE?
(1)  033626  100071                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
(1)  033630  117746  145312                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
(1)  033634  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
(1)  033640  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
(1)  033644  001062                        BNE     15$             ;;NO, RETURN TO USER
(1)  033646  123727  001134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
(1)  033654  001456                        BEQ     15$             ;;BRANCH IF YES
(1)
(1)  033656  104401  034471                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
(1)  033662  104401  034476        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
(2)  033666  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
(2)  033672  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
(1)  033674  104401  034507              TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
(1)  033700  005046              19$:    CLR     -(SP)           ;;CLEAR COUNTER
(1)  033702  005046                      CLR     -(SP)           ;;THE NEW SWR
(1)  033704  105777  145234      7$:     TSTB    @$TKS           ;;CHAR THERE?
(1)  033710  100375                      BPL     7$              ;;IF NOT TRY AGAIN
(1)
(1)  033712  117746  145230              MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
(1)  033716  042716  177600              BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)
(1)
(1)
(1)  033722  021627  000025      9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
(1)  033726  001005                      BNE     10$             ;;BRANCH IF NOT
(1)  033730  104401  034464              TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
(1)  033734  062706  000006      20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
(1)  033740  000757                      BR      19$             ;;LET'S TRY IT AGAIN
(1)
(1)
(1)  033742  021627  000015      10$:    CMP     (SP),#15        ;;IS IT A <CR>?
(1)  033746  001022                      BNE     16$             ;;BRANCH IF NO
(1)  033750  005766  000004              TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
(1)  033754  001403                      BEQ     11$             ;;BRANCH IF YES
(1)  033756  016677  000002  145154      MOV     2(SP),@SWR      ;;SAVE NEW SWR
(1)  033764  062706  000006      11$:    ADD     #6,SP           ;;CLEAR UP STACK
(1)  033770  104401  001225      14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
(1)  033774  123727  001135  000001      CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
(1)  034002  001003                      BNE     15$             ;;BRANCH IF NOT
(1)  034004  012777  000100  145132      MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  034012  000002              15$:    RTI                     ;;RETURN
(1)  034014  004737  034732      16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  034020  021627  000060              CMP     (SP),#60        ;;CHAR < 0?
(1)  034024  002420                      BLT     18$             ;;BRANCH IF YES
(1)  034026  021627  000067              CMP     (SP),#67        ;;CHAR > 7?
(1)  034032  003015                      BGT     18$             ;;BRANCH IF YES
(1)  034034  042726  000060              BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  034040  005766  000002              TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  034044  001403                      BEQ     17$             ;;BRANCH IF YES
(1)  034046  006316                      ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  034050  006316                      ASL     (SP)            ;;    CHAR OVER TO MAKE
(1)  034052  006316                      ASL     (SP)            ;;    ROOM FOR NEW ONE.
(1)  034054  005266  000002      17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  034060  056616  177776              BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  034064  000707                      BR      7$              ;;GET THE NEXT ONE
(1)  034066  104401  001224      18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  034072  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)  034074  011646                    $RDCHR: MOV     (SP),-(SP)        ;;PUSH DOWN THE PC
(1)  034076  016666  000004  000002            MOV     4(SP),2(SP)       ;;SAVE THE PS
(1)  034104  105777  145034            1$:     TSTB    @$TKS             ;;WAIT FOR
(1)  034110  100375                            BPL     1$                ;;A CHARACTER
(1)  034112  117766  145030  000004            MOVB    @$TKB,4(SP)       ;;READ THE TTY
(1)  034120  042766  177600  000004            BIC     #^C<177>,4(SP)    ;;GET RID OF JUNK IF ANY
(1)  034126  026627  000004  000023            CMP     4(SP),#23         ;;IS IT A CONTROL-S?
(1)  034134  001013                            BNE     3$                ;;BRANCH IF NO
(1)  034136  105777  145002            2$:     TSTB    @$TKS             ;;WAIT FOR A CHARACTER
(1)  034142  100375                            BPL     2$                ;;LOOP UNTIL ITS THERE
(1)  034144  117746  144776                    MOVB    @$TKB,-(SP)       ;;GET CHARACTER
(1)  034150  042716  177600                    BIC     #^C177,(SP)       ;;MAKE IT 7-BIT ASCII
(1)  034154  022627  000021                    CMP     (SP)+,#21         ;;IS IT A CONTROL-Q?
(1)  034160  001366                            BNE     2$                ;;IF NOT DISCARD IT
(1)  034162  000750                            BR      1$                ;;YES, RESUME
(1)  034164  026627  000004  000140    3$:     CMP     4(SP),#140        ;;IS IT UPPER CASE?
(1)  034172  002407                            BLT     4$                ;;BRANCH IF YES
(1)  034174  026627  000004  000175            CMP     4(SP),#175        ;;IS IT A SPECIAL CHAR?
(1)  034202  003003                            BGT     4$                ;;BRANCH IF YES
(1)  034204  042766  000040  000004            BIC     #40,4(SP)         ;;MAKE IT UPPER CASE
(1)  034212  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)  034214  010346                    $RDLIN: MOV     R3,-(SP)          ;;SAVE R3
(1)  034216  005046                            CLR     -(SP)             ;;CLEAR THE RUBOUT KEY
(1)  034220  012703  034450            1$:     MOV     #$TTYIN,R3        ;;GET ADDRESS
(1)  034224  022703  034464            2$:     CMP     #$TTYIN+12.,R3    ;;BUFFER FULL?
(1)  034230  101456                            BLOS    4$                ;;BR IF YES
(1)  034232  104410                            RDCHR                     ;;GO READ ONE CHARACTER FROM THE TTY
(1)  034234  112613                            MOVB    (SP)+,(R3)        ;;GET CHARACTER
(1)  034236  122713  000177            10$:    CMPB    #177,(R3)         ;;IS IT A RUBOUT
(1)  034242  001022                            BNE     5$                ;;BR IF NO
(1)  034244  005716                            TST     (SP)              ;;IS THIS THE FIRST RUBOUT?
(1)  034246  001007                            BNE     6$                ;;BR IF NO
(1)  034250  112737  000134  034446            MOVB    #'\,9$            ;;TYPE A BACK SLASH
(1)  034256  104401  034446                    TYPE    ,9$
(1)  034262  012716  177777                    MOV     #-1,(SP)          ;;SET THE RUBOUT KEY
(1)  034266  005303                    6$:     DEC     R3                ;;BACKUP BY ONE
(1)  034270  020327  034450                    CMP     R3,#$TTYIN        ;;STACK EMPTY?
(1)  034274  103434                            BLO     4$                ;;BR IF YES
(1)  034276  111337  034446                    MOVB    (R3),9$           ;;SETUP TO TYPEOUT THE DELETED CHAR.
(1)  034302  104401  034446                    TYPE    ,9$               ;;GO TYPE
(1)  034306  000746                            BR      2$                ;;GO READ ANOTHER CHAR.
(1)  034310  005716                    5$:     TST     (SP)              ;;RUBOUT KEY SET?
(1)  034312  001406                            BEQ     7$                ;;BR IF NO
(1)  034314  112737  000134  034446            MOVB    #'\,9$            ;;TYPE A BACK SLASH
(1)  034322  104401  034446                    TYPE    ,9$
(1)  034326  005016                            CLR     (SP)              ;;CLEAR THE RUBOUT KEY
(1)  034330  122713  000025            7$:     CMPB    #25,(R3)          ;;IS CHARACTER A CTRL U?
(1)  034334  001003                            BNE     8$                ;;BR IF NO
(1)  034336  104401  034464                    TYPE    ,$CNTLU           ;;TYPE A CONTROL "U"
```

```
(1)  034342  000726                          BR      1$              ;;GO START OVER
(1)  034344  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
(1)  034350  001011                          BNE     3$              ;;BRANCH IF NO
(1)  034352  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
(1)  034354  104401  001225                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
(1)  034360  104401  034450                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
(1)  034364  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
(1)  034366  104401  001224          4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  034372  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  034374  111337  034446          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  034400  104401  034446                  TYPE    ,9$
(1)  034404  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  034410  001305                          BNE     2$              ;;LOOP IF NOT RETURN
(1)  034412  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
(1)  034416  104401  001226                  TYPE    ,$LF            ;;TYPE A LINE FEED
(1)  034422  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
(1)  034424  012603                          MOV     (SP)+,R3        ;;RESTORE R3
(1)  034426  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  034430  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  034436  012766  034450  000004          MOV     #$TTYIN,4(SP)
(1)  034444  000002                          RTI                     ;;RETURN
(1)  034446     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  034447     000                  .BYTE   0                       ;;TERMINATOR
(1)  034450  000014          $TTYIN: .BLKB   12.                     ;;RESERVE 12. BYTES FOR TTY INPUT
(1)  034464  052536  005015  000    $CNTLU: .ASCIZ  /^U/<15><12>     ;;CONTROL "U"
(1)  034471     136  006507  000012 $CNTLG: .ASCIZ  /^G/<15><12>     ;;CONTROL "G"
(1)  034476  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  034504  020075     000
(1)  034507     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
(1)  034514  036440  000040
3125                                  .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)  034520  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 (1)  034524  100002                          BPL     1$              ;;BR IF YES
 (1)  034526  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
 (1)  034530  000430                          BR      3$              ;;LEAVE
 (1)  034532  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
 (1)  034534  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 (1)  034540  122737  000001  001250          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 (1)  034546  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
 (1)  034550  132737  000100  001251          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
```

```
(1)  034556  001405                        BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(1)  034560  010037  034570                MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(1)  034564  004737  035722                JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(1)  034570  000000              61$:      .WORD   0               ;;MESSAGE ADDRESS
(1)  034572  132737  000040  001251  62$:  BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(1)  034600  001003                        BNE     60$             ;;YES,SKIP TYPE OUT
(1)  034602  112046              2$:       MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  034604  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  034606  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  034610  012600              60$:      MOV     (SP)+,R0        ;;RESTORE R0
(1)  034612  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  034616  000002                        RTI                     ;;RETURN
(1)  034620  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  034624  001430                        BEQ     8$
(1)  034626  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)  034632  001006                        BNE     5$
(1)  034634  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  034636  104401                        TYPE                    ;;TYPE A CR AND LF
(1)  034640  001225                        $CRLF
(1)  034642  105037  034776                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)  034646  000755                        BR      2$              ;;GET NEXT CHARACTER
(1)  034650  004737  034732        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  034654  123726  001156        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  034660  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  034662  013746  001154                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                                                 ;;AND THE NULL CHAR.
(1)  034666  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  034672  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  034674  004737  034732                JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  034700  105337  034776                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  034704  000770                        BR      7$              ;;LOOP
(1)
(1)                                 ;HORIZONTAL TAB PROCESSOR
(1)
(1)  034706  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  034712  004737  034732        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  034716  132737  000007  034776        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  034724  001372                        BNE     9$              ;;TAB STOP
(1)  034726  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
(1)  034730  000724                        BR      2$              ;;GET NEXT CHARACTER
(1)  034732  105777  144212        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  034736  100375                        BPL     $TYPEC
(1)  034740  116677  000002  144204        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  034746  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  034754  001003                        BNE     1$              ;;BRANCH IF NO
(1)  034756  105037  034776                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  034762  000406                        BR      $TYPEX          ;;EXIT
(1)  034764  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  034772  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
(1)  034774  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  034776  000000              $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(1)  035000  000207              $TYPEX: RTS     PC
(1)
3126                                .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)  035002  017646  000000         $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  035006  116637  000001  035225         MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  035014  112637  035227                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  035020  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  035024  000406                          BR      $TYPON
(1)  035026  112737  000001  035225 $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
(1)  035034  112737  000006  035227         MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  035042  112737  000005  035224 $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(1)  035050  010346                          MOV     R3,-(SP)        ;;SAVE R3
(1)  035052  010446                          MOV     R4,-(SP)        ;;SAVE R4
(1)  035054  010546                          MOV     R5,-(SP)        ;;SAVE R5
(1)  035056  113704  035227                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  035062  005404                          NEG     R4
(1)  035064  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  035070  110437  035226                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  035074  113704  035225                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  035100  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  035104  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  035106  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  035110  000404                          BR      3$              ;;GO DO MSB
(1)  035112  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
(1)  035114  006105                          ROL     R5
(1)  035116  006105                          ROL     R5
(1)  035120  010503                          MOV     R5,R3
(1)  035122  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  035124  105337  035226                  DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  035130  100016                          BPL     7$              ;;BR IF NO
(1)  035132  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
(1)  035136  001002                          BNE     4$              ;;TEST FOR 0
(1)  035140  005704                          TST     R4              ;;SUPPRESS THIS 0?
(1)  035142  001403                          BEQ     5$              ;;BR IF YES
(1)  035144  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  035146  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  035152  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
```

```
(1)  035156  110337  035222                    MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  035162  104401  035222                    TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  035166  105337  035224          7$:       DECB    $OCNT           ;;COUNT BY 1
(1)  035172  003347                            BGT     2$              ;;BR IF MORE TO DO
(1)  035174  002402                            BLT     6$              ;;BR IF DONE
(1)  035176  005204                            INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  035200  000744                            BR      2$              ;;GO DO THE LAST DIGIT
(1)  035202  012605                    6$:     MOV     (SP)+,R5        ;;RESTORE R5
(1)  035204  012604                            MOV     (SP)+,R4        ;;RESTORE R4
(1)  035206  012603                            MOV     (SP)+,R3        ;;RESTORE R3
(1)  035210  016666  000002  000004            MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  035216  012616                            MOV     (SP)+,(SP)
(1)  035220  000002                            RTI                     ;;RETURN
(1)  035222     000                    8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  035223     000                            .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  035224     000                    $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(1)  035225     000                    $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(1)  035226  000000                    $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
3127                                   .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)  035230                            $TYPDS:
(3)  035230  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  035232  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  035234  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  035236  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  035240  010546                            MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  035242  012746  020200                    MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)  035246  016605  000020                    MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)  035252  100004                            BPL     1$              ;;BR IF INPUT IS POS.
(1)  035254  005405                            NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)  035256  112766  000055  000001            MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  035264  005000                    1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  035266  012703  035444                    MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  035272  112723  000040                    MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  035276  005002                    2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  035300  016001  035434                    MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  035304  160105                    3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  035306  002402                            BLT     4$              ;;BR IF DONE
(1)  035310  005202                            INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  035312  000774                            BR      3$
(1)  035314  060105                    4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  035316  005702                            TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  035320  001002                            BNE     5$              ;;FALL THROUGH IF 0
(1)  035322  105716                            TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)  035324  100407                            BMI     7$              ;;BR IF YES
(1)  035326  106316                    5$:     ASLB    (SP)            ;;MSD?
```

```
(1)  035330  103003                        BCC     6$                  ;;BR IF NO
(1)  035332  116663  000001  177777        MOVB    1(SP),-1(R3)        ;;YES--SET THE SIGN
(1)  035340  052702  000060          6$:    BIS     #'0,R2              ;;MAKE THE BCD DIGIT ASCII
(1)  035344  052702  000040          7$:    BIS     #' ,R2              ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  035350  110223                        MOVB    R2,(R3)+            ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  035352  005720                        TST     (R0)+               ;;JUST INCREMENTING
(1)  035354  020027  000010                CMP     R0,#10              ;;CHECK THE TABLE INDEX
(1)  035360  002746                        BLT     2$                  ;;GO DO THE NEXT DIGIT
(1)  035362  003002                        BGT     8$                  ;;GO TO EXIT
(1)  035364  010502                        MOV     R5,R2               ;;GET THE LSD
(1)  035366  000764                        BR      6$                  ;;GO CHANGE TO ASCII
(1)  035370  105726                  8$:   TSTB    (SP)+               ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  035372  100003                        BPL     9$                  ;;BR IF NO
(1)  035374  116663  177777  177776        MOVB    -1(SP),-2(R3)       ;;YES--SET THE SIGN FOR TYPING
(1)  035402  105013                  9$:   CLRB    (R3)                ;;SET THE TERMINATOR
(3)  035404  012605                        MOV     (SP)+,R5            ;;POP STACK INTO R5
(3)  035406  012603                        MOV     (SP)+,R3            ;;POP STACK INTO R3
(3)  035410  012602                        MOV     (SP)+,R2            ;;POP STACK INTO R2
(3)  035412  012601                        MOV     (SP)+,R1            ;;POP STACK INTO R1
(3)  035414  012600                        MOV     (SP)+,R0            ;;POP STACK INTO R0
(1)  035416  104401  035444                TYPE    ,$DBLK              ;;NOW TYPE THE NUMBER
(1)  035422  016666  000002  000004        MOV     2(SP),4(SP)         ;;ADJUST THE STACK
(1)  035430  012616                        MOV     (SP)+,(SP)
(1)  035432  000002                        RTI                         ;;RETURN TO USER
(1)  035434  023420                $DTBL:   10000.
(1)  035436  001750                         1000.
(1)  035440  000144                          100.
(1)  035442  000012                           10.
(1)  035444  000004                $DBLK:   .BLKW   4
3128                                       .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)  035454  010046                $TRAP:   MOV     R0,-(SP)            ;;SAVE R0
(1)  035456  016600  000002                 MOV     2(SP),R0            ;;GET TRAP ADDRESS
(1)  035462  005740                         TST     -(R0)               ;;BACKUP BY 2
(1)  035464  111000                         MOVB    (R0),R0             ;;GET RIGHT BYTE OF TRAP
(1)  035466  006300                         ASL     R0                  ;;POSITION FOR INDEXING
(1)  035470  016000  035510                 MOV     $TRPAD(R0),R0       ;;INDEX TO TABLE
(1)  035474  000200                         RTS     R0                  ;;GO TO ROUTINE
(1)
(1)
(1)                                        ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
(1)
(1)  035476  011646                $TRAP2:  MOV     (SP),-(SP)          ;;MOVE THE PC DOWN
(1)  035500  016666  000004  000002         MOV     4(SP),2(SP)         ;;MOVE THE PSW DOWN
(1)  035506  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)  035510  035476                     $TRPAD: .WORD   $TRAP2
(3)  035512  034520                             $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
(3)  035514  035026                             $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(3)  035516  035002                             $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(3)  035520  035042                             $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(3)  035522  035230                             $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
(1)
(3)  035524  033662                             $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
(1)
(3)  035526  033612                             $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
(3)  035530  034074                             $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
(3)  035532  034214                             $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
(3)  035534  033452                             $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
3129                                    .SBTTL  POWER DOWN AND UP ROUTINES
(1)
(2)                                     ;;**************************************************************
(1)                                     ;POWER DOWN ROUTINE
(1)  035536  012737  035676  000024     $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(1)  035544  012737  000340  000026             MOV     #340,@#PWRVEC+2 ;;PRIO:7
(3)  035552  010046                             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  035554  010146                             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  035556  010246                             MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  035560  010346                             MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  035562  010446                             MOV     R4,-(SP)        ;;PUSH R4 ON STACK
(3)  035564  010546                             MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(3)  035566  017746  143346                     MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
(1)  035572  010637  035702                     MOV     SP,$SAVR6       ;;SAVE SP
(1)  035576  012737  035610  000024             MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
(1)  035604  000000                             HALT
(1)  035606  000776                             BR      .-2             ;;HANG UP
(1)
(2)                                     ;;**************************************************************
(1)                                     ;POWER UP ROUTINE
(1)  035610  012737  035676  000024     $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1)  035616  013706  035702                     MOV     $SAVR6,SP       ;;GET SP
(1)  035622  005037  035702                     CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
(1)  035626  005237  035702             1$:     INC     $SAVR6          ;;WAIT FOR THE INC
(1)  035632  001375                             BNE     1$              ;;OF  WORD
(3)  035634  012677  143300                     MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
(3)  035640  012605                             MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  035642  012604                             MOV     (SP)+,R4        ;;POP STACK INTO R4
(3)  035644  012603                             MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  035646  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  035650  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  035652  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  035654  012737  035536  000024             MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1)  035662  012737  000340  000026             MOV     #340,@#PWRVEC+2 ;;PRIO:7
(1)  035670  104401                             TYPE                    ;REPORT THE POWER FAILURE
(1)  035672  035704                     $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
(1)  035674  000002                             RTI
(1)  035676  000000                     $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
(1)  035700  000776                             BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
(1)  035702  000000                     $SAVR6: 0                       ;;PUT THE SP HERE
```

```
(1)  035704  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
(1)  035712  000122
(1)                                          .EVEN
3130                                 .SBTTL  APT COMMUNICATIONS ROUTINE
(1)
(2)                                  ;;**********************************************************************
(1)  035714  112737  000001  036160  $ATY1:  MOVB    #1,$FFLG          ;;TO REPORT FATAL ERROR
(1)  035722  112737  000001  036156  $ATY3:  MOVB    #1,$MFLG          ;;TO TYPE A MESSAGE
(1)  035730  000403                          BR      $ATYC
(1)  035732  112737  000001  036160  $ATY4:  MOVB    #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
(1)  035740                          $ATYC:
(3)  035740  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
(3)  035742  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
(1)  035744  105737  036156                  TSTB    $MFLG             ;;SHOULD TYPE A MESSAGE?
(1)  035750  001450                          BEQ     5$                ;;IF NOT:  BR
(1)  035752  122737  000001  001250          CMPB    #APTENV,$ENV      ;;OPERATING UNDER APT?
(1)  035760  001031                          BNE     3$                ;;IF NOT:  BR
(1)  035762  132737  000100  001251          BITB    #APTSPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
(1)  035770  001425                          BEQ     3$                ;;IF NOT:  BR
(1)  035772  017600  000004                  MOV     @4(SP),R0         ;;GET MESSAGE ADDR.
(1)  035776  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
(1)  036004  005737  001230          1$:     TST     $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
(1)  036010  001375                          BNE     1$                ;;IF NOT:  WAIT
(1)  036012  010037  001244                  MOV     R0,$MSGAD         ;;PUT ADDR IN MAILBOX
(1)  036016  105720                  2$:     TSTB    (R0)+             ;;FIND END OF MESSAGE
(1)  036020  001376                          BNE     2$
(1)  036022  163700  001244                  SUB     $MSGAD,R0         ;;SUB START OF MESSAGE
(1)  036026  006200                          ASR     R0                ;;GET MESSAGE LNGTH IN WORDS
(1)  036030  010037  001246                  MOV     R0,$MSGLGT        ;;PUT LENGTH IN MAILBOX
(1)  036034  012737  000004  001230          MOV     #4,$MSGTYPE       ;;TELL APT TO TAKE MSG.
(1)  036042  000413                          BR      5$
(1)  036044  017637  000004  036070  3$:     MOV     @4(SP),4$         ;;PUT MSG ADDR IN JSR LINKAGE
(1)  036052  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDRESS
(3)  036060  013746  177776                  MOV     177776,-(SP)      ;;PUSH 177776 ON STACK
(1)  036064  004737  034520                  JSR     PC,$TYPE          ;;CALL TYPE MACRO
(1)  036070  000000                  4$:     .WORD   0
(1)  036072                          5$:
(1)  036072  105737  036160          10$:    TSTB    $FFLG             ;;SHOULD REPORT FATAL ERROR?
(1)  036076  001416                          BEQ     12$               ;;IF NOT:  BR
(1)  036100  005737  001250                  TST     $ENV              ;;RUNNING UNDER APT?
(1)  036104  001413                          BEQ     12$               ;;IF NOT:  BR
(1)  036106  005737  001230          11$:    TST     $MSGTYPE          ;;FINISHED LAST MESSAGE?
(1)  036112  001375                          BNE     11$               ;;IF NOT:  WAIT
(1)  036114  017637  000004  001232          MOV     @4(SP),$FATAL     ;;GET ERROR #
(1)  036122  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
(1)  036130  005237  001230                  INC     $MSGTYPE          ;;TELL APT TO TAKE ERROR
(1)  036134  105037  036160          12$:    CLRB    $FFLG             ;;CLEAR FATAL FLAG
(1)  036140  105037  036157                  CLRB    $LFLG             ;;CLEAR LOG FLAG
(1)  036144  105037  036156                  CLRB    $MFLG             ;;CLEAR MESSAGE FLAG
(3)  036150  012601                          MOV     (SP)+,R1          ;;POP STACK INTO R1
(3)  036152  012600                          MOV     (SP)+,R0          ;;POP STACK INTO R0
(1)  036154  000207                          RTS     PC                ;;RETURN
(1)  036156     000          $MFLG:  .BYTE   0                 ;;MESSG. FLAG
(1)  036157     000          $LFLG:  .BYTE   0                 ;;LOG FLAG
(1)  036160     000          $FFLG:  .BYTE   0                 ;;FATAL FLAG
(1)          036162                  .EVEN
```

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

```
                                                .SBTTL  MESSAGES

036162  041501  020122  044502  EM1:    .ASCIZ          /ACR BIT3-0 LOAD-READ ERROR/
036170  031524  030055  046040
036176  040517  026504  042522
036204  042101  042440  051122
036212  051117     000
036215     101  051103  041040  EM2:    .ASCIZ          /ACR BIT3-0 WERE NOT CLEARED BY RESET (ACR-CLR)/
036222  052111  026463  020060
036230  042527  042522  047040
036236  052117  041440  042514
036244  051101  042105  041040
036252  020131  042522  042523
036260  020124  040450  051103
036266  041455  051114  000051
036274  042511  024040  043520  EM7:    .ASCIZ          /IE (PGTE BIT2) CANNOT BE SET/
036302  042524  041040  052111
036310  024462  041440  047101
036316  047516  020124  042502
036324  051440  052105     000
036331     111  020105  050050  EM10:   .ASCIZ          /IE (PGTE BIT2) CANNOT BE WRITTEN TO 0/
036336  052107  020105  044502
036344  031124  020051  040503
036352  047116  052117  041040
036360  020105  051127  052111
036366  042524  020116  047524
036374  030040     000
036377     111  020105  050050  EM11:   .ASCIZ          /IE (PGTE BIT2) CANNOT BE CLEARED BY RESET (ACR CLR)/
036404  052107  020105  044502
036412  031124  020051  040503
036420  047116  052117  041040
036426  020105  046103  040505
036434  042522  020104  054502
036442  051040  051505  052105
036450  024040  041501  020122
036456  046103  024522     000
036463     120  050124  024040  EM12:   .ASCIZ          /PTP (PGTE BIT1) CANNOT BE SET/
036470  043520  042524  041040
036476  052111  024461  041440
036504  047101  047516  020124
036512  042502  051440  052105
036520     000
036521     120  050124  024040  EM13:   .ASCIZ          /PTP (PGTE BIT1) CANNOT BE WRITTEN TO 0/
036526  043520  042524  041040
036534  052111  024461  041440
036542  047101  047516  020124
036550  042502  053440  044522
036556  052124  047105  052040
036564  020117  000060
036570  052120  020120  050050  EM14:   .ASCIZ          /PTP (PGTE BIT1) CANNOT BE CLEARED BY RESET (ACR CLR)/
036576  052107  020105  044502
036604  030524  020051  040503
036612  047116  052117  041040
036620  020105  046103  040505
036626  042522  020104  054502
036634  051040  051505  052105
```

```
                036642  024040  041501  020122
                036650  046103  024522     000
  3142          036655     120  052107  020105  EM3:    .ASCIZ          /PGTE BIT11-8 LOAD-READ ERROR/
                036662  044502  030524  026461
                036670  020070  047514  042101
                036676  051055  040505  020104
                036704  051105  047522  000122
  3143          036712  043520  042524  041040  EM4:    .ASCIZ          /PGTE BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)/
                036720  052111  030461  034055
                036726  053440  051105  020105
                036734  047516  020124  046103
                036742  040505  042522  020104
                036750  054502  051040  051505
                036756  052105  024040  041501
                036764  020122  046103  024522
                036772     000
  3144          036773     120  052107  020105  EM5:    .ASCIZ          /PGTE BIT3-0 LOAD-READ ERROR/
                037000  044502  031524  030055
                037006  046040  040517  026504
                037014  042522  042101  042440
                037022  051122  051117     000
  3145          037027     120  052107  020105  EM6:    .ASCIZ          /PGTE BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)/
                037034  044502  031524  030055
                037042  053440  051105  020105
                037050  047516  020124  046103
                037056  040505  042522  020104
                037064  054502  051040  051505
                037072  052105  024040  041501
                037100  020122  046103  024522
                037106     000
  3146          037107     123  052124  020105  EM15:   .ASCIZ          /STTE BIT11-8 LOAD-READ ERROR/
                037114  044502  030524  026461
                037122  020070  047514  042101
                037130  051055  040505  020104
                037136  051105  047522  000122
  3147          037144  052123  042524  041040  EM16:   .ASCIZ          /STTE BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)/
                037152  052111  030461  034055
                037160  053440  051105  020105
                037166  047516  020124  046103
                037174  040505  042522  020104
                037202  054502  051040  051505
                037210  052105  024040  041501
                037216  020122  046103  024522
                037224     000
  3148          037225     123  052124  020105  EM17:   .ASCIZ          /STTE BIT3-0 LOAD-READ ERROR/
                037232  044502  031524  030055
                037240  046040  040517  026504
                037246  042522  042101  042440
                037254  051122  051117     000
  3149          037261     123  052124  020105  EM20:   .ASCIZ          /STTE BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)/
                037266  044502  031524  030055
                037274  053440  051105  020105
                037302  047516  020124  046103
                037310  040505  042522  020104
                037316  054502  051040  051505
                037324  052105  024040  041501
```

```
                037332  020122  046103  024522
                037340     000
3150            037341     114  042513  024040  EM21:   .ASCIZ       /LKE (STCS BIT2) CANNOT BE SET/
                037346  052123  051503  041040
                037354  052111  024462  041440
                037362  047101  047516  020124
                037370  042502  051440  052105
                037376     000
3151            037377     114  042513  024040  EM22:   .ASCIZ       /LKE (STCS BIT2) CANNOT BE WRITTEN TO 0/
                037404  052123  051503  041040
                037412  052111  024462  041440
                037420  047101  047516  020124
                037426  042502  053440  044522
                037434  052124  047105  052040
                037442  020117  000060
3152            037446  045514  020105  051450  EM23:   .ASCIZ       /LKE (STCS BIT2) CANNOT BE CLEARED BY RESET (ACR CLR)/
                037454  041524  020123  044502
                037462  031124  020051  040503
                037470  047116  052117  041040
                037476  020105  046103  040505
                037504  042522  020104  054502
                037512  051040  051505  052105
                037520  024040  041501  020122
                037526  046103  024522     000
3153            037533     123  050124  024040  EM24:   .ASCIZ       /STP (STCS BIT1) CANNOT BE SET/
                037540  052123  051503  041040
                037546  052111  024461  041440
                037554  047101  047516  020124
                037562  042502  051440  052105
                037570     000
3154            037571     123  050124  024040  EM25:   .ASCIZ       /STP (STCS BIT1) CANNOT BE WRITTEN TO 0/
                037576  052123  051503  041040
                037604  052111  024461  041440
                037612  047101  047516  020124
                037620  042502  053440  044522
                037626  052124  047105  052040
                037634  020117  000060
3155            037640  052123  020120  051450  EM26:   .ASCIZ       /STP (STCS BIT1) CANNOT BE CLEARED BY RESET (ACR CLR)/
                037646  041524  020123  044502
                037654  030524  020051  040503
                037662  047116  052117  041040
                037670  020105  046103  040505
                037676  042522  020104  054502
                037704  051040  051505  052105
                037712  024040  041501  020122
                037720  046103  024522     000
3156            037725     105  041116  024040  EM27:   .ASCIZ       /ENB (STCS BIT0) CANNOT BE SET/
                037732  052123  051503  041040
                037740  052111  024460  041440
                037746  047101  047516  020124
                037754  042502  051440  052105
                037762     000
3157            037763     105  041116  024040  EM30:   .ASCIZ       /ENB (STCS BIT0) CANNOT BE WRITTEN TO 0/
                037770  052123  051503  041040
                037776  052111  024460  041440
                040004  047101  047516  020124
```

```
                040012  042502  053440  044522
                040020  052124  047105  052040
                040026  020117  000060
  3158          040032  047105  020102  051450  EM31:   .ASCIZ       /ENB (STCS BIT0) CANNOT BE CLEARED BY RESET (ACR CLR)/
                040040  041524  020123  044502
                040046  030124  020051  040503
                040054  047116  052117  041040
                040062  020105  046103  040505
                040070  042522  020104  054502
                040076  051040  051505  052105
                040104  024040  041501  020122
                040112  046103  024522     000
  3159          040117     123  041524  020123  EM32:   .ASCIZ       /STCS BIT15-8 LOAD-READ ERROR/
                040124  044502  030524  026465
                040132  020070  047514  042101
                040140  051055  040505  020104
                040146  051105  047522  000122
  3160          040154  052123  051503  041040  EM33:   .ASCIZ       /STCS BIT15-8 WERE NOT CLEARED BY RESET (ACR CLR)/
                040162  052111  032461  034055
                040170  053440  051105  020105
                040176  047516  020124  046103
                040204  040505  042522  020104
                040212  054502  051040  051505
                040220  052105  024040  041501
                040226  020122  046103  024522
                040234     000
  3161          040235     111  051515  020113  EM34:   .ASCIZ       /IMSK BIT11-8 LOAD-READ ERROR/
                040242  044502  030524  026461
                040250  020070  047514  042101
                040256  051055  040505  020104
                040264  051105  047522  000122
  3162          040272  046511  045523  041040  EM35:   .ASCIZ       /IMSK BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)/
                040300  052111  030461  034055
                040306  053440  051105  020105
                040314  047516  020124  046103
                040322  040505  042522  020104
                040330  054502  051040  051505
                040336  052105  024040  041501
                040344  020122  046103  024522
                040352     000
  3163          040353     111  051515  020113  EM36:   .ASCIZ       /IMSK BIT3-0 LOAD-READ ERROR/
                040360  044502  031524  030055
                040366  046040  040517  026504
                040374  042522  042101  042440
                040402  051122  051117     000
  3164          040407     111  051515  020113  EM37:   .ASCIZ       /IMSK BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)/
                040414  044502  031524  030055
                040422  053440  051105  020105
                040430  047516  020124  046103
                040436  040505  042522  020104
                040444  054502  051040  051505
                040452  052105  024040  041501
                040460  020122  046103  024522
                040466     000
  3165          040467     115  041524  041040  EM40:   .ASCIZ       /MTC BIT11-8 LOAD-READ ERROR/
                040474  052111  030461  034055
```

```
        040502  046040  040517  026504
        040510  042522  042101  042440
        040516  051122  051117     000
3166    040523     115  041524  041040  EM41:   .ASCIZ       /MTC BIT11-8 WERE NOT CLEARED BY RESET (ACR CLR)/
        040530  052111  030461  034055
        040536  053440  051105  020105
        040544  047516  020124  046103
        040552  040505  042522  020104
        040560  054502  051040  051505
        040566  052105  024040  041501
        040574  020122  046103  024522
        040602     000
3167    040603     115  041524  041040  EM42:   .ASCIZ       /MTC BIT3-0 LOAD-READ ERROR/
        040610  052111  026463  020060
        040616  047514  042101  051055
        040624  040505  020104  051105
        040632  047522  000122
3168    040636  052115  020103  044502  EM43:   .ASCIZ       /MTC BIT3-0 WERE NOT CLEARED BY RESET (ACR CLR)/
        040644  031524  030055  053440
        040652  051105  020105  047516
        040660  020124  046103  040505
        040666  042522  020104  054502
        040674  051040  051505  052105
        040702  024040  041501  020122
        040710  046103  024522     000
3169    040715     111  051511  020124  EM44:   .ASCII       /IIST ERROR - REFER TO THE LISTING AT "PC"/
        040722  051105  047522  020122
        040730  020055  042522  042506
        040736  020122  047524  052040
        040744  042510  046040  051511
        040752  044524  043516  040440
        040760  020124  050042  021103
3170    040766  052200  020117  042504          .ASCIZ       <CRLF>/TO DETERMINE THE ERROR./
        040774  042524  046522  047111
        041002  020105  044124  020105
        041010  051105  047522  027122
        041016     000
3171
3172    041017     101  051103  024040  EM45:   .ASCIZ       /ACR (3:0) AUTO-INCREMENT ERROR/
        041024  035063  024460  040440
        041032  052125  026517  047111
        041040  051103  046505  047105
        041046  020124  051105  047522
        041054  000122
```

```
 3174  041056  040520  051523  020040  DH1:    .ASCIZ          /PASS    PC      EXPCTD  ACTUAL  ACR/
       041064  020040  041520  020040
       041072  020040  020040  054105
       041100  041520  042124  020040
       041106  041501  052524  046101
       041114  020040  041501  000122
 3175  041122  040520  051523  020040  DH2:    .ASCIZ          /PASS    PC      ACR/
       041130  020040  041520  020040
       041136  020040  020040  041501
       041144  000122
 3176  041146  040520  051523  020040  DH3:    .ASCIZ          /PASS    PC      PGTE/
       041154  020040  041520  020040
       041162  020040  020040  043520
       041170  042524     000
 3177  041173     120  051501  020123  DH4:    .ASCIZ          /PASS    PC      EXPCTD  ACTUAL  PGTE/
       041200  020040  050040  020103
       041206  020040  020040  042440
       041214  050130  052103  020104
       041222  040440  052103  040525
       041230  020114  050040  052107
       041236  000105
 3178  041240  040520  051523  020040  DH5:    .ASCIZ          /PASS    PC      EXPCTD  ACTUAL  STTE/
       041246  020040  041520  020040
       041254  020040  020040  054105
       041262  041520  042124  020040
       041270  041501  052524  046101
       041276  020040  052123  042524
       041304     000
 3179  041305     120  051501  020123  DH6:    .ASCIZ          /PASS    PC      STTE/
       041312  020040  050040  020103
       041320  020040  020040  051440
       041326  052124  000105
 3180  041332  040520  051523  020040  DH7:    .ASCIZ          /PASS    PC      STCS/
       041340  020040  041520  020040
       041346  020040  020040  052123
       041354  051503     000
 3181  041357     120  051501  020123  DH8:    .ASCIZ          /PASS    PC      EXPCTD  ACTUAL  STCS/
       041364  020040  050040  020103
       041372  020040  020040  042440
       041400  050130  052103  020104
       041406  040440  052103  040525
       041414  020114  051440  041524
       041422  000123
 3182  041424  040520  051523  020040  DH9:    .ASCIZ          /PASS    PC      EXPCTD  ACTUAL  IMSK/
       041432  020040  041520  020040
       041440  020040  020040  054105
       041446  041520  042124  020040
       041454  041501  052524  046101
       041462  020040  046511  045523
       041470     000
 3183  041471     120  051501  020123  DH10:   .ASCIZ          /PASS    PC      IMSK/
       041476  020040  050040  020103
       041504  020040  020040  044440
       041512  051515  000113
 3184  041516  040520  051523  020040  DH11:   .ASCIZ          /PASS    PC      EXPCTD  ACTUAL  MTCE/
       041524  020040  041520  020040
```

```
        041532  020040  020040  054105
        041540  041520  042124  020040
        041546  041501  052524  046101
        041554  020040  052115  042503
        041562     000
 3185   041563     120  051501  020123  DH12:   .ASCIZ          /PASS    PC      MTCE/
        041570  020040  050040  020103
        041576  020040  020040  046440
        041604  041524  000105
 3186   041610  040520  051523  020040  DH13:   .ASCIZ          /PASS    PC      LINE #  EXPCTD  ACTUAL  DISPLY  $TMP0   $TMP1/
        041616  020040  041520  020040
        041624  020040  020040  044514
        041632  042516  021440  020040
        041640  054105  041520  042124
        041646  020040  041501  052524
        041654  046101  020040  044504
        041662  050123  054514  020040
        041670  052044  050115  020060
        041676  020040  052044  050115
        041704  000061
 3187   041706  100200  041501  020122  ASKDVA: .ASCIZ  <CRLF><CRLF>/ACR ADDRESS: /
        041714  042101  051104  051505
        041722  035123  000040
 3188   041726  047111  042524  051122  ASKPIV: .ASCIZ  /INTERRUPT VECTOR ADDRESS: /
        041734  050125  020124  042526
        041742  052103  051117  040440
        041750  042104  042522  051523
        041756  020072     000
 3189   041761     102  020122  042514  ASKBRL: .ASCIZ  /BR LEVEL (4-7): /
        041766  042526  020114  032050
        041774  033455  035051  000040
 3190   042002  050200  052107  020105  HDRMES: .ASCII  <CRLF>/PGTE    PGCS    STTE    STCS    IMSK    PGF     STF     DCF     E
        042010  020040  050040  041507
        042016  020123  020040  051440
        042024  052124  020105  020040
        042032  051440  041524  020123
        042040  020040  044440  051515
        042046  020113  020040  050040
        042054  043107  020040  020040
        042062  051440  043124  020040
        042070  020040  042040  043103
        042076  020040  020040  042440
        042104  041530
 3191   042106  000200                  .ASCIZ  <CRLF>
 3192   042110  000040          SPACE:  .ASCIZ  / /
 3193   042112  020040     000  TWOSP:  .ASCIZ  /  /
 3194   042115     200  042523  043114  DMPSID: .ASCIZ  <CRLF>/SELF ID = /
        042122  044440  020104  020075
        042130     000
 3195   042131     200     000          CARET:  .ASCIZ  <CRLF>
 3196           042134                  .EVEN
 3197
 3198   042134  001236  001116  001124  DT1:    .WORD   $PASS,$ERRPC,$GDDAT,$BDDAT,DISPLY,0
        042142  001126  024750  000000
 3199   042150  001236  001116  024750  DT2:    .WORD   $PASS,$ERRPC,DISPLY,0
        042156  000000
```

```
 3200  042160  001236  001116  024754  DT3:    .WORD   $PASS,$ERRPC,LINMBR,$GDDAT,$BDDAT,DISPLY,$TMP0,$TMP1,0
       042166  001124  001126  024750
       042174  001176  001200  000000
 3201  042202     001     000     000  DAF1:   .BYTE   1,0,0,0,0,0,0,0
       042205     000     000     000
       042210     000     000
 3202                                   .EVEN
 3203
 3204          000001                   .END            ;THAT'S ALL FOLKS!
```

```
ABASE = 000000            254
ACDW1 = 000000            254
ACDW2 = 000000            254
ACPUOP= 000000            254
ACR     024732            778*   779    791*   792*   793    806*   809    816*   818    823*   825    830*   832
                          837*   839    854*   856*   858*   860*   862*   864*   866*   868*   870*   872*   874*
                          876*   878*   880*   901*   907*   914*   928*   933*   941*   958*   960    962    1011*
                         1014*  1022*  1032*  1039*  1044*  1049*  1057*  1059*  1061*  1079*  1103*  1107*  1123*
                         1127*  1137*  1144*  1149*  1154*  1159*  1164*  1184*  1187*  1195*  1205*  1212*  1217*
                         1222*  1230*  1232*  1234*  1252*  1263*  1267*  1283*  1287*  1297*  1304*  1309*  1314*
                         1319*  1324*  1340*  1342*  1344*  1346*  1349*  1358*  1365*  1367*  1373*  1386*  1388*
                         1390*  1392*  1395*  1404*  1411*  1425*  1427*  1429*  1431*  1434*  1443*  1450*  1452*
                         1458*  1471*  1473*  1475*  1477*  1480*  1489*  1496*  1508*  1511*  1525*  1540*  1550*
                         1554*  1562*  1577*  1590*  1593*  1599*  1608*  1630*  1641*  1648*  1652*  1666*  1672*
                         1677*  1684*  1700*  1724*  1726*  1737*  1744*  1748*  1761*  1768*  1784*  1800    1849
                         1885   1931*  1937*  1941*  1965*  1973*  1980*  1993*  2005*  2043*  2058*  2060*  2069*
                         2071*  2081*  2083*  2093*  2105*  2107*  2115*  2117*  2119   2120   2121   2123   2124
                         2126*  2143*  2149*  2152*  2153*  2157*  2181*  2188*  2192*  2193*  2201*  2208*  2220*
                         2227*  2233*  2284*  2289*  2296   2339*  2344*  2349*  2355*  2431*  2444*  2448   2449
                         2450*  2452*  2459*  2471*  2504   2505*  2511*  2533   2538*  2539   2541   2543   2602*
                         2603   2618#  2694*  2703*  2713*  2719*  2721*  2728*  2732   2733   2734   2735   2738*
                         2740*  2760*  2768*  2773*  2775*  2780*  2796*  2801   2802   2803   2804   2805   2806
                         2813*  2825*  2831*  2833*  2841*  2844*  2850*  2854*  2860*  2866*  2876*  2883*  2885*
                         2912*  2929*  2932   2933   2934   2953*  2958*  2966*  3022*  3031*  3035*  3039   3080*
                         3086*  3093*  3102
ACRH    024734           2286   2543*  2544*  2545   2619#
ADDW0 = 000000            254
ADDW1 = 000000            254
ADDW10= 000000            254
ADDW11= 000000            254
ADDW12= 000000            254
ADDW13= 000000            254
ADDW14= 000000            254
ADDW15= 000000            254
ADDW2 = 000000            254
ADDW3 = 000000            254
ADDW4 = 000000            254
ADDW5 = 000000            254
ADDW6 = 000000            254
ADDW7 = 000000            254
ADDW8 = 000000            254
ADDW9 = 000000            254
ADEVCT= 000000            254
ADEVM = 000000            254
ADR     024736            807    817*   824*   831*   902    915    934    952*   963    1801   1850   1886   2297
                         2506   2545*  2546*  2547   2621#  3051
ADRH    024740           838*   2547*  2548*  2632#
AENV  = 000000            254
AENVM = 000000            254
AFATAL= 000000            254
AMADR1= 000000            254
AMADR2= 000000            254
AMADR3= 000000            254
AMADR4= 000000            254
AMAMS1= 000000            254
AMAMS2= 000000            254
```

```
AMAMS3= 000000   254
AMAMS4= 000000   254
AMSGAD= 000000   254
AMSGLG= 000000   254
AMSGTY= 000000   254
AMTYP1= 000000   254
AMTYP2= 000000   254
AMTYP3= 000000   254
AMTYP4= 000000   254
APASS = 000000   254
APRIOR= 000000   254
APTCSU= 000040  3125  3130#
APTENV= 000001  3120  3125  3130#
APTSIZ= 000200   703  3130#
APTSPO= 000100  3125  3130#
ASKBRL  041761  2562  3189#
ASKDVA  041706  2532  3187#
ASKPIV  041726  2549  3188#
ASWREG= 000000   254
ATESTN= 000000   254
AUNIT = 000000   254
AUSWR = 000000   254
AVECT1= 000000   254
AVECT2= 000000   254
BEGIN   002666   719   730#
BIT0  = 000001   241#  504   520   531   538   549   560   567   571   582   593   606   856
                 864   876   880  2499
BIT00 = 000001   241#
BIT01 = 000002   241#
BIT02 = 000004   241#
BIT03 = 000010   241#
BIT04 = 000020   241#
BIT05 = 000040   241#
BIT06 = 000100   241#
BIT07 = 000200   241#
BIT08 = 000400   241#
BIT09 = 001000   241#  3119  3120
BIT1  = 000002   241#  503   519   530   537   548   559   566   570   581   592   605
BIT10 = 002000   241#  498   525   543   554   576   587   598  3120
BIT11 = 004000   241#  497   512   524   542   553   575   586   597  3119
BIT12 = 010000   241#  511
BIT13 = 020000   241#  510  3120
BIT14 = 040000   241#  509  3119
BIT15 = 100000   241#  483   508
BIT2  = 000004   241#  502   518   529   536   547   558   565   569   580   591   604
BIT3  = 000010   241#  501   517   528   535   546   557   564   568   579   590   603
BIT4  = 000020   241#
BIT5  = 000040   241#
BIT6  = 000100   241#
BIT7  = 000200   241#
BIT8  = 000400   241#  500   514   516   527   545   556   578   589   600   602   854   862
                 872   874   878
BIT9  = 001000   241#  499   515   516   526   544   555   577   588   601   602  2516
BMITAB  022002  1389  1474  2264#
BM0   = 000400   545#
BM1   = 001000   544#
```

```
BM2    = 002000  543#
BM3    = 004000  542#
BPTVEC= 000014  241#
BRKTAB  022002  2038  2052  2063  2074  2267#
BRK0   = 000400  578#
BRK1   = 001000  577#
BRK2   = 002000  576#
BRK3   = 004000  575#
CARET   042131  2510  2610  2611  3195#
CKSWR  = 104407  2515  3119  3120  3128#
CLEAR   023454  775  854  856  858  860  862  864  866  868  870  872  874  876
                878  880  2471#
CLEARX  023450  899  929  1935  1940  2031  2114  2144  2467#  2756  2923  2950  3021  3050
                3079
CLR    = 100000  483#  792  928  1931  2143  2152  2431  2444  2602
CNFMAX  003004  748#  2581  2588
CONFIG  025020  733  739  893  1932  1944  2034  2046  2049  2145  2160  2587*  2588  2654#
CR     = 000015  241#  3125
CRLF   = 000200  241#  613#  704  716  732  2676  2677  2678  2679  2680  2683  2689  2765
                2789  2816  2908  2925  2926  2952  2963  2986  2988  2990  2991  2992  2993
                2994  2995  2996  3000  3125  3170  3187  3190  3191  3194  3195
DAF1    042202  261  267  273  279  285  291  297  303  309  315  321  327  333
                339  345  351  357  363  369  375  381  387  393  399  405  411
                417  423  429  435  441  447  453  459  465  471  477  3201#
DCF     024736  1040  1145  1213  1305  1631  1694  1725  1778  1938  2017  2044  2061  2072
                2082*  2106*  2108  2127  2150  2156*  2234  2345  2462*  2629#  2834  2842*  2954
                2959  2967*
DCFE   = 000007  491#  1039  1144  1212  1304  1630  1724  1937  2043  2060  2071  2081  2105
                2126  2149  2233  2344  2833  2841  2953  2958  2966
DCFH    024740  2637#
DCFTAB  021772  2039  2064  2106  2256#
DCF0   = 000001  582#
DCF1   = 000002  581#
DCF2   = 000004  580#
DCF3   = 000010  579#
DDISP  = 177570  241#  254  703
DH1     041056  259  475  3174#
DH10    041471  427  439  3183#
DH11    041516  445  457  3184#
DH12    041563  451  463  3185#
DH13    041610  469  3186#
DH2     041122  265  3175#
DH3     041146  277  289  295  301  307  313  319  325  3176#
DH4     041173  271  283  3177#
DH5     041240  331  343  3178#
DH6     041305  337  349  3179#
DH7     041332  355  361  367  373  379  385  391  397  403  415  3180#
DH8     041357  409  3181#
DH9     041424  421  433  3182#
DISPLA  001142  254#  703*  3119*  3120*
DISPLY  024750  779*  780  793*  794  803*  806  816  823  830  837  844*  846  851
                854*  856*  858*  860*  862*  864*  866*  868*  870*  872*  874*  876*  878*
                880*  1040*  1045*  1050*  1062*  1063  1145*  1150*  1155*  1213*  1218*  1223*  1235*
                1236  1305*  1310*  1315*  1368*  1369  1453*  1454  1512*  1513  1526*  1528  1535
                1541*  1542  1555*  1556  1563*  1565  1572  1578*  1579  2417*  2453*  2454  2647#
                2697*  2709  2786*  2794  2811  2817  2886*  2889  2890  2954*  2955  2960  2968*
```

```
                    2969*  3198   3199   3200
DISPRE  000174       245#   703
DMPREG  023526       785    797    813    822    829    836    843    854    856    858    860    862    864
                     866    868    870    872    874    876    878    880    920    967   1020   1031   1038
                    1043   1048   1053   1066   1077   1089   1100   1111   1136   1143   1148   1153   1158
                    1169   1193   1204   1211   1216   1221   1226   1239   1250   1258   1271   1296   1303
                    1308   1313   1318   1329   1356   1363   1372   1377   1402   1409   1415   1441   1448
                    1457   1462   1487   1494   1500   1518   1533   1539   1545   1561   1570   1576   1582
                    1606   1613   1659   1663   1671   1682   1688   1692   1698   1755   1759   1765   1772
                    1776   1782   1816   1830   1843   1867   1879   1902   1918   1987   1991   1998   2010
                    2015   2021   2057   2068   2078   2088   2099   2113   2132   2200   2206   2213   2225
                    2231   2238   2320   2337   2343   2348   2353   2360   2412   2421   2427   2457   2499#
                    2724   2743   2772   2779   2830   2840   2849   2858   2864   2870   2881   2894   2899
                    2941
DMPSID  042115      2601   3194#
DSBT  = 000010       603#   607   1942   2032   2037   2059   2118   2122   2158   2285
DSWR  = 177570       241#   254    703
DTAB    003540       741    764#
DT1     042134       260    272    284    302    332    344    410    422    434    446    458    476   3198#
DT2     042150       266    278    290    296    308    314    320    326    338    350    356    362    368
                     374    380    386    392    398    404    416    428    440    452    464   3199#
DT3     042160       470   3200#
EMTVEC= 000030       241#   703*
EM1     036162       258   3134#
EM10    036331       300   3137#
EM11    036377       306   3138#
EM12    036463       312   3139#
EM13    036521       318   3140#
EM14    036570       324   3141#
EM15    037107       330   3146#
EM16    037144       336   3147#
EM17    037225       342   3148#
EM2     036215       264   3135#
EM20    037261       348   3149#
EM21    037341       354   3150#
EM22    037377       360   3151#
EM23    037446       366   3152#
EM24    037533       372   3153#
EM25    037571       378   3154#
EM26    037640       384   3155#
EM27    037725       390   3156#
EM3     036655       270   3142#
EM30    037763       396   3157#
EM31    040032       402   3158#
EM32    040117       408   3159#
EM33    040154       414   3160#
EM34    040235       420   3161#
EM35    040272       426   3162#
EM36    040353       432   3163#
EM37    040407       438   3164#
EM4     036712       276   3143#
EM40    040467       444   3165#
EM41    040523       450   3166#
EM42    040603       456   3167#
EM43    040636       462   3168#
EM44    040715       467   3169#
```

```
EM45    041017          474   3172#
EM5     036773          282   3144#
EM6     037027          288   3145#
EM7     036274          294   3136#
ENB   = 000001          538#   870   1510   1534   1552   1571   1592   1889   2730   2930   3056
ENDTST  022012          996   2273#
ERR   = 100000          508#  1808   1836   1837   1859   1872   1873   1891   1911   1912
ERRVEC= 000004          241#   703*  3119*  3122*
EXC     024736         1045   1150   1218   1310   1359   1366*  1368   1405   1444   1451*  1453   1490   1667
                       1674*  1678   1762   1994   2209   2350   2630#  2720   2739   2827   2832*
EXCE  = 000010          492#  1044   1149   1217   1309   1358   1365   1367   1404   1443   1450   1452   1489
                       1666   1761   1993   2208   2349   2719   2738   2825   2831
EXCH    024740         2638#
GETPAR  023644          710   2526#
GETSID  024634          882   2596#  2671   2754   2922   2951   2977
GNS   = ****** U        245    704    716    732   2379   2393   2407   2526   2531   2574   2585   2676   2677
                       2678   2679   2680   2683   2689   2765   2789   2816   2908   2925   2926   2952   2963
                       3000   3005   3128
GO    = 000001          520#  1013   1125   1186   1285   1345   1391   1430   1476   1804   1851   1855   2300
                       2715   2798   2815   3037
GRJ   = 040000          509#  1808   1837
GTSWR = 104406          704   3128#
HDRMES  042002         2502   3190#
HT    = 000011          241#  3125
IE    = 000004          518#   858   1080   1253   1825   2094   2300   2333   2761   2872
IIIL    024746         1097   2563   2568*  2569   2571   2642#  2758*
IIIP    024744         1067   1070*  1102*  1240   1243*  1262*  1820   1823*  1834*  2090   2092*  2102*  2291
                       2294*  2430*  2560*  2561*  2641#
IIIV    024742         1068   1069*  1078*  1101*  1241   1242*  1251*  1261*  1821   1822*  1833*  2089   2091*
                       2103*  2292   2293*  2429*  2550   2555*  2556   2558   2560   2640#  2757*
IMITAB  021772         1343   1428   2253#
IMSK    024736          874*   876*  1343*  1389*  1428*  1474*  1727*  2116*  2626#  2704*  3032*
IMSKE = 000004          488#   874    876   1342   1388   1427   1473   1726   2115   2192   2703   3031
IMSKH   024740         2635#
IM0   = 000001          549#
IM1   = 000002          548#
IM2   = 000004          547#
IM3   = 000010          546#
IOTVEC= 000020          241#   703*
IP    = 000010          517#  1063   1236   1542   1579   1651   1808   1837   1859   1873   1891   1912   1979
                       2080   2333   2872
KIPAR0= 172340          244#  3122
KIPAR1= 172342          244#
KIPAR2= 172344          244#
KIPAR3= 172346          244#
KIPAR4= 172350          244#
KIPAR5= 172352          244#
KIPAR6= 172354          244#
KIPAR7= 172356          244#  3122
KIPDR0= 172300          244#
KIPDR1= 172302          244#
KIPDR2= 172304          244#
KIPDR3= 172306          244#
KIPDR4= 172310          244#
KIPDR5= 172312          244#
KIPDR6= 172314          244#
```

```
KIPDR7= 172316           244#
LF    = 000012           241#   3125
LINE0   010536           982#
LINE1   010576           990#
LINE2   010610           993#
LINMBR  024754           998*  2287*  2649#  2685*  2696   2698*  3200
LINTST  010626           983    985    987    989    998#
LKE   = 000004           536#   866   2730   2930
MAXCNT  025026          2374*  2390*  2404*  2416   2659#
MCP     027750           765   2949#
MDSD  = 000001           606#  2059   2122   2285   2695
MENB  = 000004           604#   607
MFRM  = 002000           598#  1701   1707   1785   1791
MID0  = 000000           599#
MID1  = 000400           600#
MID2  = 001000           601#
MID3  = 001400           602#
MINCNT  025024          2375*  2389*  2403*  2423   2658#
MLEN  = 000002           605#   607   2037   2118   2285   2695   3023
MMBOOT  025030           765   2670#
MMVEC = 000250           244#
MNTBTS  023446          2443*  2447   2465#  2481*
MPDBT1  026166          2755#  2915
MPDIBT  026160           765   2753#
MPDINT  026714          2757   2824#
MTAB    002762           735    744#   748   2575
MTC     024736           878*   880*  1701*  1704*  1707*  1785*  1788*  1791*  1942*  2059*  2070*  2118*  2122*
                        2158*  2285*  2451*  2631#  2695*  3023*
MTCE  = 000015           493#   878    880    947   1700   1784   1941   2058   2069   2117   2157   2284   2450
                        2694   3022
MTCH    024740          2639#
MTYP  = 004000           597#  1701   1704   1785   1788   2481
N     = 000044           799#   854#   856#   858#   860#   862#   864#   866#   868#   870#   872#   874#   876#
                         878#   880#
NOPAR   002644           245    723#
PARCAL  023506          1635   1731   1958   2174   2488#  3082
PARFLG  025016           708    723*   727*  2652#
PBFTAB  022002          1160   2265#
PBF0  = 000400           556#
PBF1  = 001000           555#
PBF2  = 002000           554#
PBF3  = 004000           553#
PB0   = 000400           500#
PB1   = 001000           499#
PB2   = 002000           498#
PB3   = 004000           497#
PGBTAB  022002          1124   1284   1387   1398   1472   1483   2002   2004   2262#  2701   2706   2714   2797
                        3029   3036
PGCS    024736           858*   860*  1013*  1015   1023   1062   1080*  1125*  1128   1186*  1188   1196   1235
                        1253*  1285*  1288   1345*  1347   1391*  1393   1430*  1432   1476*  1478   1541   1578
                        1649*  1653   1745*  1749   1974*  1981   2084   2094*  2453   2623#  2715*  2761*  2798*
                        2815*  2877   2884*  3037*  3088*  3094
PGCSE = 000001           485#   858    860    943   1014   1022   1061   1079   1127   1187   1195   1234   1252
                        1287   1344   1346   1390   1392   1429   1431   1475   1477   1540   1577   1648   1744
                        1809   1824   1860   1871   1874   1897   1910   1913   1973   2083   2093   2298   2452
                        2760   2876   2883   3093
```

```
PGCSH   024740          2633#
PGF     024736          1033    1058*   1060    1105*   1108    1138    1161*   1165    1223    1315    1350    1396    1459
                        1497    1685    1769    2006    2154*   2228    2340    2460*   2627#   2722    2769    2774*   2886
                        2913*
PGFE  = 000005           489#   1032    1057    1059    1103    1107    1137    1159    1164    1222    1314    1349    1395
                        1458    1496    1684    1768    1818    1869    2005    2153    2227    2339    2459    2721    2768
                        2773    2885    2912
PGFH    024740          2636#
PGITAB  021772          1012    1056    1185    1341    1352    1426    1437    1803    1854    1946    1977    1999    2001
                        2251#   2814    3106
PGMR  = 020000           510#   1859    1873
PGONL   017400           992    1929#
PGTE    024736           854*    856*   1012*   1124*   1185*   1284*   1341*   1387*   1426*   1472*   1642*   1738*   1966*
                        2622#   2714*   2797*   2814*   2855    3036*   3087*
PGTEE = 000000           484#    854     856    1011    1123    1184    1283    1340    1386    1425    1471    1641    1737
                        1802    1852    1965    2713    2796    2813    2854    3035    3080    3086
PIFTAB  021772          1104    1673    2254#
PIF0  = 000001           560#
PIF1  = 000002           559#
PIF2  = 000004           558#
PIF3  = 000010           557#
PIRQ  = 177772           241#
PIRQVE= 000240           241#
PI0   = 000001           504#
PI1   = 000002           503#
PI2   = 000004           502#
PI3   = 000010           501#
POP   = 005726           611#
POPPOP= 022626           612#   1075    1091    1248    1260    1832    2101
PRDY  = 004000           512#   1016    1024    1029    1129    1134    1189    1197    1202    1289    1294    1347    1393
                        1432    1478    1651    1654    1747    1750    1808    1811    1859    1862    1891    1976    1982
                        2080    2299    2333    2454    2872    3094
PR0   = 000000           241#
PR1   = 000040           241#
PR2   = 000100           241#
PR3   = 000140           241#
PR4   = 000200           241#
PR5   = 000240           241#
PR6   = 000300           241#
PR7   = 000340           241#   2755    2758
PS    = 177776           241#   1071*   1073*   1083*   1087*   1244*   1246*   1254*   1256*   1826*   1828*   2095*   2097*
PSW   = 177776           241#    717*   1009*   2282*   2303*   2315*   2755*   2763*
PTP   = 000002           519#    860    1186    1285    1430    1476    2300    2333
PWRVEC= 000024           241#    703*   3129*
RDCHR = 104410          2527    2927    3124    3128#
RDLIN = 104411          3123    3128#
RDOCT = 104412          2535    2552    2565    2586    2684    2785    3001    3128#
RESVEC= 000010           241#
RTETAB  021772          2257#
RTE0  = 000001           593#   2257
RTE1  = 000002           592#   2258
RTE2  = 000004           591#   2259
RTE3  = 000010           590#   2260
SBFTAB  022002          1320    1551    2266#
SBF0  = 000001           567#
SBF1  = 000002           566#
```

```
SBF2  = 000004   565#
SBF3  = 000010   564#
SBO   = 000400   527#
SB1   = 001000   526#
SB2   = 002000   525#
SB3   = 004000   524#
SCBOOT  031420   3012   3021#
SCPBAL  032040   3016   3114#
SCPIAL  032026   3015   3110#
SCPISF  032000   3013   3102#
SCPXMT  031664   3079#  3098   3107   3111   3115
SCSTSF  031554   3014   3050#  3067
SELPAR  002654    246    727#
SETMNT  023306   1010   1122   1339   1385   1628   1722   1799   1848   2036   2283   2443#  2693
SETMNX  023414   2459#  2467
SETMN1  023312   2444#  2482
SIDNUM  024730    990    993   2606*  2614#  2707   2873   3033
SIDTAB  021762   1018   1028   1133   1191   1201   1293   1645   1741   1807   1858   1890   1970   2079
                 2246#  2445   2875
SIDO  = 000000    513#  2246
SID1  = 000400    514#  2247
SID2  = 001000    515#  2248
SID3  = 001400    516#  2249
SIFTAB  021772   1264   1509   1591   2255#
SIFO  = 000001    571#
SIF1  = 000002    570#
SIF2  = 000004    569#
SIF3  = 000010    568#
SIO   = 000001    531#
SI2   = 000002    529#   530#
SI3   = 000010    528#
SLDISP  031406   3009   3012#
SLMSG   030214   2983   2986#
SLOOPS  030160    766   2976#
SLOOP1  031232   2980   2984   2999#  3006   3046   3061   3092
SNAPO   024756    898    912    930    957   1631*  1693   1725*  1777   1938*  2016   2044*  2045   2053
                 2150*  2232   2650#
SPACE   042110   2534   2551   2564   3192#
SRO   = 177572    244#  3122*
SR1   = 177574    244#
SR2   = 177576    244#
SR3   = 172516    244#  3122*
SSTMNT  023476   1183   1282   1424   1470   1506   1549   1589   1884   2481#
STACK = 001100    241#   703   2762   2824   2914   2928
START   002024    703#   724    728
STBTAB  022002   2217   2219   2263#  2729
STCCNT  025022    718*  2363   2373*  2388*  2402*  2417   2656#
STCS    024736    866*   868*   870*   872*  1510*  1512   1526   1552*  1555   1563   1592*  2189*  2194
                 2202   2625#  2730*  2867   2930*
STCSE = 000003    487#   866    868    870    872    945   1511   1525   1554   1562   1892   2188   2201
                 2301   2317   2866   2929   3062
STCSH   024740   1594*  1600   1609   2634#
STF     024736   1050   1155   1206   1231*  1233   1265*  1268   1298   1321*  1325   1374   1412   1435
                 1481   1689   1773   2012   2155*  2221   2356   2461*  2628#  2741   2776   2781*  2846
                 2851*
STFE  = 000006    490#  1049   1154   1205   1230   1232   1263   1267   1297   1319   1324   1373   1411
```

```
                    1434    1480    1908    2220    2355    2740    2775    2780    2844    2850
STHLT   027500       765  2921#
STITAB  021772      1229    1888    2162    2214    2216    2252#   2290    3055
STKLMT= 177774       241#
STMR    = 010000     511#    1891    1912
STONL   021134       995  2141#
STP     = 000002     537#     868
STTE    024736      862*    864*   1509*   1551*   1591*   2182*   2290*   2624#   2729*   2861
STTEE   = 000002     486#     862     864    1508    1550    1590    1887    1895    2181    2289    2728    2860    3054
SWR     001140      254#    703*     704    2499    2512    2516    3119    3120    3124*   3129*
SWREG   000176      245#     703     704    3124
SW0     = 000001     241#
SW00    = 000001     241#
SW01    = 000002     241#
SW02    = 000004     241#
SW03    = 000010     241#
SW04    = 000020     241#
SW05    = 000040     241#
SW06    = 000100     241#
SW07    = 000200     241#
SW08    = 000400     241#
SW09    = 001000     241#
SW1     = 000002     241#
SW10    = 002000     241#
SW11    = 004000     241#
SW12    = 010000     241#
SW13    = 020000     241#
SW14    = 040000     241#
SW15    = 100000     241#
SW2     = 000004     241#
SW3     = 000010     241#
SW4     = 000020     241#
SW5     = 000040     241#
SW6     = 000100     241#
SW7     = 000200     241#
SW8     = 000400     241#
SW9     = 001000     241#
TBITVE= 000014       241#
TCR     023642     2503*   2508*   2521#
TIMPER  024752     1520*   1524*   1597    2648#
TKVEC   = 000060     241#
TMO     = 000010     535#    1534    1571    1893    2191    2195    2316    3056    3063
TMT     = 000016     607#    2446
TPVEC   = 000064     241#
TRAPVE= 000034       241#     703*
TRTVEC= 000014       241#
TST1    003562       764     774#    3117
TST10   005132       858     860#
TST11   005206       860#
TST12   005272       860#
TST13   005346       860     862#
TST14   005556       864#
TST15   005766       866#
TST16   006042       866#
TST17   006126       866#
TST2    003742       795     802#
```

```
TST20   006202            866     868#
TST21   006256            868#
TST22   006342            868#
TST23   006416            868     870#
TST24   006472            870#
TST25   006556            870#
TST26   006632            870     872#
TST27   007042            874#
TST3    004276            854#
TST30   007252            876#
TST31   007462            878#
TST32   007672            880#
TST33   010106            892#
TST34   010634           1008#
TST35   011524           1121#
TST36   012052           1182#
TST37   012646           1281#
TST4    004506            856#
TST40   013174           1338#
TST41   013470           1384#
TST42   013716           1423#
TST43   014212           1469#
TST44   014440           1505#
TST45   014710           1548#
TST46   015136           1585#
TST47   015334           1627#
TST5    004716            858#
TST50   016062           1721#
TST51   016530           1798#
TST52   017002           1847#
TST53   017174           1883#
TST54   017400           1930#
TST55   020162           2030#
TST56   021134           2142#
TST57   022012           2281#
TST6    004772            858#
TST7    005056            858#
TTMG0   003006            744     752#
TTMG1   003040            744     753#
TTMG10  003516            746     760#
TTMG2   003130            744     754#
TTMG3   003216            744     755#
TTMG4   003303            745     756#
TTMG5   003342            745     757#
TTMG6   003414            745     758#
TTMG7   003460            745     759#
TWOSP   042112           2507    3193#
TYPDS = 104405            3117    3121    3128#
TYPE  = 104401             704     716     732     736     738    2379    2393    2407    2502    2507    2510    2526    2531
                          2532    2534    2549    2551    2562    2564    2574    2576    2578    2583    2584    2585    2601
                          2610    2611    2676    2677    2678    2679    2680    2683    2689    2765    2789    2816    2908
                          2911    2925    2926    2952    2963    2983    3000    3005    3117    3120    3121    3123    3124
                          3125    3126    3127    3128#   3129
TYPOC = 104402            2506    2533    2956    2965    3120    3121    3124    3128#
TYPON = 104404            3128#
TYPOS = 104403            2550    2563    2609    2818    2909    3128#
```

```
U10     = 000400            589#  2268
U11     = 001000            588#  2269
U12     = 002000            587#  2270
U13     = 004000            586#  2271
UVITAB    022002           1357   1366   1403   1442   1451   1488   2268#
W       = 000005           1625#
XMITS     031662           3077#  3081   3087   3106*  3110*  3114*
$APTHD    001000            252#
$ASTAT=   ****** U         3130
$ATYC     035740           3130#
$ATY1     035714           3130#
$ATY3     035722           3125   3130#
$ATY4     035732           3120   3130#
$AUTOB    001134            254#   704*  3124
$BASE     001304            254#
$BDADR    001122            254#
$BDDAT    001126            254#   780*   781*   782    809*   810    818*   819    825*   826    832*   833    839*
                            840    854*   856*   862*   864*   872*   874*   876*   878*   880*   915*   917    963*
                            964   1015*  1016   1023*  1024   1033*  1035   1108*  1128*  1129   1138*  1140  1165*
                           1166   1188*  1189   1196*  1197   1206*  1208   1268*  1288*  1289   1298*  1300  1325*
                           1326   1350*  1353   1359*  1360   1369*  1374*  1396*  1399   1405*  1406   1412* 1435*
                           1438   1444*  1445   1454*  1459*  1481*  1484   1490*  1491   1497*  1527*  1528* 1530
                           1535*  1536   1564*  1565*  1567   1572*  1573   1587*  1600*  1601   1609*  1610  1653*
                           1654   1660   1667*  1668   1678*  1679   1685*  1689*  1694*  1695   1749*  1750  1756
                           1762*  1769*  1773*  1778*  1779   1810*  1811   1813   1839*  1840   1861*  1862  1864
                           1875*  1876   1898*  1899   1914*  1915   1981*  1982   1988   1994*  1995   2006* 2007
                           2012*  2017*  2018   2053*  2054   2061*  2062*  2065   2072*  2073*  2075   2084* 2085
                           2108*  2109*  2110   2127*  2128*  2129   2194*  2195   2202*  2203   2209*  2210  2221*
                           2222   2228*  2234*  2235   2318*  2330*  2334   2340*  2345*  2350*  2356*  2357  2362*
                           2367   2371   2373   2383   2386   2388   2397   2400   2402   2418   2424   2722* 2741*
                           2769*  2776*  2827*  2834*  2835   2837   2846*  2855*  2861*  2867*  2877*  2878  2887*
                           2890*  2891   3198   3200
$BELL     001220            254#  3120
$CDW1     001310            254#
$CDW2     001312            254#
$CHARC    034776           3125#*
$CKSWR    033612           3124#  3128
$CMTAG    001100            254#   703
$CM1    = 000006            254#
$CM2    = 000014            254#
$CM3    = 000006            254#
$CM4    = 000007            254#
$CNTLG    034471           3124#
$CNTLU    034464           3124#
$CORE     033352           3122#
$CPUOP    001256            254#
$CRLF     001225            254#   738    2576   2583   2584   2911   3120   3121   3123   3124   3125
$CROUT    033402           3122#
$DBLK     035444           3127#
$DDW0     001314            254#
$DDW1     001316            254#
$DDW10    001340            254#
$DDW11    001342            254#
$DDW12    001344            254#
$DDW13    001346            254#
$DDW14    001350            254#
```

```
$DDW15  001352            254#
$DDW2   001320            254#
$DDW3   001322            254#
$DDW4   001324            254#
$DDW5   001326            254#
$DDW6   001330            254#
$DDW7   001332            254#
$DDW8   001334            254#
$DDW9   001336            254#
$DEVCT  001240            254#
$DEVM   001306            254#
$DOAGN  032150           3117#
$DTBL   035434           3127#
$ENDAD  032140            250     704     714    3117#   3120
$ENDCT  032106            703    3117#
$ENDMG  032157           3117#
$ENULL  032154           3117#
$ENV    001250            254#    704    3120    3125    3130
$ENVM   001251            254#    703    3125    3130
$EOP    032052           2436    3117#
$EOPCT  032100            703*   3117#
$ERFLG  001103            254#   3119*   3120*
$ERMAX  001115            254#    703*   3119*
$ERROR  032446            703    3119    3120#
$ERRPC  001116            254#   3120*   3121    3198    3199    3200
$ERRTB  001354            254#   3121
$ERRTY  032770           3120    3121#
$ERTTL  001112            254#   3120*
$ESCAP  001216            254#    703*   3119*   3120
$ETABL  001250            254#
$ETEND  001354            252     254#
$FATAL  001232            254#   3130*
$FFLG   036160           3130#*
$FILLC  001156            254#   3125
$FILLS  001155            254#   3125
$GDADR  001120            254#
$GDDAT  001124            254#    777*    778     782     786*    787*    788     804*    810     819     826     833     840
                          845*    849*    854*    856*    862*    864*    872*    874*    876*    878*    880*    916*    917
                          959*    964    1018*   1028*   1029*   1034*   1035    1104*   1105    1106*   1133*   1134*   1139*
                         1140    1160*   1161    1162*   1163*   1166    1191*   1201*   1202*   1207*   1208    1264*   1265
                         1266*   1293*   1294*   1299*   1300    1320*   1321    1322*   1323*   1326    1351*   1352*   1353
                         1357*   1360    1364*   1397*   1398*   1399    1403*   1406    1410*   1436*   1437*   1438    1442*
                         1445    1449*   1482*   1483*   1484    1488*   1491    1495*   1529*   1530    1534*   1536    1566*
                         1567    1571*   1573    1595*   1596*   1610    1645*   1646*   1651*   1660    1665*   1668    1673*
                         1674    1675*   1676*   1679    1683*   1693*   1695    1741*   1742*   1747*   1756    1760*   1767*
                         1777*   1779    1807*   1808*   1813    1837*   1840    1858*   1859*   1864    1873*   1876    1890*
                         1891*   1899    1912*   1915    1970*   1971*   1976*   1979*   1988    1992*   1995    2001*   2004*
                         2007    2011*   2016*   2018    2045*   2048*   2051*   2052*   2054    2063*   2064*   2065    2074*
                         2075    2079*   2080*   2085    2104*   2110    2125*   2129    2186*   2191*   2203    2207*   2210
                         2216*   2219*   2222    2226*   2232*   2235    2316*   2331*   2332*   2333*   2334    2338*   2354*
                         2357    2366*   2367    2370*   2371    2382*   2383    2385*   2396*   2397    2399*   2416*   2418
                         2423*   2424    2706*   2767*   2826*   2835*   2836*   2837    2845*   2853*   2872*   2875*   2878
                         2888*   2889*   2891    2896    2959*   2960    2964    2968    3198    3200
$GET42  032130           3117#
$GTSWR  033662           3124#   3128
$HD   = 000000            239
```

```
$HIBTS  001000          252#
$HIOCT  033610          3123#*
$ICNT   001104          254#    3119*
$ILLUP  035676          3129#
$INTAG  001135          254#    3124
$ITEMB  001114          254#    3120*   3121
$KTNEX  033344          3122#
$KTOUT  033334          3122#
$KT11   033202          3122#*
$LF     001226          254#    3120    3123    3124    3125
$LFLG   036157          3130#*
$LPADR  001106          254#     703*   3119*
$LPERR  001110          254#     703*    776*    790*    805*    854*    856*    862*    864*    872*    874*    876*    878*
                        880*     926*   1588*   1629*   1723*   1936*   2148*   2518    2692*   3119*   3120
$LSTAD  033446          3122#*
$LSTBK  033450          3122#*
$MADR1  001262          254#
$MADR2  001266          254#
$MADR3  001272          254#
$MADR4  001276          254#
$MAIL   001230          252     254#    703     704     3119    3120    3125
$MAMS1  001260          254#
$MAMS2  001264          254#
$MAMS3  001270          254#
$MAMS4  001274          254#
$MBADR  001002          252#
$MFLG   036156          3130#*
$MNEW   034507          3124#
$MSGAD  001244          254#    3130*
$MSGLG  001246          254#    3130*
$MSGTY  001230          254#    3130*
$MSWR   034476          3124#
$MTYP1  001261          254#
$MTYP2  001265          254#
$MTYP3  001271          254#
$MTYP4  001275          254#
$MXCNT  032444          3119#
$NULL   001154          254#    3125
$NWTST= 000001          774#    802#    854#    856#    858#    860#    862#    864#    866#    868#    870#    872#    874#
                        876#    878#    880#    892#   1008#   1121#   1182#   1281#   1338#   1384#   1423#   1469#   1505#
                       1548#   1585#   1627#   1721#   1798#   1847#   1883#   1930#   2030#   2142#   2281#
$OCNT   035224          3126#*
$OMODE  035226          3126#*
$OVER   032430          3119#
$PASS   001236          254#     703*   2596    3117*   3119    3198    3199    3200
$PASTM  001006          252#
$POWER  035704          3129#
$PWRDN  035536          703     3129#
$PWRMG  035672          3129#
$PWRUP  035610          3129#
$QUES   001224          254#    3120    3123    3124    3125
$RDCHR  034074          3124#   3128
$RDDEC= ******  U       3128
$RDLIN  034214          3124#   3128
$RDOCT  033452          3123#   3128
$RDSZ = 000014          3124#
```

```
$REGAD  001160            254#
$REG0   001162            254#
$REG1   001164            254#
$REG2   001166            254#
$REG3   001170            254#
$REG4   001172            254#
$REG5   001174            254#
$RTNAD  032152           3117#
$R2A  = ****** U         3128
$SAVRE= ****** U         3128
$SAVR6  035702           3129#*
$SCOPE  032174            703   3119#
$SETUP= 000137            701#   703    704    3117   3119   3120   3124
$SIZE   033144           3122#
$SIZEX  033406           3122#
$STUP = 177777            701#
$SVLAD  032374           3119#
$SVPC = 000210            250#
$SWR  = 167000            231#   239    240    254    703    774    802    854    856    858    860    862    864
                           866    868    870    872    874    876    878    880    892   1008   1121   1182   1281
                          1338   1384   1423   1469   1505   1548   1585   1627   1721   1798   1847   1883   1930
                          2030   2142   2281   3117   3119   3120   3129
$SWREG  001252            254#   703
$SWRMK= 000000           3119
$TESTN  001234            254#  3119*
$TIMES  001214            254#   703*  3117*  3119*
$TKB    001146            254#  2682   2764   2924   2982   2999   3124
$TKS    001144            254#  2782   3044   3059   3090   3124*
$TMP0   001176            254#   900*   913*   914    938*   941    960   2575*  2577   2580*  2581   2705*  3200
$TMP1   001200            254#   962*  1634*  1636*  1637*  1638*  1639*  1640*  1642   1730*  1732*  1733*  1734*
                         1735*  1736*  1738   1943*  1954*  1957   1960*  1961*  1962*  1963*  1964*  1966   1977
                         1999   2002   2032*  2037*  2070   2159*  2170*  2173   2176*  2177*  2178*  2179*  2180*
                         2182   2214   2217   2720*  2739*  3200
$TMP2   001202            254#  1060*  1233*  2033*  2042*  2062   2073   2082   2109
$TMP3   001204            254#   927*  1586*  1594   1595   1601   1614*
$TMP4   001206            254#
$TMP5   001210            254#
$TMP6   001212            254#  1081*  1082*  1083   1085   1092*  1093*  1094*  1095*  1096*  1097
$TN   = 000060            232#   239    774#   795    802#   854#   856#   858#   860#   862#   864#   866#   868#
                          870#   872#   874#   876#   878#   880#   892#  1008#  1121#  1182#  1281#  1338#  1384#
                         1423#  1469#  1505#  1548#  1585#  1627#  1721#  1798#  1847#  1883#  1930#  2030#  2142#
                         2281#
$TPB    001152            254#  3125*
$TPFLG  001157            254#  3125
$TPS    001150            254#  3125
$TRAP   035454             703   3128#
$TRAP2  035476           3128#
$TRP  = 000013           3128#
$TRPAD  035510           3128#
$TSTM   001004            252#
$TSTNM  001102            254#  3117*  3119*  3120
$TTYIN  034450           3124#
$TYPBN= ****** U         3128
$TYPDS  035230           3127#  3128
$TYPE   034520           3125#  3128   3130
$TYPEC  034732           3124   3125#
```

```
$TYPEX  035000          3125#
$TYPOC  035026          3126#   3128
$TYPON  035042          3126#   3128
$TYPOS  035002          3126#   3128
$UNIT   001242           254#
$UNITM  001010           252#
$USWR   001254           254#
$VECT1  001300           254#
$VECT2  001302           254#
$XTSTR  032220          3119#
$$GET4= 000000          3117#
$OFILL  035225          3126#*
$40CAT  000000           245#   3119   3120
.       = 042212         245#    250#   251#   252#   254#   703   704#   748   761#  1933  2146  2329  2364
                        2368   2377   2407#  2526#  2529   2585#  2589   2650#  2673   2676#  2677#  2679#  2683#
                        2689#  2718   2789#  2908#  2952#  2963#  2997#  3000#  3005#  3060   3117   3119   3120
                        3121#  3123   3124#  3125   3127#  3129   3130#  3196#
.$ASTA= ****** U        3130
.$X     = 001000         252#
```

```
BITST    617#   858   860   866   868   870
COMMEN   241#
ENDCOM   241#
ERROR    241#   784   796   812   821   828   835   842   854   856   858   860   862   864   866
          868   870   872   874   876   878   880   919   966  1019  1030  1037  1042  1047  1052
         1065  1076  1088  1099  1110  1135  1142  1147  1152  1157  1168  1192  1203  1210  1215
         1220  1225  1238  1249  1257  1270  1295  1302  1307  1312  1317  1328  1355  1362  1371
         1376  1401  1408  1414  1440  1447  1456  1461  1486  1493  1499  1517  1532  1538  1544
         1560  1569  1575  1581  1605  1612  1658  1662  1670  1681  1687  1691  1697  1754  1758
         1764  1771  1775  1781  1815  1829  1842  1866  1878  1901  1917  1986  1990  1997  2009
         2014  2020  2056  2067  2077  2087  2098  2112  2131  2199  2205  2212  2224  2230  2237
         2319  2336  2342  2347  2352  2359  2411  2420  2426  2456  2723  2742  2771  2778  2829
         2839  2848  2857  2863  2869  2880  2893  2898  2940
ESCAPE   241#
GETPRI   241#  3122
GETSWR   241#   704#
INCTST   663#   854   856   862   864   872   874   876   878   880
MULT     241#
NEWTST   241#   774   802   854   856   858   860   862   864   866   868   870   872   874   876
          878   880   892  1008  1121  1182  1281  1338  1384  1423  1469  1505  1548  1585  1627
         1721  1798  1847  1883  1930  2030  2142  2281
POP      241#  2511  3123  3127  3129  3130
PUSH     241#  2504  3123  3127  3129  3130
REPORT   241#
SCOPE    241#   774   802   854   856   858   860   862   864   866   868   870   872   874   876
          878   880   892  1008  1121  1182  1281  1338  1384  1423  1469  1505  1548  1585  1627
         1721  1798  1847  1883  1930  2030  2142  2281  2670  2753  2921  2949  2976  3117
SETPRI   241#
SETTRA  3128#
SETUP    241#   703
SKIP     241#   795   858   860   866   868   870
SLASH    241#
SPACE    241#
STARS    241#   250   252   254   774   802   854   856   858   860   862   864   866   868   870
          872   874   876   878   880   892  1008  1121  1182  1281  1338  1384  1423  1469  1505
         1548  1585  1627  1721  1798  1847  1883  1930  2030  2142  2281  3117  3119  3120  3121
         3122  3123  3124  3125  3126  3127  3128  3129  3130
SWRSU    241#   703#
TRMTRP  3128#
TYPBIN   241#
TYPDEC   241#  3117  3121
TYPNAM   241#   704
TYPNUM   241#
TYPOCS   241#  2550  2563  2609
TYPOCT   241#  2506  2533  3120  3121  3124
TYPTXT   241#   716   732  2379  2393  2407  2526  2531  2574  2585  2676  2677  2678  2679  2680
         2683  2689  2765  2789  2816  2908  2925  2926  2952  2963  3000  3005
$$CMRE   254#
$$CMTM   254#
$$ESCA   241#
$$NEWT   241#   774   802   854   856   858   860   862   864   866   868   870   872   874   876
          878   880   892  1008  1121  1182  1281  1338  1384  1423  1469  1505  1548  1585  1627
         1721  1798  1847  1883  1930  2030  2142  2281
$$SET   3128#
$$SETM   703#
$$SKIP   241#   795   858   860   866   868   870
```

```
.EQUAT    235#    241
.HEADE    237#    239
.KT11     236#    244
.SETUP    237#    701
.SWRHI    235#    240
.SWRLO    240#
.$ACT1    238#    250
.$APTB    238#    254#
.$APTH    238#    252
.$APTY    238#   3130
.$CMTA    235#    254
.$EOP     235#   3117
.$ERRO    235#   3120
.$ERRT    235#   3121
.$POWE    236#   3129
.$RDOC    237#   3123
.$READ    237#   3124
.$SCOP    236#   3119
.$SIZE    236#   3122
.$TRAP    236#   3128
.$TYPD    236#   3127
.$TYPE    237#   3125
.$TYPO    236#   3126
.$40CA    235#    245

. ABS.  042212     000

 ERRORS DETECTED:  0

 CRIIAB,CRIIAB/CRF:CRIIAB.P11
 RUN-TIME: 128 92 TO SECONDS
 RUN-TIME RATIO: 1050/232=4.5
 CORE USED:  28K  (55 PAGES)
```