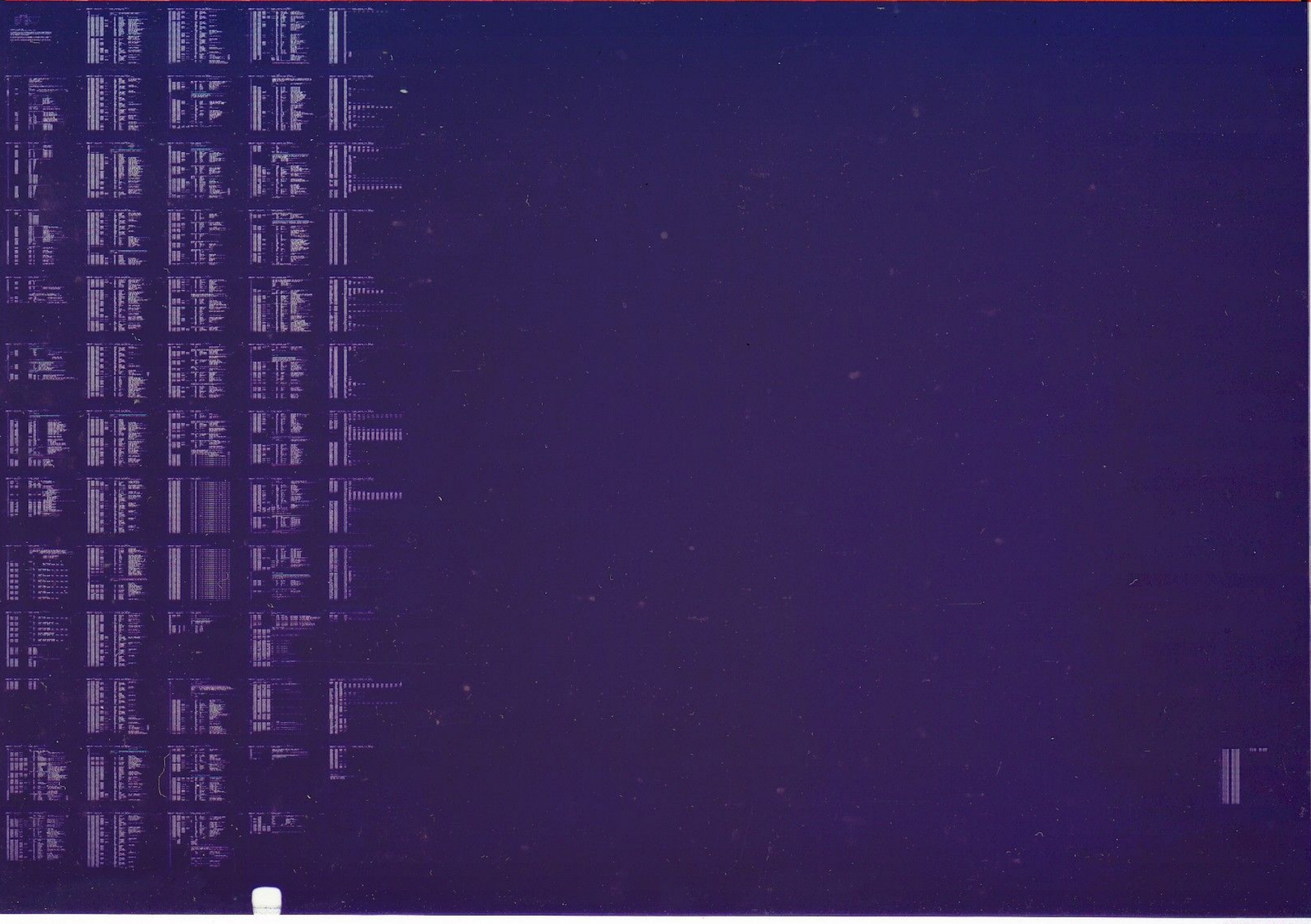
DRV11-J

DIAG TST PRT2
CNDRDA0

AH-T446A-MC
FICHE 1 OF 1



IDENTIFICATION

PRODUCT NAME: CNDRDA0 DRV11J DIAG TST PRT2

PRODUCT CODE: AC-T445A-MC

PRODUCT DATE: DECEMBER,1982

MAINTAINER: DIAGNOSTICS SERVICES/ISS

AUTHOR: W.HEAVEY

```
                                ;;GPA   .HEADER ^/CNDRDA DRV11J DIAG TST PRT2 /,1982,^/BILL HEAVEY/
                                .TITLE  CNDRDA DRV11J DIAG TST PRT2
                                ;*COPYRIGHT (C) 1982
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY BILL HEAVEY
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
                                ;* EDITED TO PERMIT DRV'S TO RUN ON FALCON (KXT11).              ;;GPA
                                ;*              G. PASQUANTONIO, JULY '81                      ;;GPA
                                ;*
165400                          $SWR=165400
000001                          $TN=1
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        11            INHIBIT ITERATIONS
                                ;*         9            LOOP ON ERROR
                                ;*         8            LOOP ON TEST IN SWR<7:0>
                                .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
                                ;*****  THE FOLLOWING ODT START ADDRESS FOR SBC 11/21 IS ADDED
170000                          ODTST=  170000
                                ;*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
```

```
(1)          000007                  R7=     %7              ;;GENERAL REGISTER
(1)          000006                  SP=     %6              ;;STACK POINTER
(1)          000007                  PC=     %7              ;;PROGRAM COUNTER
(1)
(1)                                  ;*PRIORITY LEVEL DEFINITIONS
(1)          000000                  PR0=    0               ;;PRIORITY LEVEL 0
(1)          000040                  PR1=    40              ;;PRIORITY LEVEL 1
(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)                                  ;*****  THE FOLLOWING BREAK VECTOR AND LINE CLOCK VECTOR ARE INCLUDED
(1)          000100                  LKVEC=  100             ;;LINE CLOCK VECTOR
(1)          000140                  BRKVEC= 140             ;;BREAK VECTOR
(1)          000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
5694         174160                  ABASE=  174160          ;BASE ADDRESS
5695         000001                  ADEVM=  1               ;DEFAULT TO ONE DRV11J
5696         100000                  RDY=    BIT15
5697         000400                  DIR=    BIT8
5698         001000                  IE=     BIT9
5699
5700                                 ;CHIP COMMAND SUMMARY
5701         000020                  CIRMR=  20              ;CLEAR IRR AND IMR
5702         000030                  CSIRMR= 30              ;CLEAR SINGLE IRR AND IMR BIT
5703
5704         000040                  CIMR=   40              ;CLEAR IMR
5705         000050                  CSIMR=  50              ;CLEAR SINGLE IMR BIT
5706         000060                  SIMR=   60              ;SET ALL IMR BITS
5707         000070                  SSIMR=  70              ;SET SINGLE IMR BITS
5708
5709         000100                  CIRR=   100             ;CLEAR IRR
5710         000110                  CSIRR=  110             ;CLEAR SINGLE IRR BITS
5711         000120                  SIRR=   120             ;SET ALL IRR BITS
5712         000130                  SSIRR=  130             ;SET SINGLE IRR BITS
5713
5714         000140                  CHPISR= 140             ;CLEAR HIGHEST PRIORITY ISR BIT
5715         000160                  CISR=   160             ;CLEAR ISR
5716         000170                  CSISR=  170             ;CLEAR SINGLE ISR BIT
5717
5718         000200                  LMD04=  200             ;LOAD MODE BITS M0-M4
5719         000240                  LMD57=  240             ;LOAD MODE BITS M5-M7
```

```
                                ;CHIP MODE BIT PRESELECTION
            000240              MISR=   240
            000244              MIMR=   244
            000250              MIRR=   250
            000254              MACR=   254

                                ;CHIP WRITE PRESELECTION
            000300              PACR=   300             ;PRESELECT AUTO CLEAR REG. FOR WRITING
            000260              PIMR=   260             ;PRESELECT IMR REG. FOR WRITING
            000340              PVMA=   340             ;PRESELECT VECTOR MEMORY ADDRESS

                                .SBTTL  TRAP CATCHER
  (1)
  (1)       000000                    .=0
  (1)                           ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
  (1)                           ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
  (1)                           ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
  (1)       000174              .=174
  (1) 000174  000000            DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
  (1) 000176  000000            SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
  (1)                           .SBTTL  STARTING ADDRESS(ES)
  (1) 000200  000137  002476            JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
            000100              .=100
  000100  000104  000200  000002       .WORD   104,200,2       ;IF 'B EVENT' ON Q BUS IS CONNECTED
                                                                ;IGNORE IT'S INTERRUPT - JUST DO A RTI
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
000106                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  010712                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000106                                  .=$SVPC                 ;; RESTORE PC
002000                                  .=2000

                                                                ;LONGEST TEST TIME
                                                                ;1ST PASS RUN TIME
                                                                ;ADDITIONAL RUN TIME
                                .SBTTL  APT PARAMETER BLOCK

                                ;;******************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;******************************************************************
002000                                  .$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  002000                          $APTHDR ;;POINT TO APT HEADER BLOCK
002000                                  .=.$X   ;;RESET LOCATION COUNTER
                                ;;******************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

002000                          $APTHD:
002000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
002002  002170                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
002004  000014                  $TSTM:  .WORD   12.     ;;RUN TIM OF LONGEST TEST
002006  000036                  $PASTM: .WORD   30.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
002010  000036                  $UNITM: .WORD   30.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
002012  000031                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
 5743                                   .SBTTL  COMMON TAGS
  (1)
  (2)                                   ;;**********************************************************************
  (1)                                   ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
  (1)                                   ;*USED IN THE PROGRAM.
  (1)
  (1)           002100                          .=2100
  (1)  002100                           $CMTAG:                         ;;START OF COMMON TAGS
  (1)  002100  000000                           .WORD   0
  (1)  002102     000                   $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
  (1)  002103     000                   $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
  (1)  002104  000000                   $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
  (1)  002106  000000                   $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
  (1)  002110  000000                   $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
  (1)  002112  000000                   $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
  (1)  002114     000                   $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
  (1)  002115     001                   $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
  (1)  002116  000000                   $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
  (1)  002120  000000                   $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
  (1)  002122  000000                   $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
  (1)  002124  000000                   $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
  (1)  002126  000000                   $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
  (1)  002130  000000                           .WORD   0               ;;RESERVED--NOT TO BE USED
  (1)  002132  000000                           .WORD   0
  (1)  002134     000                   $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
  (1)  002135     000                   $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
  (1)  002136  000000                           .WORD   0
  (1)  002140  177570                   SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
  (1)  002142  177570                   DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
  (1)  002144  177560                   $TKS:   177560                  ;;TTY KBD STATUS
  (1)  002146  177562                   $TKB:   177562                  ;;TTY KBD BUFFER
  (1)  002150  177564                   $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
  (1)  002152  177566                   $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
  (1)  002154     000                   $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
  (1)  002155     002                   $FILLS: .BYTE   2
  (1)                                           ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
  (1)  002156     012                   $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
  (1)  002157     000                   $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
  (1)  002160  000000                   $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
  (1)  002162  000000                   $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
  (1)  002164     077                   $QUES:  .ASCII  /?/             ;;QUESTION MARK
  (1)  002165     015                   $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
  (1)  002166  000012                   $LF:    .ASCIZ  <12>            ;;LINE FEED
  (2)                                   ;;**********************************************************************
  (2)                                   .SBTTL  APT MAILBOX-ETABLE
  (2)
  (3)                                   ;;**********************************************************************
  (2)                                   .EVEN
  (2)  002170                           $MAIL:                          ;;APT MAILBOX
  (2)  002170  000000                   $MSGTY: .WORD   AMSGTY           ;;MESSAGE TYPE CODE
  (2)  002172  000000                   $FATAL: .WORD   AFATAL           ;;FATAL ERROR NUMBER
  (2)  002174  000000                   $TESTN: .WORD   ATESTN           ;;TEST NUMBER
  (2)  002176  000000                   $PASS:  .WORD   APASS            ;;PASS COUNT
  (2)  002200  000000                   $DEVCT: .WORD   ADEVCT           ;;DEVICE COUNT
  (2)  002202  000000                   $UNIT:  .WORD   AUNIT            ;;I/O UNIT NUMBER
  (2)  002204  000000                   $MSGAD: .WORD   AMSGAD           ;;MESSAGE ADDRESS
```

```
(2)  002206  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
(2)  002210                         $ETABLE:                ;;APT ENVIRONMENT TABLE
(2)  002210     000                 $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
(2)  002211     000                 $ENVM:  .BYTE   AENVM
(2)                                 ;;ENVIRONMENT MODE BITS
(2)  002212  000000                 $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
(2)  002214  000000                 $USWR:  .WORD   AUSWR   ;;USER SWITCHES
(2)  002216  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)  002220     000                 $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
(2)  002221     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)  002222  000000                 $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
(2)                                 ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
(2)  002224     000                 $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
(2)  002225     000                 $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
(2)  002226  000000                 $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
(2)  002230     000                 $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
(2)  002231     000                 $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
(2)  002232  000000                 $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
(2)  002234     000                 $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
(2)  002235     000                 $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
(2)  002236  000000                 $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
(2)  002240  000000                 $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)  002242  000000                 $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(2)  002244  174160                 $BASE:  .WORD   ABASE
(2)                                 ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)  002246  000001                 $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
(2)  002250  000000                 $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
(2)  002252                         $ETEND:
(2)                                 .MEXIT
```

```
                                                                J 1
   (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)  002252                  $ERRTB:
5744                            ;ERROR  1
5745  002252  015751                   EM1             ;REG TIMEOUT ER
5746  002254  016266                   DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
5747  002256  016536                   DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
5748  002260  000000                   0
5749
5750                            ;ERROR  2
5751  002262  015770                   EM2     ;REG READ/WRITE ER
5752  002264  016333                   DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
5753  002266  016552                   DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
5754  002270  000000                   0
5755
5756                            ;ERROR  3
5757  002272  016012                   EM3     ;IRR REG ER
5758  002274  016333                   DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
5759  002276  016552                   DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
5760  002300  000000                   0
5761
5762                            ;ERROR  4
5763  002302  016025                   EM4     ;ACR REG ER
5764  002304  016333                   DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
5765  002306  016552                   DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
5766  002310  000000                   0
5767
5768                            ;ERROR  5
5769  002312  016040                   EM5     ;IMR REG ERROR
5770  002314  016333                   DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
5771  002316  016552                   DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
5772  002320  000000                   0
5773
5774                            ;ERROR  6
5775  002322  016053                   EM6     ;ISR REG ERROR
5776  002324  016333                   DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
5777  002326  016552                   DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
5778  002330  000000                   0
5779
5780                            ;ERROR  7
5781  002332  016245                   EM14    ;VECTOR ADDR MEM ER
5782  002334  016266                   DH1     ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
5783  002336  016536                   DT1     ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
5784  002340  000000                   0
```

```
                                        ;ERROR  10
002342  016121                                EM10    ;CHIP STAT ER
002344  016333                                DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002346  016552                                DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002350  000000                                0

                                        ;ERROR  11
002352  016136                                EM11    ;ILLEGAL INTERRUPT  RECEIVED
002354  016420                                DH3     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR
002356  016572                                DT3     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC
002360  000000                                0

                                        ;ERROR  12
002362  016166                                EM12    ;INTERRUPT TEST ERROR
002364  016333                                DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002366  016552                                DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002370  000000                                0

                                        ;ERROR  13      ;MULTIPLE INTERRUPTS RECEIVED
002372  016213                                EM13    ;MULTIPLE INTERRUPTS RECEIVED
002374  016333                                DH2     ;ERRPC  TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD
002376  016552                                DT2     ;$ERRPC TSTNUM  $BDADR  VECVAL  VECLOC  $GDDAT  $BDDAT
002400  000000                                0

                                        ;ERROR  14
002402  016066                                EM7     ;ILLEGAL VECTOR ADDR MEM ERROR
002404  016465                                DH4     ;ERRPC  TSTNUM  TESTPC  BUSADR  VAM     ADDR
002406  016606                                DT4     ;$ERRPC TSTNUM  $GDADR  $BDADR  VECVAL  VECLOC
002410  000000                                0


                                        ; BUS REGISTER ADDRESS POINTERS


002412  174160                          DRCSA:  ABASE
002414  174162                          DRDBA:  ABASE+2
002416  174164                          DRCSB:  ABASE+4
002420  174166                          DRDBB:  ABASE+6
002422  174170                          DRCSC:  ABASE+10
002424  174172                          DRDBC:  ABASE+12
002426  174174                          DRCSD:  ABASE+14
002430  174176                          DRDBD:  ABASE+16


                                        ;COMMON PROGRAM LOCATION(S)

002432  000000                          TSTNUM: 0               ;CONTAINS TEST NUMBER ON ERROR
002434  000001                          DMAP:   1
002436  000000                          INTFLG: .WORD   0
002440  000000                          XXDP:   .WORD   0
002442  000000                          IMRLOC: .WORD   0
002444  000000                          ISRLOC: .WORD   0
002446  000000                          IRRLOC: .WORD   0
002450  000000                          ACRLOC: .WORD   0
002452  000000                          VECLOC: .WORD   0
```

```
                                                                L 1
 5841  002454  000000                VECPAT: .WORD   0
 5842  002456  000000                VECVAL: .WORD   0
 5843  002460  000000                SWRSAV: .WORD   0
 5844  002462  000000                GRPCNT: .WORD   0
 5845  002464  000000                BDSAV:  .WORD   0
 5846  002466  000000                LVLCNT: .WORD   0
 5847  002470  000000                BYCNT:  .WORD   0
 5848  002472  000000                BYNUM:  .WORD   0
 5849  002474  000000                LVLSAV: .WORD   0
```

```
                                        .SBTTL  PROGRAM START
002476                          START:
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
002476  012706  002100                MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
002502  005026                        CLR     (R6)+           ;;CLEAR MEMORY LOCATION
002504  022706  002140                CMP     #SWR,R6 ;;DONE?
002510  001374                        BNE     .-6             ;;LOOP BACK IF NO
002512  012706  002100                MOV     #2100,SP                ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
002516  012737  014342  000020        MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
002524  012737  000300  000022        MOV     #PR6,@#IOTVEC+2 ;;LEVEL 6
002532  012737  014014  000030        MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
002540  012737  000300  000032        MOV     #PR6,@#EMTVEC+2 ;;LEVEL 6
                                ;;BIT02
002546  012737  015602  000034        MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002554  012737  000300  000036        MOV     #PR6,@#TRAPVEC+2;LEVEL 6
002562  012737  015376  000024        MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
002570  012737  000300  000026        MOV     #PR6,@#PWRVEC+2 ;;LEVEL 6
002576  005037  002160                CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
002602  005037  002162                CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002606  112737  000001  002115        MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
002614  012737  002614  002106        MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
002622  012737  002622  002110        MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
002630  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002634  012737  002670  000004        MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002642  012737  177570  002140        MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002650  012737  177570  002142        MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002656  022777  177777  177254        CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002664  001012                        BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
002666  000403                        BR      65$             ;;BRANCH IF NO TIMEOUT
002670  012716  002676        64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002674  000002                        RTI
002676  012737  000176  002140  65$:  MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002704  012737  000174  002142        MOV     #DISPREG,DISPLAY
002712  012637  000004        66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002716  005037  002176                CLR     $PASS           ;;CLEAR PASS COUNT
002722  132737  000200  002211        BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002730  001403                        BEQ     67$             ;;YES,USE NON-APT SWITCH
002732  012737  002212  002140        MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002740                          67$:
002740  005037  002172                CLR     $FATAL          ;CLEAR ERROR NUMBER
002744  005037  002170                CLR     $MSGTYP         ;CLEAR MESSAGE TYPE
002750  005037  002174                CLR     $TESTN          ;CLEAR TEST NUMBER
                                ;;THE FOLLOWING LINES OF CODE WERE DELETED IN ORDER TO MAKE
                                ;;THIS DIAGNOSTIC SPECIFIC TO FALCON (11/21).
                                        ;CALL   FALCON          ; CHECK FOR FALCON (KXT11)      ;;GPA
                                        ;BEQ    1000$           ; BR IF NOT KXT11 SYSTEM        ;;GPA
                                        ;BIC    #40,@#22        ; FALCON, LOWER IOT...          ;;GPA
                                        ;BIC    #40,@#32        ;...EMT...                      ;;GPA
                                        ;BIC    #40,@#36        ;...AND TRAP TO LEVEL 6.        ;;GPA
                                        ;CMP    $BASE,#ABASE    ; IS $BASE VIRGIN ??            ;;GPA
```

```
                                        ;BNE    1000$           ; BR IF NOT.                          ;;GPA
                                        ;MOV    #174160,$BASE   ; YES, USE ENGINEERING DEFAULT        ;;GPA
                                ;1000$:                                                               ;;GPA
                                ;CHECK OPERATING ENVIRONMENT
002754  005737  000042                  TST     @#42            ;ARE WE IN ACT/XXDP AUTO MODE?
002760  001410                          BEQ     1$              ;BRANCH IF NO
002762  023737  000042  000046          CMP     @#42,@#46       ;IS IT ACT AUTO MODE?
002770  001410                          BEQ     2$              ;BRANCH IF YES
002772  012737  177777  002440          MOV     #-1,XXDP        ;SET XXDP CHAIN MODE INDICATOR
003000  000404                          BR      2$
003002  123727  002210  000001  1$:     CMPB    $ENV,#1         ;ARE WE IN APT AUTO MODE?
003010  001003                          BNE     3$              ;BRANCH IF NO
003012  112737  000001  002134  2$:     MOVB    #1,$AUTOB       ;SET AUTO MODE INDICATOR

                                ;PRINT TITLE IF NOT IN ACT OR APT AUTO MODE
003020  005227  177777          3$:     INC     #-1             ;FIRST TIME?
003024  001014                          BNE     5$              ;SKIP TITLE IF NO
003026  004537  011306                  JSR     R5,TOBUF        ;STORE 0-202 INTO BUFFER AREA
003032  005737  002134                  TST     $AUTOB          ;ARE WE IN AUTO MODE?
003036  001403                          BEQ     4$              ;BRANCH TO TITLE TYPEOUT IF NOT
003040  005737  002440                  TST     XXDP            ;IS THE AUTO MODE UNDER XXDP?
003044  001404                          BEQ     5$              ;SKIP TITLE IF NOT
003046  104401  015662          4$:     TYPE    ,TITLED         ;PRINT OUT THE TITLE
003052  104401  015724                  TYPE    ,TLCABL         ;PRINT DRV11J CABLE REQ'D

                                ;GET THE VALUE IN THE SOFTWARE SWITCH REGISTER
003056  005737  002134          5$:     TST     $AUTOB          ;ARE WE IN AUTOMATIC MODE?
003062  001001                          BNE     START1          ;BRANCH IF YES
003064  104406                          GTSWR                   ;ASK FOR SWR INPUT FROM CONSOLE
003066  005037  002202          START1: CLR     $UNIT           ;CLEAR UNIT NUMBER
003072  013737  002246  002434          MOV     $DEVM,DMAP      ;UP TO 4 DRV11J'S
003100  042737  177760  002434          BIC     #177760,DMAP    ;POSITION OF DRV11J'S
003106  013701  002244                  MOV     $BASE,R1        ;GET BASE ADDRESS
003112  010137  002412                  MOV     R1,DRCSA        ;MAKE BUS REG.POINTER = BASE
003116  032737  000001  002434          BIT     #1,DMAP         ;IS FIRST DRV11-J SELECTED
003124  001002                          BNE     NEXPAS          ;YES
003126  000137  010570                  JMP     NXDEV1          ;NO,ADVANCE BASE DRV11J ADDRESS
003132  012700  002412          NEXPAS: MOV     #DRCSA,R0       ;SET UP REGISTER ADDRESS POINTERS
003136  010120                  NEXPA1: MOV     R1,(R0)+        ;LOAD EM
003140  062701  000002                  ADD     #2,R1           ;
003144  022700  002432                  CMP     #DRDBD+2,R0     ;ALL DONE?
003150  001372                          BNE     NEXPA1          ;BR IF NOT
003152  004537  011456                  JSR     R5,TRPCAT       ;EXTEND TRAP CATCHER 204-1774
003156  012706  002100                  MOV     #2100,SP        ;ALWAYS RESET STACK
003162  013737  002202  002200          MOV     $UNIT,$DEVCT    ;LOAD APT COUNTER WITH UNIT #
003170  123727  002210  000001          CMPB    $ENV,#1         ;ARE WE IN APT MODE?
003176  001401                          BEQ     NEXPA2          ;BR IF YES,SKIP RESET
003200  000005                          RESET                   ;INITIALIZE
003202  106427  000300          NEXPA2: MTPS    #PR6            ;VER:0
```

```
                                ;;*****************************************************************
                                ;*TEST 1        TEST CHIP INTERRUPT ,GROUP 1,GROUP 2
                                ;;*****************************************************************
003206  000004                  TST1:   SCOPE
003210  013700  002412                  MOV     DRCSA,R0        ;START WITH GROUP 1
003214  013701  002416                  MOV     DRCSB,R1
003220  013702  002412                  MOV     DRCSA,R2        ;REGISTER IS COMMON FOR BOTH GROUPS
003224  012737  003276  002110          MOV     #111$,$LPERR    ;SET UP LOOP RETURN
003232  012737  000002  002462          MOV     #2,GRPCNT       ;COUNTER FOR BOTH GROUPS
003240  004537  010746          120$:   JSR     R5,CLRCSR       ;CLEAR ALL CSRS
003244  012703  012570                  MOV     #BGCHP3,R3      ;EXPECTED GDDAT FOR ISR,IMR
003250  012737  000050  002442          MOV     #CSIMR,IMRLOC   ;STORE CLEAR SINGLE IMR CODE
003256  004537  011026                  JSR     R5,CLRIRR       ;CLEAR IRR REGS
003262  012737  011576  000300          MOV     #INTSR1,300     ;STORE VECTOR ROUTINE
003270  012737  000300  000302          MOV     #300,302        ;STORE PSW ;VER:0
003276  005037  002436          111$:   CLR     INTFLG          ;CLEAR INT. FLAG
003302  012706  002100                  MOV     #2100,SP        ;INIT STACK IN CASE OF ILLEGAL INT. LOOP
003306  112710  000340                  MOVB    #PVMA,(R0)      ;PRESELECT VECTOR MEM ADDR
003312  112711  000060                  MOVB    #60,(R1)        ;WRITE VECTOR 300,BY0,LV0
003316  010037  002122                  MOV     R0,$BDADR       ;SAVE BUS ADDRESS
003322  112710  000060                  MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
003326  113710  002442                  MOVB    IMRLOC,(R0)     ;CLEAR MASK ON SINGLE BIT
003332  112710  000202                  MOVB    #202,(R0)       ;LMD04 - COMMON VECTOR
003336  106427  000000                  MTPS    #PR0
003342  112712  000241                  MOVB    #241,(R2)       ;USED IN TEST OF GROUP 2
                                                                ;ARM GROUP 1 TO PASS
                                                                ;ENABLE TO GROUP 2.
003346  112710  000241                  MOVB    #241,(R0)       ;LMD57 - ARM THE CHIP
003352  005737  002436                  TST     INTFLG          ;CHECK FOR INTERRUPTS
003356  001401                          BEQ     1$
003360  104011                          ERROR   11              ;ERROR,ILLEGAL INTERRUPT
003362  005037  002436          1$:     CLR     INTFLG          ;RESET INT. COUNTER
003366  112710  000120                  MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
003372  106427  000300                  MTPS    #PR6    ;VER:0
003376  005737  002436                  TST     INTFLG          ;CHECK FOR NO INTERRUPTS
003402  001401                          BEQ     2$
003404  104011                          ERROR   11              ;ILLEGAL INTERRUPTS
003406  005037  002436          2$:     CLR     INTFLG          ;CLEAR INT. COUNTER
003412  106427  000000                  MTPS    #PR0
003416  152762  000002  000001          BISB    #BIT1,1(R2)     ;SET I/E,EXPECT INTERRUPT
003424  012737  000001  002124          MOV     #1,$GDDAT       ;EXPECT ONE INTERRUPT
003432  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SHOULD HAVE ONE INTERRUPT
003440  023737  002124  002126          CMP     $GDDAT,$BDDAT
003446  001401                          BEQ     3$
003450  104012                          ERROR   12              ;INTERRUPT TEST ERROR
003452  106427  000300          3$:     MTPS    #PR6    ;VER:0
003456  012737  101710  002124          MOV     #101710,$GDDAT  ;RDY,DIR,I/E, CHIP - 31X
003464  010237  002122                  MOV     R2,$BDADR       ;CSRA ADDRESS
003470  011237  002126                  MOV     (R2),$BDDAT
003474  042737  000007  002126          BIC     #7,$BDDAT       ;CLEAR UNDEFINED BITS
003502  023737  002124  002126          CMP     $GDDAT,$BDDAT
003510  001401                          BEQ     4$
003512  104002                          ERROR   2               ;CSRA REG ERROR
003514  010137  002122          4$:     MOV     R1,$BDADR       ;READY TO READ ISR REG.
```

```
 5968  003520  005037  002126                  CLR     $BDDAT          ;SET UP FOR BYTE READS
 5969  003524  005037  002124                  CLR     $GDDAT          ;SET UP FOR BYTE EXPECTED
 5970  003530  111337  002124                  MOVB    (R3),$GDDAT     ;EXPECTED DATA
 5971  003534  112710  000240                  MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
 5972  003540  111137  002126                  MOVB    (R1),$BDDAT
 5973  003544  023737  002124  002126          CMP     $GDDAT,$BDDAT
 5974  003552  001401                          BEQ     5$
 5975  003554  104006                          ERROR   6               ;ISR ERROR
 5976  003556  116337  000001  002124  5$:     MOVB    1(R3),$GDDAT    ;STORE EXPECTED
 5977  003564  112710  000244                  MOVB    #MIMR,(R0)      ;LOAD MODE FOR IMR
 5978  003570  111137  002126                  MOVB    (R1),$BDDAT
 5979  003574  023737  002124  002126          CMP     $GDDAT,$BDDAT
 5980  003602  001401                          BEQ     6$
 5981  003604  104005                          ERROR   5               ;IMR ERROR
 5982  003606  116337  000001  002124  6$:     MOVB    1(R3),$GDDAT    ;EXPECTED IRR
 5983  003614  112710  000250                  MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
 5984  003620  111137  002126                  MOVB    (R1),$BDDAT
 5985  003624  023737  002124  002126          CMP     $GDDAT,$BDDAT
 5986  003632  001401                          BEQ     7$
 5987  003634  104003                          ERROR   3               ;IRR ERROR
 5988  003636  005037  002124          7$:     CLR     $GDDAT
 5989  003642  112710  000254                  MOVB    #MACR,(R0)
 5990  003646  111137  002126                  MOVB    (R1),$BDDAT
 5991  003652  023737  002124  002126          CMP     $GDDAT,$BDDAT
 5992  003660  001401                          BEQ     10$
 5993  003662  104004                          ERROR   4               ;ACR ERROR
 5994  003664  005037  002124          10$:    CLR     $GDDAT
 5995  003670  112710  000140                  MOVB    #CHPISR,(R0)    ;CLEAR HIGHEST PRIOR ISR
 5996  003674  112710  000240                  MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
 5997  003700  111137  002126                  MOVB    (R1),$BDDAT
 5998  003704  023737  002124  002126          CMP     $GDDAT,$BDDAT
 5999  003712  001401                          BEQ     11$
 6000  003714  104006                          ERROR   6               ;ISR REG ERROR
 6001  003716  012737  101710  002124  11$:    MOV     #101710,$GDDAT  ;RDY,DIR,I/E, CHIP - 31X
 6002  003724  010237  002122                  MOV     R2,$BDADR       ;CSRA ADDRESS
 6003  003730  011237  002126                  MOV     (R2),$BDDAT
 6004  003734  042737  000007  002126          BIC     #7,$BDDAT       ;CLEAR UNDEFINED BITS
 6005  003742  023737  002124  002126          CMP     $GDDAT,$BDDAT
 6006  003750  001401                          BEQ     12$
 6007  003752  104002                          ERROR   2               ;CSRA REG ERROR
 6008  003754  012737  100710  002124  12$:    MOV     #100710,$GDDAT  ;RDY,DIR, CHIP - 31X
 6009  003762  042712  001000                  BIC     #BIT9,(R2)      ;CLEAR I/E
 6010  003766  011237  002126                  MOV     (R2),$BDDAT     ;READ CSRA
 6011  003772  042737  000007  002126          BIC     #7,$BDDAT       ;CLEAR UNDEFINED BITS
 6012  004000  023737  002124  002126          CMP     $GDDAT,$BDDAT
 6013  004006  001401                          BEQ     13$
 6014  004010  104002                          ERROR   2               ;CSRA REG ERROR
 6015  004012  005723                  13$:    TST     (R3)+           ;GET NEXT ISR,IMR EXPECTED RESULTS
 6016  004014  020327  012610                  CMP     R3,#EDCHP3      ;CHECK FOR END
 6017  004020  001404                          BEQ     14$
 6018  004022  005237  002442                  INC     IMRLOC          ;INCREMENT MASK BIT TO CLEAR
 6019  004026  000137  003276                  JMP     111$            ;TEST NEXT ISR BIT
 6020  004032  005337  002462          14$:    DEC     GRPCNT          ;TESTED BOTH GROUPS?
 6021  004036  001406                          BEQ     TST2            ;;BR IF DONE
 6022  004040  013700  002422                  MOV     DRCSC,R0        ;SETUP FOR GROUP 2
 6023  004044  013701  002426                  MOV     DRCSD,R1        ;CSRC = CONTROL GROUP 2
```

```
                                                        ;CSRD = DATA    GROUP 2
004050  000137  003240                JMP     120$            ;DO GROUP 2


                            ;;*****************************************************************
                            ;*TEST 2        TEST ACR/ISR INTERRUPT ,GROUP 1,GROUP 2
                            ;;*****************************************************************
004054  000004              TST2:   SCOPE

004056  013700  002412              MOV     DRCSA,R0
004062  013701  002416              MOV     DRCSB,R1
004066  013702  002412              MOV     DRCSA,R2
004072  012737  004144  002110      MOV     #111$,$LPERR    ;SET UP LOOP RETURN
004100  012737  000002  002462      MOV     #2,GRPCNT       ;DO TWO GROUPS
004106  004537  010746          120$:   JSR     R5,CLRCSR       ;CLEAR ALL CSRS
004112  012703  012570              MOV     #BGCHP3,R3      ;EXPECTED GDDAT FOR ISR
004116  012737  000050  002442      MOV     #CSIMR,IMRLOC   ;STORE CLEAR SINGLE IMR CODE
004124  004537  011026              JSR     R5,CLRIRR       ;CLEAR IRR REGS
004130  012737  011576  000300      MOV     #INTSR1,300     ;STORE VECTOR ROUTINE
004136  012737  000300  000302      MOV     #300,302        ;STORE PSW ;VER:0
004144  005037  002436          111$:   CLR     INTFLG          ;CLEAR INT. FLAG
004150  012706  002100              MOV     #2100,SP        ;INIT STACK IN CASE OF ILLEGAL INT. LOOP
004154  042712  001000              BIC     #BIT9,(R2)      ;CLEAR INTERRUPT ENABLE
004160  112712  000242              MOVB    #242,(R2)       ;CLEAR MASTER MASK GP1
004164  112710  000242              MOVB    #242,(R0)       ;CLEAR MASTER MASK GP1,GP2
004170  112710  000300              MOVB    #PACR,(R0)      ;PRESELECT ACR FOR WRITING
004174  111311                      MOVB    (R3),(R1)       ;NEXT BIT INTO ACR
004176  112710  000340              MOVB    #PVMA,(R0)      ;PRESELECT VECTOR MEM ADDR
004202  112711  000060              MOVB    #60,(R1)        ;WRITE VECTOR 300,BY0,LV0
004206  010037  002122              MOV     R0,$BDADR       ;STORE BUS ADDRESS
004212  112710  000060              MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
004216  113710  002442              MOVB    IMRLOC,(R0)     ;CLEAR MASK ON SINGLE BIT
004222  112710  000202              MOVB    #202,(R0)       ;LMD04 - COMMON VECTOR
004226  106427  000000              MTPS    #PR0
004232  152762  000002  000001      BISB    #BIT1,1(R2)     ;SET I/E,NO INTERRUPT
004240  005737  002436              TST     INTFLG          ;CHECK FOR INTERRUPTS
004244  001401                      BEQ     1$
004246  104011                      ERROR   11              ;ERROR,ILLEGAL INTERRUPT
004250  005037  002436          1$:     CLR     INTFLG          ;RESET INT. COUNTER
004254  112710  000120              MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
004260  106427  000300              MTPS    #PR6    ;VER:0
004264  005737  002436              TST     INTFLG          ;CHECK FOR NO INTERRUPTS
004270  001401                      BEQ     2$
004272  104011                      ERROR   11              ;ILLEGAL INTERRUPTS
004274  005037  002436          2$:     CLR     INTFLG          ;CLEAR INT. COUNTER
004300  106427  000000              MTPS    #PR0
004304  112712  000241              MOVB    #241,(R2)       ;USED IN TEST OF GROUP 2
                                                        ;ARM GROUP 1 CHIP TO PASS
                                                        ;ENABLE TO GROUP 2 CHIP.
004310  112710  000241              MOVB    #241,(R0)       ;ARM THE CHIP
004314  012737  000001  002124      MOV     #1,$GDDAT       ;EXPECT ONE INTERRUPT
004322  013737  002436  002126      MOV     INTFLG,$BDDAT   ;SHOULD HAVE ONE INTERRUPT
004330  023737  002124  002126      CMP     $GDDAT,$BDDAT
004336  001401                      BEQ     3$
004340  104012                      ERROR   12              ;INTERRUPT TEST ERROR
004342  106427  000300          3$:     MTPS    #PR6    ;VER:0
```

```
 6076  004346  010137  002122          4$:   MOV   R1,$BDADR       ;READY TO READ ISR REG.
 6077  004352  005037  002126                CLR   $BDDAT         ;SET UP FOR BYTE READS
 6078  004356  005037  002124                CLR   $GDDAT         ;SET UP FOR BYTE EXPECTED
 6079  004362  112710  000240                MOVB  #MISR,(R0)     ;LOAD MODE TO READ ISR
 6080  004366  111137  002126                MOVB  (R1),$BDDAT
 6081  004372  023737  002124  002126        CMP   $GDDAT,$BDDAT  ;ACR SHOULD CLEAR ISR
 6082  004400  001401                       BEQ   5$
 6083  004402  104006                       ERROR 6              ;ISR ERROR
 6084  004404  116337  000001  002124  5$:   MOVB  1(R3),$GDDAT   ;STORE EXPECTED
 6085  004412  112710  000244                MOVB  #MIMR,(R0)     ;LOAD MODE FOR IMR
 6086  004416  111137  002126                MOVB  (R1),$BDDAT
 6087  004422  023737  002124  002126        CMP   $GDDAT,$BDDAT
 6088  004430  001401                       BEQ   6$
 6089  004432  104005                       ERROR 5              ;IMR ERROR
 6090  004434  116337  000001  002124  6$:   MOVB  1(R3),$GDDAT   ;EXPECTED IRR
 6091  004442  112710  000250                MOVB  #MIRR,(R0)     ;LOAD MODE BITS FOR IRR
 6092  004446  111137  002126                MOVB  (R1),$BDDAT
 6093  004452  023737  002124  002126        CMP   $GDDAT,$BDDAT
 6094  004460  001401                       BEQ   7$
 6095  004462  104003                       ERROR 3              ;IRR ERROR
 6096  004464  111337  002124          7$:   MOVB  (R3),$GDDAT    ;EXPECTED ACR
 6097  004470  112710  000254                MOVB  #MACR,(R0)
 6098  004474  111137  002126                MOVB  (R1),$BDDAT
 6099  004500  023737  002124  002126        CMP   $GDDAT,$BDDAT
 6100  004506  001401                       BEQ   10$
 6101  004510  104004                       ERROR 4              ;ACR ERROR
 6102  004512  005723                 10$:   TST   (R3)+          ;GET NEXT ISR,IMR,IRR EXPECTED RESULTS
 6103  004514  020327  012610                CMP   R3,#EDCHP3     ;CHECK FOR END
 6104  004520  001404                       BEQ   11$            ;DONE WITH GROUP TEST
 6105  004522  005237  002442                INC   IMRLOC         ;INCREMENT MASK BIT TO CLEAR
 6106  004526  000137  004144                JMP   111$           ;TEST NEXT ISR BIT
 6107  004532  005337  002462          11$:  DEC   GRPCNT         ;TESTED BOTH GROUPS?
 6108  004536  001406                       BEQ   TST3           ;;BR IF DONE
 6109  004540  013700  002422                MOV   DRCSC,R0       ;SETUP FOR GROUP 2
 6110  004544  013701  002426                MOV   DRCSD,R1       ;CSRC = CONTROL GROUP 2
 6111                                                             ;CSRD = DATA    GROUP 2
 6112  004550  000137  004106                JMP   120$           ;DO GROUP 2
 6113
 6114
 6115
 6116                                 ;;*****************************************************************
   (3)                                ;*TEST 3        TEST VECTOR ADDR MEM,LEV 0-7,BY0-3,(204-1774),GRPS 1,2
   (3)                                ;;*****************************************************************
   (2)  004554  000004                TST3:  SCOPE
   (2)
 6117  004556  013700  002412                MOV   DRCSA,R0
 6118  004562  013701  002416                MOV   DRCSB,R1
 6119  004566  013702  002412                MOV   DRCSA,R2
 6120  004572  012737  000002  002462        MOV   #2,GRPCNT      ;DO TWO GROUPS
 6121  004600  012737  000001  002472  120$:  MOV   #1,BYNUM       ;INIT FOR BYTE COUNT 0
 6122  004606  004537  010746                JSR   R5,CLRCSR      ;CLEAR ALL CSRS
 6123  004612  012737  000340  002456        MOV   #PVMA,VECVAL   ;START VECTOR LEVEL 0,BYTE COUNT 0
 6124  004620  012704  012570          121$: MOV   #BGCHP3,R4     ;EXPECTED ISR,IMR PATTERN
 6125  004624  012737  000010  002466        MOV   #8.,LVLCNT     ;COUNTER FOR 0-7 LEVELS
 6126  004632  012737  000050  002442        MOV   #CSIMR,IMRLOC  ;STORE SINGLE IMR CODE
 6127  004640  012737  000130  002446        MOV   #SSIRR,IRRLOC  ;STORE SINGLE IRR CODE
```

```
6128  004646  012737  000170  002444          MOV     #CSISR,ISRLOC   ;STORE CLEAR SINGLE ISR CODE
6129  004654  004537  011456                  JSR     R5,TRPCAT       ;RESTORE TRAP CATCHER
6130  004660  012737  004674  002110          MOV     #112$,$LPERR    ;STORE SCOPE LOOP
6131  004666  012737  000204  002452  111$:   MOV     #204,VECLOC     ;START AT VECTOR 204
6132  004674  106427  000300          112$:   MTPS    #300            ;VER:0
6133  004700  012706  002100                  MOV     #2100,SP        ;INIT STACK IN CASE OF ILLEGAL INT. LOOP
6134  004704  042712  001000                  BIC     #BIT9,(R2)      ;CLEAR INTERRUPT ENABLE
6135  004710  004537  011026                  JSR     R5,CLRIRR       ;CLEAR IRR REGS
6136  004714  112712  000241                  MOVB    #241,(R2)       ;USED IN TEST OF GROUP 2
6137                                                                  ;ARM GROUP 1 CHIP TO PASS
6138                                                                  ;ENABLE TO GROUP2
6139  004720  013703  002452                  MOV     VECLOC,R3       ;GET VECTOR
6140  004724  010337  002454                  MOV     R3,VECPAT
6141  004730  006037  002454                  ROR     VECPAT          ;ROTATE VECTOR ADDR TWO RIGHT
6142  004734  006037  002454                  ROR     VECPAT          ;TO WRITE VECTOR MEM ADDR.
6143  004740  012713  011632                  MOV     #INTBY4,(R3)    ;STORE VECTOR ROUTINE
6144  004744  012763  000300  000002          MOV     #300,2(R3)      ;STORE PSW ;VER:0
6145  004752  005037  002436                  CLR     INTFLG          ;CLEAR INT. FLAG
6146  004756  013737  002472  002470          MOV     BYNUM,BYCNT     ;BYTE COUNTS OF 0-3
6147  004764  113710  002456                  MOVB    VECVAL,(R0)     ;PRESELECT VECTOR MEM ADDR
6148  004770  113711  002454          113$:   MOVB    VECPAT,(R1)     ;WRITE VECTOR
6149  004774  005337  002470                  DEC     BYCNT           ;DONE WITH BYTE COUNT VALUE
6150  005000  001373                          BNE     113$            ;NO,LOAD VECTOR FOR NEXT BYTE COUNT
6151  005002  010037  002122                  MOV     R0,$BDADR       ;SAVE BUS ADDRESS
6152  005006  112710  000060                  MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
6153  005012  113710  002442                  MOVB    IMRLOC,(R0)     ;CLEAR MASK ON SINGLE BIT
6154  005016  106427  000000                  MTPS    #PR0
6155  005022  112710  000241                  MOVB    #241,(R0)       ;LMD57 - ARM THE CHIP
6156  005026  005737  002436                  TST     INTFLG          ;CHECK FOR INTERRUPTS
6157  005032  001401                          BEQ     1$
6158  005034  104011                          ERROR   11              ;ERROR,ILLEGAL INTERRUPT
6159  005036  005037  002436          1$:     CLR     INTFLG          ;RESET INT. COUNTER
6160  005042  113710  002446                  MOVB    IRRLOC,(R0)     ;SET SINGLE IRR BIT
6161  005046  106427  000300                  MTPS    #PR6    ;VER:0
6162  005052  005737  002436                  TST     INTFLG          ;CHECK FOR NO INTERRUPTS
6163  005056  001401                          BEQ     2$
6164  005060  104011                          ERROR   11              ;ILLEGAL INTERRUPTS
6165  005062  005037  002436          2$:     CLR     INTFLG          ;CLEAR INT. COUNTER
6166  005066  106427  000000                  MTPS    #PR0
6167  005072  152762  000002  000001          BISB    #BIT1,1(R2)     ;SET I/E,EXPECT INTERRUPT
6168  005100  013737  002472  002124          MOV     BYNUM,$GDDAT    ;EXPECT 1 TO 4 INTERRUPTS
6169                                                                  ;BASED ON BYTE COUNT VALUE
6170  005106  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SHOULD HAVE 1 TO 4 INTERRUPTS
6171                                                                  ;BASED ON BYTE COUNT VALUE
6172  005114  023737  002124  002126          CMP     $GDDAT,$BDDAT
6173  005122  001401                          BEQ     3$
6174  005124  104012                          ERROR   12              ;INTERRUPT TEST ERROR
6175  005126  106427  000300          3$:     MTPS    #PR6    ;VER:0
6176  005132  010137  002122          4$:     MOV     R1,$BDADR       ;READY TO READ ISR REG.
6177  005136  005037  002126                  CLR     $BDDAT          ;SET UP FOR BYTE READS
6178  005142  005037  002124                  CLR     $GDDAT          ;SET UP FOR BYTE EXPECTED
6179  005146  111437  002124                  MOVB    (R4),$GDDAT     ;EXPECTED DATA
6180  005152  112710  000240                  MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
6181  005156  111137  002126                  MOVB    (R1),$BDDAT
6182  005162  023737  002124  002126          CMP     $GDDAT,$BDDAT
6183  005170  001401                          BEQ     5$
```

```
                                                ERROR   6               ;ISR ERROR
005172  104006
005174  116437  000001  002124  5$:           MOVB    1(R4),$GDDAT    ;STORE EXPECTED
005202  112710  000244                        MOVB    #MIMR,(R0)      ;LOAD MODE FOR IMR
005206  111137  002126                        MOVB    (R1),$BDDAT
005212  023737  002124  002126                CMP     $GDDAT,$BDDAT
005220  001401                                BEQ     6$
005222  104005                                ERROR   5               ;IMR ERROR
005224  005037  002124          6$:           CLR     $GDDAT
005230  112710  000250                        MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
005234  111137  002126                        MOVB    (R1),$BDDAT
005240  023737  002124  002126                CMP     $GDDAT,$BDDAT
005246  001401                                BEQ     7$
005250  104003                                ERROR   3               ;IRR ERROR
005252  005037  002124          7$:           CLR     $GDDAT
005256  112710  000254                        MOVB    #MACR,(R0)
005262  111137  002126                        MOVB    (R1),$BDDAT
005266  023737  002124  002126                CMP     $GDDAT,$BDDAT
005274  001401                                BEQ     10$
005276  104004                                ERROR   4               ;ACR ERROR
005300  005037  002124          10$:          CLR     $GDDAT
005304  113710  002444                        MOVB    ISRLOC,(R0)     ;CLEAR SINGLE ISR BIT
005310  112710  000240                        MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
005314  111137  002126                        MOVB    (R1),$BDDAT
005320  023737  002124  002126                CMP     $GDDAT,$BDDAT
005326  001401                                BEQ     11$
005330  104006                                ERROR   6               ;ISR REG ERROR
005332  004537  011364          11$:          JSR     R5,RESTRP       ;RESTORE VECTOR JUST TESTED
005336  005737  017302                        TST     KXTFLAG         ; KXT11 ??                      ;;GPA
005342  001404                                BEQ     100$            ; BR IF NOT                     ;;GPA
005344  022737  000400  002452                CMP     #400,VECLOC     ; YES, FINISHED 204-374 ??      ;;GPA
005352  000403                                403                     ; SKIP NEXT TO FIND OUT         ;;GPA
005354                          100$:                                                                 ;;GPA
005354  022737  002000  002452                CMP     #2000,VECLOC    ;FINISHED?
005362  001402                                BEQ     12$             ;FINISHED VECTORS 204-1774?
005364  000137  004674                        JMP     112$            ;TEST NEXT VECTOR ADDR
005370  005724                  12$:          TST     (R4)+           ;INDEX EXPECTED ISR AND IMR PATTERN
005372  005237  002442                        INC     IMRLOC          ;STORE CLEAR FOR NEXT MASK BIT
005376  005237  002446                        INC     IRRLOC          ;STORE SET FOR NEXT IRR BIT
005402  005237  002444                        INC     ISRLOC          ;STORE CLEAR FOR NEXT ISR BIT
005406  005237  002456                        INC     VECVAL          ;SETUP TO TEST NEXT VECTOR LEVEL
005412  005337  002466                        DEC     LVLCNT          ;FINISHED LEVELS 0-7
005416  001402                                BEQ     13$             ;YES,CHECK BYTE COUNT END
005420  000137  004666                        JMP     111$            ;NO,DO NEXT LEVEL
005424  005237  002472          13$:          INC     BYNUM           ;INDEX BYTE COUNT
005430  104407                                CKSWR                   ;LOOK FOR SWITCH CHANGE REQUEST
005432  022737  000400  002456                CMP     #400,VECVAL     ;FINISHED  ALL BYTE COUNTS 0-3
005440  001402                                BEQ     14$             ;FINISHED TWO GROUPS?
005442  000137  004620                        JMP     121$            ;DO NEXT BYTE COUNT FOR
                                                                      ;LEVELS 0-7
005446  005337  002462          14$:          DEC     GRPCNT          ;FINISHED TWO GROUPS?
005452  001406                                BEQ     TST4            ;;BR IF DONE
005454  013700  002422                        MOV     DRCSC,R0        ;CSRC = CONTROL GROUP 2
005460  013701  002426                        MOV     DRCSD,R1        ;CSRD = DATA    GROUP 2
005464  000137  004600                        JMP     120$            ;DO GROUP 2 FOR BYTE COUNTS 0-3
                                                                      ;AND LEVELS 0-7 FOR EACH BYTE COUNT.
```

```
                                ;;*******************************************************************
                                ;*TEST 4         TEST VECTOR ADDR MEM,LVLS 0-7,BY0-3,(0-200)GRPS 1,2
                                ;;*******************************************************************
(2) 005470  000004              TST4:   SCOPE
(2)
6242  005472  013700  002412              MOV     DRCSA,R0
6243  005476  013701  002416              MOV     DRCSB,R1
6244  005502  013702  002412              MOV     DRCSA,R2
6245  005506  012737  000002  002462      MOV     #2,GRPCNT       ;DO TWO GROUPS
6246  005514  012737  000001  002472 120$: MOV     #1,BYNUM        ;INIT FOR FIRST BYTE COUNT
6247  005522  004537  010746              JSR     R5,CLRCSR       ;CLEAR ALL CSRS
6248  005526  012737  000340  002456      MOV     #PVMA,VECVAL    ;START VECTOR LEVEL 0,BYTE COUNT 0
6249  005534  012704  012570        121$: MOV     #BGCHP3,R4      ;ISR,IMR PATTERN
6250  005540  012737  000010  002466      MOV     #8.,LVLCNT      ;START LEVEL COUNT 0-7
6251  005546  012737  000050  002442      MOV     #CSIMR,IMRLOC   ;STORE CLEAR SINGLE IMR BIT CODE
6252  005554  012737  000130  002446      MOV     #SSIRR,IRRLOC   ;STORE SET SINGLE IRR BIT CODE
6253  005562  012737  005574  002110      MOV     #112$,$LPERR    ;LOOP RETURN
6254  005570  005037  002452        111$: CLR     VECLOC          ;START AT VECTOR 0
6255  005574  106427  000300        112$: MTPS    #300            ; VER:0
6256  005600  012706  002100              MOV     #2100,SP        ;INIT STACK IN CASE OF ILLEGAL INT. LOOP
6257  005604  042712  001000              BIC     #BIT9,(R2)      ;CLEAR INTERRUPT ENABLE
6258  005610  004537  011026              JSR     R5,CLRIRR       ;CLEAR IRR REGS
6259  005614  112712  000241              MOVB    #241,(R2)       ;USED IN TEST OF GROUP 2
6260                                                            ;ARM GROUP 1 CHIP TO PASS
6261                                                            ;ENABLE TO GROUP 2.
6262  005620  013703  002452              MOV     VECLOC,R3       ;GET VECTOR
6263  005624  004537  011756              JSR     R5,CAT200       ;STORE TRAP CATCHER FOR 0-200
6264  005630  010337  002454              MOV     R3,VECPAT
6265  005634  006037  002454              ROR     VECPAT          ;ROTATE VECTOR ADDR TWO RIGHT
6266  005640  006037  002454              ROR     VECPAT          ;TO WRITE VECTOR MEM ADDR.
6267  005644  012713  012056              MOV     #INTSR3,(R3)    ;STORE VECTOR ROUTINE
6268  005650  012763  000300  000002      MOV     #300,2(R3)      ;STORE PSW;VER:0
6269  005656  005037  002436              CLR     INTFLG          ;CLEAR INT. FLAG
6270  005662  013737  002472  002470      MOV     BYNUM,BYCNT     ;BYTE COUNT OF 0 TO 3
6271                                                            ;FOR 1 TO 4 VECTORS.
6272  005670  113710  002456              MOVB    VECVAL,(R0)     ;PRESELECT VECTOR MEM ADDR
6273  005674  113711  002454        113$: MOVB    VECPAT,(R1)     ;WRITE VECTOR
6274  005700  005337  002470              DEC     BYCNT           ;DO 1 TO 4 VECTORS
6275  005704  001373                      BNE     113$            ;WRITE VECTORS BASED ON BYTE COUNT
6276  005706  010037  002122              MOV     R0,$BDADR       ;SAVE BUS ADDRESS
6277  005712  112710  000060              MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
6278  005716  113710  002442              MOVB    IMRLOC,(R0)     ;CLEAR MASK ON SINGLE BIT
6279  005722  106427  000000              MTPS    #PR0
6280  005726  112710  000241              MOVB    #241,(R0)       ;LMD57 - ARM THE CHIP
6281  005732  005737  002436              TST     INTFLG          ;CHECK FOR INTERRUPTS
6282  005736  001405                      BEQ     1$
6283  005740  004537  011326              JSR     R5,FRMBUF       ;RESTORE 0-202
6284  005744  104011                      ERROR   11              ;ERROR,ILLEGAL INTERRUPT
6285  005746  004537  011756              JSR     R5,CAT200       ;RESTORE TRAP CATCHER
6286  005752  005037  002436        1$:   CLR     INTFLG          ;RESET INT. COUNTER
6287  005756  113710  002446              MOVB    IRRLOC,(R0)     ;SET SINGLE IRR BIT
6288  005762  106427  000300              MTPS    #PR6    ;VER:0
6289  005766  005737  002436              TST     INTFLG          ;CHECK FOR NO INTERRUPTS
6290  005772  001405                      BEQ     2$
6291  005774  004537  011326              JSR     R5,FRMBUF       ;RESTORE 0-202
```

```
6292  006000  104011                        ERROR   11              ;ILLEGAL INTERRUPTS
6293  006002  004537  011756                JSR     R5,CAT200       ;RESTORE TRAP CATCHER
6294  006006  005037  002436          2$:   CLR     INTFLG          ;CLEAR INT. COUNTER
6295  006012  106427  000000                MTPS    #PR0
6296  006016  152762  000002  000001        BISB    #BIT1,1(R2)     ;SET I/E,EXPECT INTERRUPT
6297  006024  013737  002472  002124        MOV     BYNUM,$GDDAT    ;EXPECT NUMBER OF INTERRUPTS
6298                                                                ;EQUAL TO BYTE COUNT.
6299  006032  013737  002436  002126        MOV     INTFLG,$BDDAT   ;SHOULD HAVE NUMBER OF INTERRUPTS
6300                                                                ;EQUAL TO BYTE COUNT.
6301  006040  023737  002124  002126        CMP     $GDDAT,$BDDAT
6302  006046  001405                        BEQ     3$
6303  006050  004537  011326                JSR     R5,FRMBUF       ;RESTORE 0-202
6304  006054  104012                        ERROR   12              ;INTERRUPT TEST ERROR
6305  006056  004537  011756                JSR     R5,CAT200       ;RESTORE TRAP CATCHER SUB
6306  006062  106427  000300          3$:   MTPS    #PR6    ;VER:0
6307  006066  010137  002122          4$:   MOV     R1,$BDADR       ;READY TO READ ISR REG.
6308  006072  005037  002126                CLR     $BDDAT          ;SET UP FOR BYTE READS
6309  006076  005037  002124                CLR     $GDDAT          ;SET UP FOR BYTE EXPECTED
6310  006102  111437  002124                MOVB    (R4),$GDDAT     ;EXPECTED DATA
6311  006106  112710  000240                MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
6312  006112  111137  002126                MOVB    (R1),$BDDAT
6313  006116  023737  002124  002126        CMP     $GDDAT,$BDDAT
6314  006124  001405                        BEQ     5$
6315  006126  004537  011326                JSR     R5,FRMBUF       ;RESTORE 0-200
6316  006132  104006                        ERROR   6               ;ISR ERROR
6317  006134  004537  011756                JSR     R5,CAT200       ;RESTORE TRAP CATCHER
6318  006140  116437  000001  002124  5$:   MOVB    1(R4),$GDDAT    ;STORE EXPECTED
6319  006146  112710  000244                MOVB    #MIMR,(R0)      ;LOAD MODE FOR IMR
6320  006152  111137  002126                MOVB    (R1),$BDDAT
6321  006156  023737  002124  002126        CMP     $GDDAT,$BDDAT
6322  006164  001405                        BEQ     6$
6323  006166  004537  011326                JSR     R5,FRMBUF       ;RESTORE 0-202
6324  006172  104005                        ERROR   5               ;IMR ERROR
6325  006174  004537  011756                JSR     R5,CAT200       ;RESTORE CATCHER
6326  006200  005037  002124          6$:   CLR     $GDDAT
6327  006204  112710  000250                MOVB    #MIRR,(R0)      ;LOAD MODE BITS FOR IRR
6328  006210  111137  002126                MOVB    (R1),$BDDAT
6329  006214  023737  002124  002126        CMP     $GDDAT,$BDDAT
6330  006222  001405                        BEQ     7$
6331  006224  004537  011326                JSR     R5,FRMBUF       ;RESTORE 0-202
6332  006230  104003                        ERROR   3               ;IRR ERROR
6333  006232  004537  011756                JSR     R5,CAT200       ;RESTORE TRAP CATCHER
6334  006236  005037  002124          7$:   CLR     $GDDAT
6335  006242  112710  000254                MOVB    #MACR,(R0)
6336  006246  111137  002126                MOVB    (R1),$BDDAT
6337  006252  023737  002124  002126        CMP     $GDDAT,$BDDAT
6338  006260  001405                        BEQ     10$
6339  006262  004537  011326                JSR     R5,FRMBUF       ;RESTORE 0-202
6340  006266  104004                        ERROR   4               ;ACR ERROR
6341  006270  004537  011756                JSR     R5,CAT200       ;RESTORE TRAP CATCHER
6342  006274  005037  002124          10$:  CLR     $GDDAT
6343  006300  112710  000160                MOVB    #CISR,(R0)      ;CLEAR ISR
6344  006304  112710  000240                MOVB    #MISR,(R0)      ;LOAD MODE TO READ ISR
6345  006310  111137  002126                MOVB    (R1),$BDDAT
6346  006314  023737  002124  002126        CMP     $GDDAT,$BDDAT
6347  006322  001405                        BEQ     11$
```

```
 6348  006324  004537  011326            JSR     R5,FRMBUF       ;RESTORE 0-202
 6349  006330  104006                    ERROR   6               ;ISR REG ERROR
 6350  006332  004537  011756            JSR     R5,CAT200       ;RESTORE TRAP CATCHER
 6351  006336  004537  011714    11$:    JSR     R5,RES200       ;RESTORE VECTOR JUST TESTED
 6352  006342  004537  011326            JSR     R5,FRMBUF       ;RESTORE 0-200 TO SYSMAC CONTROL
 6353  006346  022737  000204  002452    CMP     #204,VECLOC     ;FINISHED?
 6354  006354  001402                    BEQ     12$
 6355  006356  000137  005574            JMP     112$            ;TEST NEXT VECTOR ADDR
 6356  006362  005724            12$:    TST     (R4)+           ;INDEX EXPECTED ISR AND IMR PATTERN
 6357  006364  005237  002442            INC     IMRLOC          ;STORE CLEAR NEXT IMR BIT
 6358  006370  005237  002446            INC     IRRLOC          ;STORE SET NEXT IRR BIT
 6359  006374  005237  002456            INC     VECVAL          ;SETUP TO TEST NEXT VECTOR LEVEL
 6360  006400  005337  002466            DEC     LVLCNT          ;FINISHED LEVELS 0-7?
 6361  006404  001402                    BEQ     13$             ;YES,CHECK BYTE COUNT END
 6362  006406  000137  005570            JMP     111$            ;NO,DO NEXT LEVEL,SAME BYTE COUNT
 6363  006412  005237  002472    13$:    INC     BYNUM           ;INDEX BYTE COUNT
 6364  006416  104407                    CKSWR                   ;LOOK FOR SWITCH CHANGE
 6365  006420  022737  000400  002456    CMP     #400,VECVAL     ;FINISHED ALL FOUR BYTE COUNTS
 6366  006426  001402                    BEQ     14$             ;YES,FINISHED GROUP
 6367  006430  000137  005534            JMP     121$            ;NO,DO NEXT BYTE COUNT WITH
 6368                                                            ;LEVELS 0-7.
 6369  006434  005337  002462    14$:    DEC     GRPCNT          ;FINISHED BOTH GROUP 1 AND GROUP 2?
 6370  006440  001406                    BEQ     15$             ;YES,BOTH GROUPS TESTED
 6371  006442  013700  002422            MOV     DRCSC,R0        ;NO,TEST GROUP 2
 6372                                                            ;CSRC = CONTROL GROUP 2
 6373  006446  013701  002426            MOV     DRCSD,R1        ;CSRD = DATA    GROUP 2
 6374  006452  000137  005514            JMP     120$            ;RETURN AND TEST GROUP2
 6375                                                            ;FOR BYTE COUNTS 0-3 AND
 6376                                                            ;LEVELS 0-7 FOR EACH BYTE COUNT.
 6377  006456  004537  011326    15$:    JSR     R5,FRMBUF       ;RESTORE 0-202,EXIT TEST
 6378
 6379
 6380                            ;;*******************************************************************
  (3)                            ;*TEST 5        TEST VECTOR ADDR UNIQUENESS IN FIXED MODE FOR GRPS 1,2
  (3)                            ;;*******************************************************************
  (2)  006462  000004            TST5:   SCOPE
  (2)
 6381                                                            ;BYTE COUNT = 4
 6382                                                            ;LEVELS = 0-7
 6383  006464  004537  010746            JSR     R5,CLRCSR       ;CLEAR ALL CSRS
 6384  006470  004537  011266            JSR     R5,CLRBF        ;CLEAR VECTOR BUFFER AREA
 6385  006474  004537  011456            JSR     R5,TRPCAT       ;RESTORE TRAP CATCHER
 6386  006500  004537  011232            JSR     R5,STRVEC       ;STORE VECTOR AREA 300-674
 6387                                                            ;WITH INT. SERV ROUTINES
 6388  006504  012737  000377  002456    MOV     #377,VECVAL     ;VECTOR MEM  FINAL VALUE
 6389  006512  004537  011026            JSR     R5,CLRIRR       ;CLEAR IRR REGS
 6390  006516  012704  017030            MOV     #VBUF,R4        ;VECTOR ADDRESS BUFFER
 6391  006522  004537  011064            JSR     R5,VECFIL       ;FILL VECTOR MEMORY FOR GROUPS 1,2
 6392                                                            ;WITH VECTORS 300-674
 6393  006526  012705  012142            MOV     #SETVEC,R5      ; R5 POINTS TO COMMON HANDLER   ;;GPA
 6394  006532  013700  002412            MOV     DRCSA,R0        ;GROUP 1 CSR
 6395  006536  013701  002422            MOV     DRCSC,R1        ;GROUP 2 CSR
 6396  006542  013702  002416            MOV     DRCSB,R2
 6397  006546  013703  002426            MOV     DRCSD,R3
 6398  006552  010037  002122            MOV     R0,$BDADR       ;STORE BUS ADDRESS
 6399  006556  112710  000300            MOVB    #PACR,(R0)      ;PRESELECT GP1 ACR FOR WRITING
```

```
6400  006562  112712  000377                    MOVB    #377,(R2)         ;SET ALL ACR BITS GP1
6401  006566  112710  000040                    MOVB    #CIMR,(R0)        ;CLEAR ALL IMR BITS GP1
6402  006572  005037  002436                    CLR     INTFLG            ;CLEAR INT. FLAG
6403  006576  106427  000000                    MTPS    #PR0
6404  006602  112710  000120                    MOVB    #SIRR,(R0)        ;SET ALL IRR BITS
6405  006606  112710  000241                    MOVB    #241,(R0)         ;LMD57 - ARM THE CHIP
6406  006612  106427  000300                    MTPS    #300              ;VER:0
6407  006616  005737  002436                    TST     INTFLG            ;CHECK FOR NO INTERRUPTS
6408  006622  001401                            BEQ     1$
6409  006624  104011                            ERROR   11                ;ILLEGAL INTERRUPT
6410  006626  005037  002436            1$:     CLR     INTFLG
6411  006632  112711  000300                    MOVB    #PACR,(R1)        ;PRESELECT GROUP2 FOR WRITING
6412  006636  112713  000377                    MOVB    #377,(R3)         ;ALL ACR BITS SET GP2
6413  006642  112711  000040                    MOVB    #CIMR,(R1)        ;CLEAR ALL IMR BITS GP2
6414  006646  106427  000000                    MTPS    #PR0
6415  006652  112711  000120                    MOVB    #SIRR,(R1)        ;SET ALL IRR BITS
6416  006656  112711  000241                    MOVB    #241,(R1)         ;LMD57 - ARM THE CHIP
6417  006662  106427  000300                    MTPS    #PR6    ;VER:0
6418  006666  005737  002436                    TST     INTFLG            ;CHECK FOR NO INTERRUPTS
6419  006672  001401                            BEQ     2$
6420  006674  104011                            ERROR   11                ;ILLEGAL INTERRUPT
6421  006676  005037  002436            2$:     CLR     INTFLG
6422  006702  106427  000000                    MTPS    #PR0
6423  006706  152760  000002  000001            BISB    #BIT1,1(R0)       ;SET INT. ENABLE,EXPECT INTERRUPTS
6424  006714  012737  000100  002124            MOV     #64.,$GDDAT       ;EXPECT 64 INTERRUPTS
6425  006722  013737  002436  002126            MOV     INTFLG,$BDDAT     ;SHOULD HAVE 64 INTERRUPTS
6426  006730  106427  000300                    MTPS    #PR6    ;VER:0
6427  006734  023737  002124  002126            CMP     $GDDAT,$BDDAT
6428  006742  001401                            BEQ     3$
6429  006744  104012                            ERROR   12                ;INTERRUPT TEST ERROR
6430  006746  012737  101710  002124    3$:     MOV     #101710,$GDDAT    ;RDY,DIR,I/E,CHIP - 31X
6431  006754  010037  002122                    MOV     R0,$BDADR         ;CSRA
6432  006760  011037  002126                    MOV     (R0),$BDDAT
6433  006764  042737  000007  002126            BIC     #7,$BDDAT         ;CLEAR UNDEFINED BITS
6434  006772  023737  002124  002126            CMP     $GDDAT,$BDDAT
6435  007000  001401                            BEQ     4$
6436  007002  104002                            ERROR   2                 ;CSRA REG ERROR
6437  007004  012737  000310  002124    4$:     MOV     #310,$GDDAT       ;STORE EXPECTED
6438  007012  010137  002122                    MOV     R1,$BDADR         ;CSRC
6439  007016  011137  002126                    MOV     (R1),$BDDAT       ;
6440  007022  042737  000007  002126            BIC     #7,$BDDAT         ;CLEAR UNDEFINED BITS
6441  007030  023737  002124  002126            CMP     $GDDAT,$BDDAT
6442  007036  001401                            BEQ     5$
6443  007040  104002                            ERROR   2                 ;CSRC ERROR
6444  007042  010237  002122            5$:     MOV     R2,$BDADR         ;CSRB ADDRESS
6445  007046  005037  002126                    CLR     $BDDAT
6446  007052  005037  002124                    CLR     $GDDAT
6447  007056  112710  000240                    MOVB    #MISR,(R0)        ;READ ISR
6448  007062  111237  002126                    MOVB    (R2),$BDDAT
6449  007066  023737  002124  002126            CMP     $GDDAT,$BDDAT
6450  007074  001401                            BEQ     6$
6451  007076  104006                            ERROR   6                 ;ISR ERROR,GP1
6452  007100  112710  000244            6$:     MOVB    #MIMR,(R0)        ;MODE BITS FOR IMR
6453  007104  111237  002126                    MOVB    (R2),$BDDAT
6454  007110  023737  002124  002126            CMP     $GDDAT,$BDDAT
6455  007116  001401                            BEQ     7$
```

```
 6456  007120  104005                    ERROR   5               ;IMR ERROR,GP1
 6457  007122  112710  000250    7$:     MOVB    #MIRR,(R0)      ;MODE BITS FOR IRR
 6458  007126  111237  002126            MOVB    (R2),$BDDAT
 6459  007132  023737  002124  002126    CMP     $GDDAT,$BDDAT
 6460  007140  001401                    BEQ     10$
 6461  007142  104003                    ERROR   3               ;IRR ERROR
 6462  007144  012737  000377  002124  10$:     MOV     #377,$GDDAT     ;EXPECTED ACR
 6463  007152  112710  000254            MOVB    #MACR,(R0)      ;READ ACR
 6464  007156  111237  002126            MOVB    (R2),$BDDAT
 6465  007162  023737  002124  002126    CMP     $GDDAT,$BDDAT
 6466  007170  001401                    BEQ     11$
 6467  007172  104004                    ERROR   4               ;ACR ERROR
 6468  007174  005037  002124    11$:    CLR     $GDDAT
 6469  007200  005037  002126            CLR     $BDDAT
 6470  007204  010337  002122            MOV     R3,$BDADR       ;CSRD ADDRESS
 6471  007210  112711  000240            MOVB    #MISR,(R1)      ;READ ISR ,GP2
 6472  007214  111337  002126            MOVB    (R3),$BDDAT
 6473  007220  023737  002124  002126    CMP     $GDDAT,$BDDAT
 6474  007226  001401                    BEQ     12$
 6475  007230  104006                    ERROR   6               ;ISR ERROR,GP1
 6476  007232  112711  000244    12$:    MOVB    #MIMR,(R1)      ;READ IMR GP2
 6477  007236  111337  002126            MOVB    (R3),$BDDAT
 6478  007242  023737  002124  002126    CMP     $GDDAT,$BDDAT
 6479  007250  001401                    BEQ     13$
 6480  007252  104005                    ERROR   5               ;IMR ERROR,GP2
 6481
 6482  007254  112711  000250    13$:    MOVB    #MIRR,(R1)      ;READ IRR
 6483  007260  111337  002126            MOVB    (R3),$BDDAT
 6484  007264  023737  002124  002126    CMP     $GDDAT,$BDDAT
 6485  007272  001401                    BEQ     14$
 6486  007274  104003                    ERROR   3               ;IRR ERROR,GP2
 6487  007276  112711  000254    14$:    MOVB    #MACR,(R1)      ;READ ACR
 6488  007302  012737  000377  002124    MOV     #377,$GDDAT
 6489  007310  111337  002126            MOVB    (R3),$BDDAT
 6490  007314  023737  002124  002126    CMP     $GDDAT,$BDDAT
 6491  007322  001401                    BEQ     15$
 6492  007324  104004                    ERROR   4               ;ACR ERROR,GP2
 6493  007326  012702  000100    15$:    MOV     #64.,R2
 6494  007332  012704  017030            MOV     #VBUF,R4        ;VECTOR BUFFER
 6495  007336  012737  000300  002124    MOV     #300,$GDDAT     ;START WITH FIRST VECTOR 300
 6496  007344  010437  002122    16$:    MOV     R4,$BDADR       ;VECTOR BUFFER ADDRESS
 6497  007350  011437  002126            MOV     (R4),$BDDAT
 6498  007354  023737  002124  002126    CMP     $GDDAT,$BDDAT
 6499  007362  001401                    BEQ     17$             ;VECTORED PROPERLY
 6500  007364  104007                    ERROR   7               ;VECTOR ADDR MEM ERROR
 6501  007366  062737  000004  002124  17$:     ADD     #4,$GDDAT
 6502  007374  005737  017302            TST     KXTFLAG         ; KXT11 ??                      ;;GPA
 6503  007400  001407                    BEQ     1000$           ; BR IF NOT                     ;;GPA
 6504  007402  023727  002124  000400    CMP     $GDDAT,#400     ; YES, REACHED VECTOR 400 ??    ;;GPA
 6505  007410  103403                    BLO     1000$           ; NOT YET, CONTINUE             ;;GPA
 6506  007412  012737  000300  002124    MOV     #300,$GDDAT     ; YES, RESET EXPECTED TO 300    ;;GPA
 6507  007420                    1000$:                                                          ;;GPA
 6508  007420  062704  000002            ADD     #2,R4           ;NEXT BUFFER ADDRESS
 6509  007424  005302                    DEC     R2              ;COMPLETED 64 BUFFER CHECKS?
 6510  007426  001346                    BNE     16$             ;CHECK NEXT VECTOR BUFFER LOCATION
 6511                                                            ;THAT WAS STORED BY THE INTERRUPT
```

```
                                                        ;SERVICE ROUTINES.
                                                ;;*****************************************************************
                                                ;*TEST 6        TEST VECTOR ADDR UNIQUENESS IN ROTATING MODE FOR GRPS 1,2
                                                ;;*****************************************************************
007430  000004                                  TST6:   SCOPE
                                                        ;BYTE COUNT = 4
                                                        ;LEVELS = 0-7
007432  004537  010746                                  JSR     R5,CLRCSR       ;CLEAR ALL CSRS
007436  004537  011266                                  JSR     R5,CLRBF        ;CLEAR VECTOR BUFFER AREA
007442  004537  011456                                  JSR     R5,TRPCAT       ;RESTORE TRAP CATCHER
007446  004537  011232                                  JSR     R5,STRVEC       ;STORE VECTOR AREA 300-674
                                                        ;WITH INT. SERV ROUTINES
007452  012737  000377  002456                          MOV     #377,VECVAL     ;VECTOR MEM  FINAL VALUE
007460  004537  011026                                  JSR     R5,CLRIRR       ;CLEAR IRR REGS
007464  012704  017030                                  MOV     #VBUF,R4        ;VECTOR ADDRESS BUFFER
007470  004537  011064                                  JSR     R5,VECFIL       ;FILL VECTOR MEMORY FOR GROUPS 1,2
                                                        ;WITH VECTORS 300-674
007474  012705  012142                                  MOV     #SETVEC,R5      ; R5 POINTS TO COMMON HANDLER.  ;;GPA
007500  013700  002412                                  MOV     DRCSA,R0        ;GROUP 1 CSR
007504  013701  002422                                  MOV     DRCSC,R1        ;GROUP 2 CSR
007510  013702  002416                                  MOV     DRCSB,R2
007514  013703  002426                                  MOV     DRCSD,R3
007520  010037  002122                                  MOV     R0,$BDADR       ;STORE BUS ADDRESS
007524  012737  000010  002474                          MOV     #8.,LVLSAV      ;COUNTER FOR 7 LEVELS
007532  112710  000201                                  MOVB    #201,(R0)       ;ROTATING PRIORITY GROUP 1
007536  112710  000040                                  MOVB    #CIMR,(R0)      ;CLEAR ALL IMR BITS GP1
007542  005037  002436                                  CLR     INTFLG          ;CLEAR INT. FLAG
007546  106427  000000                                  MTPS    #PR0
007552  112710  000120                                  MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
007556  112710  000241                                  MOVB    #241,(R0)       ;LMD57 - ARM THE CHIP
007562  106427  000300                                  MTPS    #300            ;VER:0
007566  005737  002436                                  TST     INTFLG          ;CHECK FOR NO INTERRUPTS
007572  001401                                          BEQ     1$
007574  104011                                          ERROR   11              ;ILLEGAL INTERRUPT
007576  005037  002436                  1$:             CLR     INTFLG
007602  112711  000201                                  MOVB    #201,(R1)       ;ROTATING PRIORITY FOR GROUP 2
007606  112711  000040                                  MOVB    #CIMR,(R1)      ;CLEAR ALL IMR BITS GP2
007612  106427  000000                                  MTPS    #PR0
007616  112711  000120                                  MOVB    #SIRR,(R1)      ;SET ALL IRR BITS
007622  112711  000241                                  MOVB    #241,(R1)       ;LMD57 - ARM THE CHIP
007626  106427  000300                                  MTPS    #PR6    ;VER:0
007632  005737  002436                                  TST     INTFLG          ;CHECK FOR NO INTERRUPTS
007636  001401                                          BEQ     2$
007640  104011                                          ERROR   11              ;ILLEGAL INTERRUPT
007642  005037  002436                  2$:             CLR     INTFLG
007646  106427  000000                                  MTPS    #PR0
007652  152760  000002  000001                          BISB    #BIT1,1(R0)     ;SET INT. ENABLE,EXPECT INTERRUPTS
007660  012737  000004  002124                          MOV     #4,$GDDAT       ;EXPECT 4 INTERRUPTS
007666  013737  002436  002126                          MOV     INTFLG,$BDDAT   ;SHOULD HAVE 4 INTERRUPTS
007674  106427  000300                                  MTPS    #PR6    ;VER:0
007700  023737  002124  002126                          CMP     $GDDAT,$BDDAT
007706  001401                                          BEQ     100$
007710  104012                                          ERROR   12              ;INTERRUPT TEST ERROR
007712  112710  000140                  100$:           MOVB    #CHPISR,(R0)    ;CLEAR HIGHEST PRIORITY ISR
```

```
 6564  007716  112710  000120                    MOVB    #SIRR,(R0)      ;SET ALL IRR BITS
 6565  007722  005337  002474                    DEC     LVLSAV          ;DONE 8 LEVELS,GROUP 1
 6566  007726  001345                            BNE     2$              ;DO NEXT LEVEL IN ROTATION
 6567  007730  112710  000100                    MOVB    #CIRR,(R0)      ;CLEAR IRR BITS GROUP 1
 6568  007734  012737  000010  002474            MOV     #8.,LVLSAV      ;DO 8 LEVELS,GROUP 2
 6569  007742  005037  002436          101$:     CLR     INTFLG          ;CLEAR INT FLAG
 6570  007746  106427  000000                    MTPS    #PR0
 6571  007752  012737  000004  002124            MOV     #4,$GDDAT
 6572  007760  013737  002436  002126            MOV     INTFLG,$BDDAT   ;SHOULD HAVE FOUR INTERRUPTS
 6573  007766  106427  000300                    MTPS    #PR6    ;VER:0
 6574  007772  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6575  010000  001401                            BEQ     102$
 6576  010002  104012                            ERROR   12              ;INTERRUPT TEST ERROR
 6577  010004  112711  000140          102$:     MOVB    #CHPISR,(R1)    ;CLEAR HIGHEST PRIORITY ISR
 6578  010010  112711  000120                    MOVB    #SIRR,(R1)      ;SET ALL IRR BITS
 6579  010014  005337  002474                    DEC     LVLSAV          ;DONE 8 LEVELS?
 6580  010020  001350                            BNE     101$            ;DO NEXT LEVEL IN ROTATION,GP2
 6581  010022  112711  000100                    MOVB    #CIRR,(R1)      ;CLEAR IRR BITS,GROUP 2
 6582  010026  012737  101750  002124    3$:     MOV     #101750,$GDDAT  ;RDY,DIR,I/E,CHIP - 35X
 6583  010034  010037  002122                    MOV     R0,$BDADR       ;CSRA
 6584  010040  011037  002126                    MOV     (R0),$BDDAT
 6585  010044  042737  000007  002126            BIC     #7,$BDDAT       ;CLEAR UNDEFINED BITS
 6586  010052  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6587  010060  001401                            BEQ     4$
 6588  010062  104002                            ERROR   2               ;CSRA REG ERROR
 6589  010064  012737  000350  002124    4$:     MOV     #350,$GDDAT     ;STORE EXPECTED
 6590  010072  010137  002122                    MOV     R1,$BDADR       ;CSRC
 6591  010076  011137  002126                    MOV     (R1),$BDDAT     ;
 6592  010102  042737  000007  002126            BIC     #7,$BDDAT       ;CLEAR UNDEFINED BITS
 6593  010110  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6594  010116  001401                            BEQ     5$
 6595  010120  104002                            ERROR   2               ;CSRC ERROR
 6596  010122  010237  002122          5$:       MOV     R2,$BDADR       ;CSRB ADDRESS
 6597  010126  005037  002126                    CLR     $BDDAT
 6598  010132  005037  002124                    CLR     $GDDAT
 6599  010136  112710  000240                    MOVB    #MISR,(R0)      ;READ ISR
 6600  010142  111237  002126                    MOVB    (R2),$BDDAT
 6601  010146  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6602  010154  001401                            BEQ     6$
 6603  010156  104006                            ERROR   6               ;ISR ERROR,GP1
 6604  010160  112710  000244          6$:       MOVB    #MIMR,(R0)      ;MODE BITS FOR IMR
 6605  010164  111237  002126                    MOVB    (R2),$BDDAT
 6606  010170  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6607  010176  001401                            BEQ     7$
 6608  010200  104005                            ERROR   5               ;IMR ERROR,GP1
 6609  010202  112710  000250          7$:       MOVB    #MIRR,(R0)      ;MODE BITS FOR IRR
 6610  010206  111237  002126                    MOVB    (R2),$BDDAT
 6611  010212  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6612  010220  001401                            BEQ     10$
 6613  010222  104003                            ERROR   3               ;IRR ERROR
 6614  010224  112710  000254          10$:      MOVB    #MACR,(R0)      ;READ ACR
 6615  010230  111237  002126                    MOVB    (R2),$BDDAT
 6616  010234  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6617  010242  001401                            BEQ     11$
 6618  010244  104004                            ERROR   4               ;ACR ERROR
 6619  010246  010337  002122          11$:      MOV     R3,$BDADR       ;CSRD ADDRESS
```

```
 6620  010252  112711  000240                    MOVB    #MISR,(R1)          ;READ ISR ,GP2
 6621  010256  111337  002126                    MOVB    (R3),$BDDAT
 6622  010262  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6623  010270  001401                            BEQ     12$
 6624  010272  104006                            ERROR   6                   ;ISR ERROR,GP1
 6625  010274  112711  000244            12$:    MOVB    #MIMR,(R1)          ;READ IMR GP2
 6626  010300  111337  002126                    MOVB    (R3),$BDDAT
 6627  010304  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6628  010312  001401                            BEQ     13$
 6629  010314  104005                            ERROR   5                   ;IMR ERROR,GP2
 6630
 6631  010316  112711  000250            13$:    MOVB    #MIRR,(R1)          ;READ IRR
 6632  010322  111337  002126                    MOVB    (R3),$BDDAT
 6633  010326  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6634  010334  001401                            BEQ     14$
 6635  010336  104003                            ERROR   3                   ;IRR ERROR,GP2
 6636  010340  112711  000254            14$:    MOVB    #MACR,(R1)          ;READ ACR
 6637  010344  111337  002126                    MOVB    (R3),$BDDAT
 6638  010350  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6639  010356  001401                            BEQ     200$
 6640  010360  104004                            ERROR   4                   ;ACR ERROR,GP2
 6641  010362  105010                    200$:   CLRB    (R0)                ;INIT GROUP 1 MODE BITS
 6642  010364  105011                            CLRB    (R1)                ;INIT GROUP 2 MODE BITS
 6643  010366  012737  101700  002124            MOV     #101700,$GDDAT      ;EXPECTED CSRA
 6644  010374  010037  002122                    MOV     R0,$BDADR           ;CSRA
 6645  010400  011037  002126                    MOV     (R0),$BDDAT
 6646  010404  042737  000007  002126            BIC     #7,$BDDAT           ;CLEAR UNDEFINED BITS
 6647  010412  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6648  010420  001401                            BEQ     201$
 6649  010422  104002                            ERROR   2                   ;CSRA REG ERROR
 6650  010424  012737  000200  002124    201$:   MOV     #200,$GDDAT         ;STORE EXPECTED
 6651  010432  010137  002122                    MOV     R1,$BDADR           ;CSRC
 6652  010436  011137  002126                    MOV     (R1),$BDDAT         ;
 6653  010442  042737  000007  002126            BIC     #7,$BDDAT           ;CLEAR UNDEFINED BITS
 6654  010450  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6655  010456  001401                            BEQ     15$
 6656  010460  104002                            ERROR   2                   ;CSRC ERROR
 6657  010462  012702  000100            15$:    MOV     #64.,R2
 6658  010466  012704  017030                    MOV     #VBUF,R4            ;VECTOR BUFFER
 6659  010472  012737  000300  002124            MOV     #300,$GDDAT         ;START WITH FIRST VECTOR 300
 6660  010500  010437  002122            16$:    MOV     R4,$BDADR           ;VECTOR BUFFER ADDRESS
 6661  010504  011437  002126                    MOV     (R4),$BDDAT
 6662  010510  023737  002124  002126            CMP     $GDDAT,$BDDAT
 6663  010516  001401                            BEQ     17$                 ;VECTORED PROPERLY
 6664  010520  104007                            ERROR   7                   ;VECTOR ADDR MEM ERROR
 6665  010522  062737  000004  002124    17$:    ADD     #4,$GDDAT
 6666  010530  005737  017302                    TST     KXTFLAG             ; KXT11 ??                          ;;GPA
 6667  010534  001407                            BEQ     1000$               ; BR IF NOT                         ;;GPA
 6668  010536  023727  002124  000400            CMP     $GDDAT,#400         ; YES, REACHED VECTOR 400 ??        ;;GPA
 6669  010544  103403                            BLO     1000$               ; NOT YET, CONTINUE                 ;;GPA
 6670  010546  012737  000300  002124            MOV     #300,$GDDAT         ; YES, RESET EXPECTED TO 300        ;;GPA
 6671  010554                            1000$:                                                                  ;;GPA
 6672  010554  062704  000002                    ADD     #2,R4               ;NEXT BUFFER ADDRESS
 6673  010560  005302                            DEC     R2                  ;COMPLETED 64 BUFFER CHECKS?
 6674  010562  001346                            BNE     16$                 ;CHECK NEXT VECTOR BUFFER LOCATION
 6675                                                                        ;THAT WAS STORED BY THE INTERRUPT
```

```
                                                        ;SERVICE ROUTINES.


                                ;DON'T REPORT 'END OF PASS' UNTIL ALL SELECTED DRV11J'S HAVE BEEN TESTED.
010564  013701  002412          NXDEV:  MOV     DRCSA,R1        ;INIT TO SETUP DRV11J ADDRESS
010570  000241                  NXDEV1: CLC                     ;CLEAR CARRY FOR DEVICE MAP
010572  006037  002434                  ROR     DMAP            ;LOOK FOR NEXT DRV11J
010576  001412                          BEQ     $EOP            ;BR IF ALL TESTED
010600  162701  000020                  SUB     #20,R1          ;NEXT DRV11J STARTS -20 FROM PAST CSR
010604  005237  002202                  INC     $UNIT           ;UPDAT UNIT #
010610  032737  000001  002434          BIT     #1,DMAP         ;IS UNIT SELECTED?
010616  001764                          BEQ     NXDEV1          ;BR IF NOT
010620  000137  003132                  JMP     NEXPAS          ;TEST NEXT DRV11J

                                .SBTTL  END OF PASS ROUTINE
                                ;;**************************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO START1

010624                          $EOP:
010624  000240                          NOP
010626  005037  002102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
010632  005037  002160                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
010636  005237  002176                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
010642  042737  100000  002176          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
010650  005327                          DEC     (PC)+           ;;LOOP?
010652  000001                  $EOPCT: .WORD   1
010654  003022                          BGT     $DOAGN          ;;YES
010656  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
010660  000001                  $ENDCT: .WORD   1
010662  010652                          $EOPCT
010664  104401  010731                  TYPE    ,$ENDMG         ;;TYPE "END PASS #"
010670  013746  002176                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
010674  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
010676  104401  010726                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
010702  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
010706  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
010710  000005                          RESET                   ;;CLEAR THE WORLD
010712  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
010714  000240                          NOP                     ;;SAVE ROOM
010716  000240                          NOP                     ;;FOR
010720  000240                          NOP                     ;;ACT11
010722                          $DOAGN:
010722  000137                          JMP     @(PC)+          ;;RETURN
010724  003066                  $RTNAD: .WORD   START1


010726     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
010731     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
010736  050040  051501  020123
010744  000043
```

```
                                .SBTTL  PROGRAM SUBROUTINES


                                ;;*****************************************************************
                                ;CLEAR ALL CONTROL/STATUS REGISTERS
                                ;;*****************************************************************
010746  010046                  CLRCSR: MOV     R0,-(SP)        ;SAVE R0
010750  106427  000300                  MTPS    #PR6    ;VER:0          ;PSW = 300
010754  012737  000340  002456          MOV     #PVMA,VECVAL    ;INIT VECTOR ADDR MEM = 340
010762  012737  000300  002452          MOV     #300,VECLOC     ;INIT VECTOR ADDR TO 300
010770  013700  002412                  MOV     DRCSA,R0        ;START OF CSR ADDRESS
010774  005037  002124                  CLR     $GDDAT          ;CLEAR EXPECTED
011000  005037  002126                  CLR     $BDDAT          ;CLEAR REC'D
011004  005010                          CLR     (R0)            ;CLEAR CSRA;CHIP RESET GROUP 1
011006  105060  000005                  CLRB    5(R0)           ;CLEAR HIGH BYTE CSRB
011012  005060  000010                  CLR     10(R0)          ;CLEAR CSRC;CHIP RESET GROUP 2
011016  105060  000015                  CLRB    15(R0)          ;CLEAR HIGH BYTE CSRD
011022  012600                          MOV     (SP)+,R0        ;RETURN R0
011024  000205                          RTS     R5

                                ;CLEAR IRR REGISTERS,GROUP 1,GROUP 2 WITH CHIP RESET
011026  010046                  CLRIRR: MOV     R0,-(SP)
011030  013700  002412                  MOV     DRCSA,R0        ;START OF CSR ADDRESS
011034  105060  000011                  CLRB    11(R0)          ;CSRC TO INPUT MODE
011040  112760  000001  000001          MOVB    #BIT0,1(R0)     ;CSRA TO OUTPUT MODE
011046  005060  000002                  CLR     2(R0)           ;CLEAR DBRA
011052  105010                          CLRB    (R0)            ;CHIP RESET OF GROUP1
011054  105060  000010                  CLRB    10(R0)          ;CHIP RESET OF GROUP 2
011060  012600                          MOV     (SP)+,R0        ;RESTORE REGISTER
011062  000205                          RTS     R5              ;EXIT

                                ;ROUTINE TO FILL VECTOR MEMORY FOR VECTOR UNIQUENESS TEST
                                ;300-474        GROUP 1
                                ;500-674        GROUP 2
011064  013700  002412          VECFIL: MOV     DRCSA,R0        ;START GROUP 1
011070  013701  002416                  MOV     DRCSB,R1
011074  012702  000002                  MOV     #2,R2           ;TWO GROUPS
011100  012737  000060  002454          MOV     #60,VECPAT      ;VECTOR START 300
011106  012737  000370  002456  1$:     MOV     #370,VECVAL     ;START VECTOR LEVEL 0,BY4
011114  012737  000010  002466          MOV     #8.,LVLCNT      ;LEVELS 0-7
011122  113710  002456          2$:     MOVB    VECVAL,(R0)
011126  012737  000004  002470          MOV     #4,BYCNT        ;DO FOUR BYTE COUNTS
011134  113711  002454          3$:     MOVB    VECPAT,(R1)     ;WRITE VECTOR
011140  005237  002454                  INC     VECPAT          ;SETUP FOR NEXT VECTOR
011144  005737  017302                  TST     KXTFLAG         ; KXT11 ??                      ;;GPA
011150  001407                          BEQ     1000$           ; BR IF NOT                     ;;GPA
011152  023727  002454  000100          CMP     VECPAT,#100     ; YES, REACED VECTOR 400 ??     ;;GPA
011160  103403                          BLO     1000$           ; NOT YET, CONTINUE             ;;GPA
011162  012737  000060  002454          MOV     #60,VECPAT      ; YES, WRAP-AROUND TO 300 AGAIN ;;GPA
                                                                ; WE'LL END UP WITH 300-374...  ;;GPA
                                                                ;...REPLICATED 4 TIMES.         ;;GPA
011170                          1000$:
011170  005337  002470                  DEC     BYCNT           ;DONE FOUR BYTE COUNTS?
011174  001357                          BNE     3$              ;DO NEXT VECTOR
011176  005337  002466          4$:     DEC     LVLCNT          ;DONE ALL LEVELS IN GROUP?
```

```
                        BEQ     5$
                        INC     VECVAL          ;INC NEXT LEVEL
                        BR      2$              ;DO NEXT LEVEL
                5$:     DEC     R2              ;DONE BOTH GROUPS?
                        BEQ     6$              ;DONE,EXIT
                        MOV     DRCSC,R0        ;DO GROUP 2
                        MOV     DRCSD,R1
                        BR      1$
                6$:     RTS     R5              ;EXIT

                ;ROUTINE TO STORE VECTOR AREA 300-674 WITH INTERRUPT SERVICE
                STRVEC: MOV     #300,R0         ;START AT VECTOR 300
                        MOV     #INT0,R1        ;FIRST SERVICE ROUTINE
                        MOV     #64.,R2         ;STORE 64 SERVICE ROUTINES
                1$:     MOV     R1,(R0)+
                        MOV     #300,(R0)+              ;VER:0
                ;;GPA   ADD     #6,R1           ;SETUP FOR NEXT VECTOR STORAGE
                        ADD     #4,R1           ; SETUP NEXT VETOR.             ;;GPA
                        DEC     R2              ;DONE WITH 64 VECTOR ROUTINES?
                        BNE     1$
                        RTS     R5              ;EXIT

                ;ROUTINE TO CLEAR VECTOR BUFFER
                CLRBF:  MOV     #VBUF,R0        ;VECTOR BUFFER START
                        MOV     #64.,R1
                1$:     CLR     (R0)+
                        DEC     R1
                        BNE     1$
                        RTS     R5

                ;ROUTINE TO STORE 0-202 INTO BUFFER AREA TO ALLOW
                ;VECTOR TESTS TO 0-200.
                TOBUF:  CLR     R0
                        MOV     #DBUF,R1        ;BUFFER START
                1$:     MOV     (R0)+,(R1)+     ;STORE 0-202 INTO BUFFER
                        CMP     #204,R0         ;FINISHED?
                        BNE     1$              ;NO
                        RTS     R5              ;RETURN

                ;STORE BUFFER AREA INTO LOCATIONS 0-202
                ;AFTER VECTOR TESTS
                FRMBUF: MOV     R0,-(SP)        ;SAVE R0
                        MOV     R1,-(SP)
                        CLR     R0
                        MOV     #DBUF,R1        ;BUFFER START
                1$:     MOV     (R1)+,(R0)+     ;STORE BUFFER INTO LOCS 0-202
                        CMP     #204,R0         ;FINISHED?
                        BNE     1$              ;NO
                        MOV     SWRSAV,@SWR     ;STORE LOCATION 176,SOFT SWR
                        MOV     (SP)+,R1        ;RESTORE R1
                        MOV     (SP)+,R0        ;RESTORE R0
                        RTS     R5              ;RETURN

                ;RESTORE VECTOR JUST TESTED TO ORIGINAL TRAP FOR 204-1774
                RESTRP: CMP     #1,$ENV         ;CHECK FOR APT
                        BEQ     1$              ;YES,BRANCH
```

```
 6804  011374  032777  010000  170536          BIT     #SW12,@SWR      ;CHECK TYPE OF TRAP RETURN
 6805  011402  001412                          BEQ     10$             ;RESTORE ILLEGAL VECTOR ROUTINE
 6806  011404  062737  000002  002452          ADD     #2,VECLOC       ;RESTORE TRAP
 6807  011412  013723  002452                  MOV     VECLOC,(R3)+    ;TO VECTOR
 6808  011416  005013                          CLR     (R3)            ;RESTORE HALT
 6809  011420  062737  000002  002452          ADD     #2,VECLOC       ;SETUP FOR NEXT
 6810  011426  000412                          BR      2$              ;RETURN
 6811  011430  012723  011704          10$:    MOV     #TRPALL,(R3)+   ;RESTORE ILLEGAL TRAP ROUTINE
 6812  011434  000402                          BR      11$             ;RESTORE PSW
 6813  011436  012723  011666          1$:     MOV     #TRPOUT,(R3)+   ;RESTORE APT SUB TRAP
 6814                                                                  ;IF ON APT
 6815  011442  012713  000300          11$:    MOV     #300,(R3)       ;RESTORE PSW SAVE :VER:0
 6816  011446  062737  000004  002452          ADD     #4,VECLOC       ;UPDATE FOR NEXT VECTOR
 6817  011454  000205                  2$:     RTS     R5              ;RETURN
 6818
 6819                                  ;ROUTINE TO SET UP TRAP CATCHER 204-1774 IN STANDALONE MODE OR
 6820                                  ;A COMMON VECTOR ROUTINE SETUP UNDER APT.
 6821                                  ;VECTOR ROUTINE IS USED ONLY IN AN APT ENVIRONMENT AND WILL STORE
 6822                                  ;A COMMON SUBROUTINE IN VECTOR AREA 204-1774.
 6823
 6824  011456  010046                  TRPCAT: MOV     R0,-(SP)        ;SAVE R0
 6825  011460  010146                          MOV     R1,-(SP)
 6826  011462  022737  000001  002210          CMP     #1,$ENV         ;CHECK FOR APT
 6827  011470  001426                          BEQ     APTVEC          ;YES,SET UP TRAPS FOR APT
 6828  011472  012700  000204                  MOV     #204,R0         ;START CATCHER AT 204
 6829  011476  032777  010000  170434          BIT     #SW12,@SWR      ;TEST IF SW12 IS SET
 6830  011504  001010                          BNE     1$              ;YES,BR AND STORE REGULAR TRAP
 6831                                                                  ;CATCHER(.+2,HALT) IN LOCATIONS
 6832                                                                  ;204-1774
 6833  011506  012720  011704          11$:    MOV     #TRPALL,(R0)+   ;NO,PLACE ILLEGAL VECTOR RETURN
 6834  011512  012720  000300                  MOV     #300,(R0)+      ;ROUTINES IN LOCATIONS 204-1774;VER:0
 6835  011516  022700  002000                  CMP     #2000,R0        ;STORED ALL VECTORS?
 6836  011522  001371                          BNE     11$
 6837  011524  000421                          BR      ENDTRP          ;RETURN AFTER VECTOR STORAGE
 6838  011526  010001                  1$:     MOV     R0,R1           ;DRV11J CAN VECTOR 0-1774
 6839  011530  005721                          TST     (R1)+
 6840  011532  010120                          MOV     R1,(R0)+
 6841  011534  005020                          CLR     (R0)+
 6842  011536  022701  001776                  CMP     #1776,R1        ;CHECK FOR VECTOR END
 6843  011542  001371                          BNE     1$
 6844  011544  000411                          BR      ENDTRP          ;RETURN IN STAND ALONE MODE
 6845  011546  012701  000204          APTVEC: MOV     #204,R1         ;STARTING VECTOR ADDRESS
 6846  011552  012721  011666          1$:     MOV     #TRPOUT,(R1)+   ;ILL INTERRUPT ROUTINE
 6847  011556  012721  000300                  MOV     #300,(R1)+      ;PSW NEXT;VER:0
 6848  011562  022701  002000                  CMP     #2000,R1        ;DONE?
 6849  011566  001371                          BNE     1$              ;DO NEXT VECTOR
 6850  011570  012601                  ENDTRP: MOV     (SP)+,R1
 6851  011572  012600                          MOV     (SP)+,R0
 6852  011574  000205                          RTS     R5              ;EXIT
 6853
 6854                                  ;INTERRUPT SERVICE ROUTINE USED TO VERIFY INTERRUPTS 204-1774
 6855  011576  005237  002436          INTSR1: INC     INTFLG          ;COUNT INTERRUPT
 6856  011602  012737  000001  002124          MOV     #1,$GDDAT       ;STORE EXPECTED
 6857  011610  013737  002436  002126          MOV     INTFLG,$BDDAT   ;SAVE INT. COUNT
 6858  011616  023737  002124  002126          CMP     $GDDAT,$BDDAT   ;SHOULD BE ONE INTERRUPT
 6859  011624  001401                          BEQ     1$
```

```
                                        ERROR   13              ;MULTIPLE INTERRUPTS RECEIVED
                                1$:     RTI                     ;RETURN FROM INT.

                                ;INTERRUPT SERVICE ROUTINE USED TO VERIFY INTERRUPTS 204-1774
                                ;FOR A BYTE COUNT OF 1 TO 4 VECTORS IN VECTOR ADDR MEMORY.
                                INTBY4: INC     INTFLG          ;COUNT INTERRUPT
                                        MOV     BYNUM,$GDDAT    ;STORE EXPECTED
                                        MOV     INTFLG,$BDDAT   ;SAVE INT. COUNT
                                        CMP     $GDDAT,$BDDAT   ;INTERRUPTS SHOULD NOT EXCEED BYTE COUNT
                                        BGE     1$
                                        ERROR   13              ;MULTIPLE INTERRUPTS RECEIVED
                                                                ;MORE INTERRUPTS THAN BYTE COUNT
                                1$:     RTI                     ;RETURN FROM INT.

                                ;COMMON ILLEGAL VECTOR RETURN ROUTINE FOR APT MODE.
                                TRPOUT: MOV     (SP),$GDADR     ;SAVE ADDRESS OF TEST
                                        JSR     R5,FRMBUF       ;RETURN LOCATIONS 0-202
                                        ERROR   14              ;ILLEGAL VECTOR ADDR MEM ERROR
                                        HALT                    ;CANNOT CONTINUE
                                        HALT                    ;CANNOT CONTINUE

                                ;COMMON ILLEGAL VECTOR RETURN ROUTINE FOR VECTOR AREA
                                ;204-1774
                                TRPALL: MOV     (SP),$GDADR     ;SAVE PC OF TEST
                                        ERROR   14              ;ILLEGAL VECTOR ADDR MEM ERROR
                                        RTI                     ;RETURN

                                ;RESTORE VECTOR JUST TESTED TO ERROR SUBROUTINE FOR 0-202
                                RES200: CMP     #1,$ENV         ;CHECK FOR APT
                                        BEQ     1$              ;YES,BRANCH
                                        MOV     #TRP200,(R3)+   ;ERROR TRAP SUB. FOR 0-200
                                        MOV     #300,(R3)       ;PSW;VER:0
                                        BR      2$              ;RETURN
                                1$:     MOV     #TRPOUT,(R3)+   ;RESTORE APT SUB TRAP
                                                                ;IF ON APT
                                        MOV     #300,(R3)       ;RESTORE PSW SAVE;VER:0
                                2$:     ADD     #4,VECLOC       ;UPDATE FOR NEXT VECTOR
                                        RTS     R5              ;RETURN

                                ;ROUTINE TO SET UP TRAP SUBROUTINE 0-200 IN STANDALONE MODE OR
                                ;A COMMON ERROR VECTOR ROUTINE SETUP UNDER APT.

                                CAT200: MOV     R0,-(SP)        ;SAVE R0
                                        MOV     @SWR,SWRSAV     ;SAVE LOCATION 176,SOFT SWR
                                        CMP     #1,$ENV         ;CHECK FOR APT
                                        BEQ     APT200          ;YES,SET UP TRAPS FOR APT
                                        CLR     R0              ;START CATCHER SUB AT 0
                                1$:     MOV     #TRP200,(R0)+   ;STORE VECTOR TRAP SUBROUTINE
                                        MOV     #300,(R0)+      ;PSW;VER:0
                                        CMP     #204,R0         ;FINISHED 0-202
                                        BNE     1$
                                        BR      END200          ;RETURN IN STAND ALONE MODE
                                APT200: CLR     R0              ;START AT VECTOR 0
                                1$:     MOV     #TRPOUT,(R0)+   ;VECTOR ERROR ROUTINE
                                        MOV     #300,(R0)+      ;PSW NEXT;VER:0
```

```
                                                                H 3
 6916  012032  022700  000204                    CMP     #204,R0         ;DONE?
 6917  012036  001371                            BNE     1$              ;DO NEXT VECTOR
 6918  012040  012600                    END200: MOV     (SP)+,R0
 6919  012042  012713  012056                    MOV     #INTSR3,(R3)    ;STORE VECTOR ROUTINE
 6920  012046  012763  000300  000002            MOV     #300,2(R3)      ;STORE PSW;VER:0
 6921  012054  000205                            RTS     R5              ;EXIT
 6922
 6923
 6924                                    ;INTERRUPT SERVICE ROUTINE USED TO VERIFY INTERRUPTS 0-200,BYTE COUNT 0-3
 6925                                    ;FOR BYTE COUNTS OF 1 TO 4 VECTORS IN VECTOR ADDRESS MEMORY.
 6926  012056  005237  002436            INTSR3: INC     INTFLG          ;COUNT INTERRUPT
 6927  012062  013737  002472  002124            MOV     BYNUM,$GDDAT    ;STORE EXPECTED
 6928  012070  013737  002436  002126            MOV     INTFLG,$BDDAT   ;SAVE INT. COUNT
 6929  012076  023737  002124  002126            CMP     $GDDAT,$BDDAT   ;INTERRUPTS SHOULD NOT EXCEED BYTE COUNT
 6930  012104  002005                            BGE     1$
 6931  012106  004537  011326                    JSR     R5,FRMBUF       ;RESTORE 0-202 BEFORE ERROR
 6932  012112  104013                            ERROR   13              ;MULTIPLE INTERRUPTS RECEIVED
 6933                                                                    ;MORE INTERRUPTS THAN BYTE COUNT
 6934  012114  004537  011756                    JSR     R5,CAT200       ;RESTORE VECTOR SUBROUTINES
 6935  012120  000002                    1$:     RTI                     ;RETURN FROM INT.
 6936
 6937                                    ;COMMON ILLEGAL VECTOR ERROR RETURN ROUTINE FOR 0-200
 6938  012122  011637  002120            TRP200: MOV     (SP),$GDADR     ;SAVE PC OF TEST
 6939  012126  004537  011326                    JSR     R5,FRMBUF       ;RETURN LOC 0-200
 6940  012132  104014                            ERROR   14              ;ILLEGAL VECTOR ADDR MEM ERROR
 6941  012134  004537  011756                    JSR     R5,CAT200       ;RESTORE VECTOR SUBROUTINES FOR PROCEED
 6942  012140  000002                            RTI                     ;RETURN
 6943
 6944                                    ;VECTOR SERVICE ROUTINE FOR VECTOR UNIQUENESS TEST
 6945                                    ;SETVEC:        MOV     (R5)+,(R4)+     ;STORE VALUE IN BUFFER  ;;GPA
 6946  012142  017624  000000            SETVEC: MOV     @(SP),(R4)+     ; STORE VECTOR NUMBER IN BUFFER ;;GPA
 6947  012146  005726                            TST     (SP)+           ;RESTORE STACK FOR RTI
 6948  012150  005237  002436                    INC     INTFLG          ;COUNT INTERRUPT
 6949  012154  022737  000100  002436            CMP     #64.,INTFLG
 6950  012162  002001                            BGE     1$              ;64 EXPECTED INTERRUPTS?
 6951  012164  104013                            ERROR   13              ;ERROR ,MORE THAN 64 INTERRUPTS
 6952  012166  000002                    1$:     RTI                     ;RETURN FOR NEXT INTERRUPT
 6953
 6954                                    ;INTERRUPT SERVICE ROUTINES THAT ARE STORED FROM 300-674
 6955                                    ;IN THE VECTOR UNIQUENESS TEST.
 6956                                    ; R5 POINTS TO "SETVEC:" DURING TESTS 5 AND 6, AND CALLS        ;;GPA
 6957                                    ; CHANGED FROM "JSR R5,SETVEC" TO "JSR PC,(R5)" TO CONSERVE     ;;GPA
 6958                                    ; SPACE AND REMAIN UNDER 4KW TOTAL PROGRAM SIZE.                ;;GPA
 6959  012170  004715                    INT0:   JSR     PC,(R5) ;STORE 300 INTO VBUF,TO BE              ;;GPA
 6960  012172  000300                            300             ;CHECKED AFTER ALL INTERRUPTS.
 6961  012174  004715                    INT1:   JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6962  012176  000304                            304
 6963  012200  004715                    INT2:   JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6964  012202  000310                            310
 6965  012204  004715                    INT3:   JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6966  012206  000314                            314
 6967  012210  004715                    INT4:   JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6968  012212  000320                            320
 6969  012214  004715                    INT5:   JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6970  012216  000324                            324
 6971  012220  004715                    INT6:   JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
```

```
 6972  012222  000330                        330
 6973  012224  004715              INT7:     JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6974  012226  000334                        334
 6975  012230  004715              INT8:     JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6976  012232  000340                        340
 6977  012234  004715              INT9:     JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6978  012236  000344                        344
 6979  012240  004715              INT10:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6980  012242  000350                        350
 6981  012244  004715              INT11:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6982  012246  000354                        354
 6983  012250  004715              INT12:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6984  012252  000360                        360
 6985  012254  004715              INT13:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6986  012256  000364                        364
 6987  012260  004715              INT14:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6988  012262  000370                        370
 6989  012264  004715              INT15:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6990  012266  000374                        374
 6991  012270  004715              INT16:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6992  012272  000400                        400
 6993  012274  004715              INT17:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6994  012276  000404                        404
 6995  012300  004715              INT18:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6996  012302  000410                        410
 6997  012304  004715              INT19:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 6998  012306  000414                        414
 6999  012310  004715              INT20:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7000  012312  000420                        420
 7001  012314  004715              INT21:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7002  012316  000424                        424
 7003  012320  004715              INT22:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7004  012322  000430                        430
 7005  012324  004715              INT23:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7006  012326  000434                        434
 7007  012330  004715              INT24:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7008  012332  000440                        440
 7009  012334  004715              INT25:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7010  012336  000444                        444
 7011  012340  004715              INT26:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7012  012342  000450                        450
 7013  012344  004715              INT27:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7014  012346  000454                        454
 7015  012350  004715              INT28:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7016  012352  000460                        460
 7017  012354  004715              INT29:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7018  012356  000464                        464
 7019  012360  004715              INT30:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7020  012362  000470                        470
 7021  012364  004715              INT31:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7022  012366  000474                        474
 7023  012370  004715              INT32:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7024  012372  000500                        500
 7025  012374  004715              INT33:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
 7026  012376  000504                        504
 7027  012400  004715              INT34:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
```

```
7028  012402  000510                    510
7029  012404  004715          INT35:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7030  012406  000514                    514
7031  012410  004715          INT36:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7032  012412  000520                    520
7033  012414  004715          INT37:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7034  012416  000524                    524
7035  012420  004715          INT38:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7036  012422  000530                    530
7037  012424  004715          INT39:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7038  012426  000534                    534
7039  012430  004715          INT40:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7040  012432  000540                    540
7041  012434  004715          INT41:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7042  012436  000544                    544
7043  012440  004715          INT42:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7044  012442  000550                    550
7045  012444  004715          INT43:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7046  012446  000554                    554
7047  012450  004715          INT44:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7048  012452  000560                    560
7049  012454  004715          INT45:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7050  012456  000564                    564
7051  012460  004715          INT46:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7052  012462  000570                    570
7053  012464  004715          INT47:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7054  012466  000574                    574
7055  012470  004715          INT48:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7056  012472  000600                    600
7057  012474  004715          INT49:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7058  012476  000604                    604
7059  012500  004715          INT50:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7060  012502  000610                    610
7061  012504  004715          INT51:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7062  012506  000614                    614
7063  012510  004715          INT52:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7064  012512  000620                    620
7065  012514  004715          INT53:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7066  012516  000624                    624
7067  012520  004715          INT54:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7068  012522  000630                    630
7069  012524  004715          INT55:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7070  012526  000634                    634
7071  012530  004715          INT56:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7072  012532  000640                    640
7073  012534  004715          INT57:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7074  012536  000644                    644
7075  012540  004715          INT58:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7076  012542  000650                    650
7077  012544  004715          INT59:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7078  012546  000654                    654
7079  012550  004715          INT60:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7080  012552  000660                    660
7081  012554  004715          INT61:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7082  012556  000664                    664
7083  012560  004715          INT62:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
```

```
7084  012562  000670                    670
7085  012564  004715          INT63:    JSR     PC,(R5) ;VECTOR UNIQUENESS TEST, INT. SERV RTN  ;;GPA
7086  012566  000674                    674
7087
7088                          .SBTTL  PATTERNS FOR REGISTER R/W
7089                          ;
7090                          ;PATTERNS USED FOR LOADING/READING REGISTERS
7091                          ;ISR    INTERRUPT SERVICE REGISTER
7092                          ;IRR    INTERRUPT REQUEST REGISTER
7093                          ;IMR    INTERRUPT MASK    REGISTER
7094                          ;
7095  012570     001     376  BGCHP3:  .BYTE   1,376
7096  012572     002     375           .BYTE   2,375
7097  012574     004     373           .BYTE   4,373
7098  012576     010     367           .BYTE   10,367
7099  012600     020     357           .BYTE   20,357
7100  012602     040     337           .BYTE   40,337
7101  012604     100     277           .BYTE   100,277
7102  012606     200     177           .BYTE   200,177
7103  012610  000000          EDCHP3:  000000
7104
7105
```

```
 7107                                   .SBTTL  SYSMAC ROUTINES
 7108
 7109                                   .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)  012612  105737  002157           $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
  (1)  012616  100002                           BPL     1$              ;;BR IF YES
  (1)  012620  000000                           HALT                    ;;HALT HERE IF NO TERMINAL
  (1)  012622  000430                           BR      3$              ;;LEAVE
  (1)  012624  010046                   1$:     MOV     R0,-(SP)        ;;SAVE R0
  (1)  012626  017600  000002                   MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
  (1)  012632  122737  000001  002210           CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
  (1)  012640  001011                           BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
  (1)  012642  132737  000100  002211           BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
  (1)  012650  001405                           BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
  (1)  012652  010037  012662                   MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
  (1)  012656  004737  013102                   JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
  (1)  012662  000000                   61$:    .WORD   0               ;;MESSAGE ADDRESS
  (1)  012664  132737  000040  002211   62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
  (1)  012672  001003                           BNE     60$             ;;YES,SKIP TYPE OUT
  (1)  012674  112046                   2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
  (1)  012676  001005                           BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
  (1)  012700  005726                           TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
  (1)  012702  012600                   60$:    MOV     (SP)+,R0        ;;RESTORE R0
  (1)  012704  062716  000002           3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
  (1)  012710  000002                           RTI                     ;;RETURN
  (1)  012712  122716  000011           4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
  (1)  012716  001430                           BEQ     8$
  (1)  012720  122716  000200                   CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
  (1)  012724  001006                           BNE     5$
  (1)  012726  005726                           TST     (SP)+           ;;POP  <CR><LF> EQUIV
  (1)  012730  104401                           TYPE                    ;;TYPE A CR AND LF
  (1)  012732  002165                           $CRLF
  (1)  012734  105037  013070                   CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
  (1)  012740  000755                           BR      2$              ;;GET NEXT CHARACTER
  (1)  012742  004737  013024           5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
  (1)  012746  123726  002156           6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
  (1)  012752  001350                           BNE     2$              ;;IF NO GO GET NEXT CHAR.
  (1)  012754  013746  002154                   MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
  (1)                                                                   ;;AND THE NULL CHAR.
  (1)  012760  105366  000001           7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
  (1)  012764  002770                           BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
```

```
   (1)  012766  004737  013024                 JSR     PC,$TYPEC       ;;GO TYPE A NULL
   (1)  012772  105337  013070                 DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
   (1)  012776  000770                         BR      7$              ;;LOOP
   (1)
   (1)                                 ;HORIZONTAL TAB PROCESSOR
   (1)
   (1)  013000  112716  000040          8$:    MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
   (1)  013004  004737  013024          9$:    JSR     PC,$TYPEC       ;;TYPE A SPACE
   (1)  013010  132737  000007  013070         BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
   (1)  013016  001372                         BNE     9$              ;;TAB STOP
   (1)  013020  005726                         TST     (SP)+           ;;POP SPACE OFF STACK
   (1)  013022  000724                         BR      2$              ;;GET NEXT CHARACTER
   (1)  013024  105777  167120          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
   (1)  013030  100375                         BPL     $TYPEC
   (1)  013032  116677  000002  167112         MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
   (1)
   (1)  013040  122766  000015  000002  CMPB   #CR,2(SP)        ;;IS CHARACTER A CARRIAGE RETURN?
   (1)  013046  001003                         BNE     1$              ;;BRANCH IF NO
   (1)  013050  105037  013070                 CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
   (1)  013054  000406                         BR      $TYPEX          ;;EXIT
   (1)  013056  122766  000012  000002  1$:    CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
   (1)  013064  001402                         BEQ     $TYPEX          ;;BRANCH IF YES
   (1)  013066  105227                         INCB    (PC)+           ;;COUNT THE CHARACTER
   (1)  013070  000000                 $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
   (1)  013072  000207                 $TYPEX: RTS     PC
   (1)
   (1)
   (1)
7110                                   .SBTTL  APT COMMUNICATIONS ROUTINE
  (1)
  (2)                                  ;;*****************************************************************************
  (1)  013074  112737  000001  013340  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
  (1)  013102  112737  000001  013336  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
  (1)  013110  000403                         BR      $ATYC
  (1)  013112  112737  000001  013340  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
  (1)  013120                          $ATYC:
  (3)  013120  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)  013122  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (1)  013124  105737  013336                 TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
  (1)  013130  001450                         BEQ     5$              ;;IF NOT:  BR
  (1)  013132  122737  000001  002210         CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
  (1)  013140  001031                         BNE     3$              ;;IF NOT:  BR
  (1)  013142  132737  000100  002211         BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
  (1)  013150  001425                         BEQ     3$              ;;IF NOT:  BR
  (1)  013152  017600  000004                 MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
  (1)  013156  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
  (1)  013164  005737  002170          1$:    TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
  (1)  013170  001375                         BNE     1$              ;;IF NOT:  WAIT
  (1)  013172  010037  002204                 MOV     R0,$MSGAD
  (1)                                  ;;PUT ADDR IN MAILBOX
  (1)  013176  105720                  2$:    TSTB    (R0)+           ;;FIND END OF MESSAGE
  (1)  013200  001376                         BNE     2$
  (1)  013202  163700  002204                 SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
  (1)  013206  006200                         ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
  (1)  013210  010037  002206                 MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
  (1)  013214  012737  000004  002170         MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
```

```
(1)  013222  000413                        BR      5$
(1)  013224  017637  000004  013250  3$:   MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
(1)  013232  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
(3)  013240  013746  177776                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
(1)  013244  004737  012612                JSR     PC,$TYPE        ;;CALL TYPE MACRO
(1)  013250  000000                 4$:    .WORD   0
(1)  013252                         5$:
(1)  013252  105737  013340        10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
(1)  013256  001416                        BEQ     12$             ;;IF NOT:  BR
(1)  013260  005737  002210                TST     $ENV            ;;RUNNING UNDER APT?
(1)  013264  001413                        BEQ     12$             ;;IF NOT:  BR
(1)  013266  005737  002170        11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(1)  013272  001375                        BNE     11$             ;;IF NOT:  WAIT
(1)  013274  017637  000004  002172        MOV     @4(SP),$FATAL   ;;GET ERROR #
(1)  013302  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
(1)  013310  005237  002170                INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(1)  013314  105037  013340        12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
(1)  013320  105037  013337                CLRB    $LFLG           ;;CLEAR LOG FLAG
(1)  013324  105037  013336                CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
(3)  013330  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  013332  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  013334  000207                        RTS     PC              ;;RETURN
(1)  013336     000         $MFLG:  .BYTE   0               ;;MESSG. FLAG
(1)  013337     000         $LFLG:  .BYTE   0
(1)                                         ;;LOG FLAG
(1)  013340     000         $FFLG:  .BYTE   0               ;;FATAL FLAG
(1)          013342                 .EVEN
(1)          000200         APTSIZE=200
(1)          000001         APTENV=001
(1)          000100         APTSPOOL=100
(1)          000040         APTCSUP=040
7111                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(2)                         ;;**********************************************************************
(1)                         ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1)                         ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(1)                         ;*$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)  013342  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  013346  116637  000001  013565          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  013354  112637  013567                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  013360  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  013364  000406                          BR      $TYPON
(1)  013366  112737  000001  013565  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
(1)  013374  112737  000006  013567          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  013402  112737  000005  013564  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(1)  013410  010346                          MOV     R3,-(SP)        ;;SAVE R3
(1)  013412  010446                          MOV     R4,-(SP)        ;;SAVE R4
(1)  013414  010546                          MOV     R5,-(SP)        ;;SAVE R5
(1)  013416  113704  013567                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  013422  005404                          NEG     R4
(1)  013424  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  013430  110437  013566                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  013434  113704  013565                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  013440  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  013444  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  013446  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  013450  000404                          BR      3$              ;;GO DO MSB
(1)  013452  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
(1)  013454  006105                          ROL     R5
(1)  013456  006105                          ROL     R5
(1)  013460  010503                          MOV     R5,R3
(1)  013462  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  013464  105337  013566                  DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  013470  100016                          BPL     7$              ;;BR IF NO
(1)  013472  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
(1)  013476  001002                          BNE     4$              ;;TEST FOR 0
(1)  013500  005704                          TST     R4              ;;SUPPRESS THIS 0?
(1)  013502  001403                          BEQ     5$              ;;BR IF YES
(1)  013504  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  013506  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  013512  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  013516  110337  013562                  MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  013522  104401  013562                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  013526  105337  013564          7$:     DECB    $OCNT           ;;COUNT BY 1
(1)  013532  003347                          BGT     2$              ;;BR IF MORE TO DO
(1)  013534  002402                          BLT     6$              ;;BR IF DONE
(1)  013536  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  013540  000744                          BR      2$              ;;GO DO THE LAST DIGIT
(1)  013542  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
(1)  013544  012604                          MOV     (SP)+,R4        ;;RESTORE R4
(1)  013546  012603                          MOV     (SP)+,R3        ;;RESTORE R3
(1)  013550  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  013556  012616                          MOV     (SP)+,(SP)
(1)  013560  000002                          RTI                     ;;RETURN
(1)  013562     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  013563     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  013564     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(1)  013565     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(1)  013566  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
7112                                 .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)  013570                         $TYPDS:
(3)  013570  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  013572  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  013574  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  013576  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  013600  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  013602  012746  020200                 MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)  013606  016605  000020                 MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)  013612  100004                         BPL     1$              ;;BR IF INPUT IS POS.
(1)  013614  005405                         NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)  013616  112766  000055  000001         MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  013624  005000                 1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  013626  012703  014004                 MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  013632  112723  000040                 MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  013636  005002                 2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  013640  016001  013774                 MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  013644  160105                 3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  013646  002402                         BLT     4$              ;;BR IF DONE
(1)  013650  005202                         INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  013652  000774                         BR      3$
(1)  013654  060105                 4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  013656  005702                         TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  013660  001002                         BNE     5$              ;;FALL THROUGH IF 0
(1)  013662  105716                         TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)  013664  100407                         BMI     7$              ;;BR IF YES
(1)  013666  106316                 5$:     ASLB    (SP)            ;;MSD?
(1)  013670  103003                         BCC     6$              ;;BR IF NO
(1)  013672  116663  000001  177777         MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
(1)  013700  052702  000060         6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)  013704  052702  000040         7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  013710  110223                         MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  013712  005720                         TST     (R0)+           ;;JUST INCREMENTING
(1)  013714  020027  000010                 CMP     R0,#10          ;;CHECK THE TABLE INDEX
(1)  013720  002746                         BLT     2$              ;;GO DO THE NEXT DIGIT
(1)  013722  003002                         BGT     8$              ;;GO TO EXIT
(1)  013724  010502                         MOV     R5,R2           ;;GET THE LSD
(1)  013726  000764                         BR      6$              ;;GO CHANGE TO ASCII
(1)  013730  105726                 8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  013732  100003                         BPL     9$              ;;BR IF NO
(1)  013734  116663  177777  177776         MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)  013742  105013                 9$:     CLRB    (R3)            ;;SET THE TERMINATOR
(3)  013744  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  013746  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  013750  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  013752  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  013754  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  013756  104401  014004                 TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
(1)  013762  016666  000002  000004         MOV     2(SP),4(SP)     ;;ADJUST THE STACK
```

```
(1)  013770  012616                    MOV     (SP)+,(SP)
(1)  013772  000002                    RTI                         ;;RETURN TO USER
(1)  013774  023420            $DTBL:  10000.
(1)  013776  001750                    1000.
(1)  014000  000144                    100.
(1)  014002  000012                    10.
(1)  014004  000004            $DBLK:  .BLKW   4
7113                           .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 SWRCK ON ERROR
(1)                            ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                            ;*SW15=1        HALT ON ERROR
(1)                            ;*SW13=1        INHIBIT ERROR TYPEOUTS
(1)                            ;*SW09=1        LOOP ON ERROR
(1)                            ;*CALL
(1)                            ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(1)
(1)  014014                    $ERROR:
(1)  014014  104407                    CKSWR                       ;;TEST FOR CHANGE IN SOFT-SWR
(2)  014016  104407                    CKSWR                   ;GO LOOK FOR SWR CHANGE
(1)  014020  105237  002103    7$:     INCB    $ERFLG              ;;SET THE ERROR FLAG
(1)  014024  001775                    BEQ     7$                  ;;DON'T LET THE FLAG GO TO ZERO
(1)  014026  013777  002102  166106    MOV     $TSTNM,@DISPLAY     ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  014034  005237  002112            INC     $ERTTL              ;;INC THE ERROR COUNT
(1)  014040  011637  002116            MOV     (SP),$ERRPC         ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  014044  162737  000002  002116    SUB     #2,$ERRPC
(1)  014052  117737  166040  002114    MOVB    @$ERRPC,$ITEMB      ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  014060  032777  020000  166052    BIT     #BIT13,@SWR         ;;SKIP TYPEOUT IF SET
(1)  014066  001004                    BNE     20$                 ;;SKIP TYPEOUTS
(1)  014070  004737  014170            JSR     PC,SWRCK            ;;GO TO USER ERROR ROUTINE
(1)  014074  104401  002165            TYPE    ,$CRLF
(1)  014100                    20$:
(1)  014100  122737  000001  002210    CMPB    #APTENV,$ENV        ;;RUNNING IN APT MODE
(1)  014106  001007                    BNE     2$                  ;;NO,SKIP APT ERROR REPORT
(1)  014110  113737  002114  014122    MOVB    $ITEMB,21$          ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  014116  004737  013112            JSR     PC,$ATY4            ;;REPORT FATAL ERROR TO APT
(1)  014122     000            21$:    .BYTE   0
(1)  014123     000                    .BYTE   0
(1)  014124  000777            22$:    BR      22$                 ;;APT ERROR LOOP
(1)  014126  005777  166006    2$:     TST     @SWR                ;;HALT ON ERROR
(1)  014132  100002                    BPL     3$                  ;;SKIP IF CONTINUE
(1)  014134  000000                    HALT                        ;;HALT ON ERROR!
(1)  014136  104407                    CKSWR                       ;;TEST FOR CHANGE IN SOFT-SWR
(1)  014140  032777  001000  165772    BIT     #BIT09,@SWR         ;;LOOP ON ERROR SWITCH SET?
(1)  014146  001402                    BEQ     4$                  ;;BR IF NO
(1)  014150  013716  002110            MOV     $LPERR,(SP)         ;;FUDGE RETURN FOR LOOPING
(1)  014154  005737  002162    4$:     TST     $ESCAPE             ;;CHECK FOR AN ESCAPE ADDRESS
(1)  014160  001402                    BEQ     5$                  ;;BR IF NONE
(1)  014162  013716  002162            MOV     $ESCAPE,(SP)        ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  014166                    5$:
(1)  014166  000002                    RTI                         ;;RETURN
7114                           ;;**********************************************************************
7115                           ;GO TYPE ERROR
```

```
                                        ;GO UPDATE SOFTWARE SWR IF 'CNTRL/G'
                                        ;;*****************************************************************
014170  113737  002102  002432  SWRCK:  MOVB    $TSTNM,TSTNUM   ;SET UP TEST # ON ER
014176  004737  014206                  JSR     PC,$ERRTYP      ;GO TYPE ERROR
014202  104407                          CKSWR                   ;GO LOOK FOR SWR CHANGE
014204  000207                          RTS     PC              ;RETURN TO ERROR HANDLER
                                        .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)  014206                            $ERRTYP:
 (1)  014206  104401  002165                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  014212  010046                            MOV     R0,-(SP)        ;;SAVE R0
 (1)  014214  005000                            CLR     R0              ;;PICKUP THE ITEM INDEX
 (1)  014216  153700  002114                    BISB    @#$ITEMB,R0
 (1)  014222  001004                            BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
 (1)                                            ;;TYPE THE PC OF THE ERROR
 (2)  014224  013746  002116                    MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
 (2)                                            ;;ERROR ADDRESS
 (2)  014230  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 (1)  014232  000426                            BR      6$              ;;GET OUT
 (1)  014234  005300                    1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
 (1)  014236  006300                            ASL     R0              ;;      WORK FOR THE ERROR TABLE
 (1)  014240  006300                            ASL     R0
 (1)  014242  006300                            ASL     R0
 (1)  014244  062700  002252                    ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
 (1)  014250  012037  014260                    MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
 (1)  014254  001404                            BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
 (1)  014256  104401                            TYPE                    ;;TYPE THE "ERROR MESSAGE"
 (1)  014260  000000                    2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
 (1)  014262  104401  002165                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  014266  012037  014276            3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
 (1)  014272  001404                            BEQ     5$              ;;SKIP TYPEOUT IF 0
 (1)  014274  104401                            TYPE                    ;;TYPE THE "DATA HEADER"
 (1)  014276  000000                    4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
 (1)  014300  104401  002165                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  014304  011000                    5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
 (1)  014306  001004                            BNE     7$              ;;GO TYPE THE DATA
 (1)  014310  012600                    6$:     MOV     (SP)+,R0        ;;RESTORE R0
 (1)
 (1)  014312  104401  002165                    TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
 (1)  014316  000207                            RTS     PC              ;;RETURN
 (1)  014320                            7$:
 (2)  014320  013046                            MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
 (2)  014322  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 (1)  014324  005710                            TST     (R0)            ;;IS THERE ANOTHER NUMBER?
 (1)  014326  001770                            BEQ     6$              ;;BR IF NO
 (1)  014330  104401  014336                    TYPE    ,8$             ;;TYPE TWO(2) SPACES
 (1)  014334  000771                            BR      7$              ;;LOOP
 (1)  014336  020040      000           8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
 (1)          014342                            .EVEN
                                        .SBTTL  SCOPE HANDLER ROUTINESTARS
 (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)                                 ;*SW08=1        LOOP ON TEST IN SWR<7:0>
(1)                                 ;*CALL
(1)                                 ;*      SCOPE           ;;SCOPE=IOT
(1)
(1)  014342                         $SCOPE:
(1)  014342  104407                         CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
(1)  014344  032777  040000  165566  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(1)  014352  001114                         BNE     $OVER           ;;YES IF SW14=1
(1)                                 ;#####START OF CODE FOR THE XOR TESTER#####
(1)  014354  000416                 $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                                                 ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(1)  014356  013746  000004                 MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  014362  012737  014402  000004         MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  014370  005737  177060                 TST     @#177060        ;;TIME OUT ON XOR?
(1)  014374  012637  000004                 MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  014400  000463                         BR      $SVLAD          ;;GO TO THE NEXT TEST
(1)  014402  022626                 5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(1)  014404  012637  000004                 MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  014410  000423                         BR      7$              ;;LOOP ON THE PRESENT TEST
(1)  014412                         6$:;#####END OF CODE FOR THE XOR TESTER#####
(1)  014412  032777  000400  165520         BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
(1)  014420  001404                         BEQ     2$              ;;BR IF NO
(1)  014422  127737  165512  002102         CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
(1)  014430  001465                         BEQ     $OVER           ;;BR IF YES
(1)  014432  105737  002103         2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(1)  014436  001421                         BEQ     3$              ;;BR IF NO
(1)  014440  123737  002115  002103         CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
(1)  014446  101015                         BHI     3$              ;;BR IF NO
(1)  014450  032777  001000  165462         BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
(1)  014456  001404                         BEQ     4$              ;;BR IF NO
(1)  014460  013737  002110  002106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
(1)  014466  000446                         BR      $OVER
(1)  014470  105037  002103         4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  014474  005037  002160                 CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  014500  000415                         BR      1$              ;;ESCAPE TO THE NEXT TEST
(1)  014502  032777  004000  165430  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(1)  014510  001011                         BNE     1$              ;;BR IF YES
(1)  014512  005737  002176                 TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  014516  001406                         BEQ     1$              ;;      INHIBIT ITERATIONS
(1)  014520  005237  002104                 INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  014524  023737  002160  002104         CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  014532  002024                         BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(1)  014534  012737  000001  002104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  014542  013737  014620  002160         MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  014550  105237  002102         $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(1)  014554  113737  002102  002174         MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
(1)  014562  011637  002106                 MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  014566  011637  002110                 MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(1)  014572  005037  002162                 CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  014576  112737  000001  002115         MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
```

```
(1)  014604  013777  002102  165330  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1)  014612  013716  002106             MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  014616  000002                     RTI                     ;;FIXES PS
(1)  014620  000001                     $MXCNT: 1.              ;;MAX. NUMBER OF ITERATIONS
7124
(1)                                     .SBTTL  TTY INPUT ROUTINE
(2)                                     ;;**********************************************************************
(1)
(1)
(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)  014622  022737  000176  002140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
(1)  014630  001074                     BNE     15$             ;;BRANCH IF NO
(1)  014632  105777  165306             TSTB    @$TKS           ;;CHAR THERE?
(1)  014636  100071                     BPL     15$             ;;IF NO, DON'T WAIT AROUND
(1)  014640  117746  165302             MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
(1)  014644  042716  177600             BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
(1)  014650  022726  000007             CMP     #7,(SP)+        ;;IS IT A CONTROL G?
(1)  014654  001062                     BNE     15$             ;;NO, RETURN TO USER
(1)  014656  123727  002134  000001     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
(1)  014664  001456                     BEQ     15$             ;;BRANCH IF YES
(1)
(1)  014666  104401  015347             TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
(1)  014672  104401  015354  $GTSWR:  TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
(2)  014676  013746  000176             MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
(2)  014702  104402                     TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  014704  104401  015365             TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
(1)  014710  005046             19$:    CLR     -(SP)           ;;CLEAR COUNTER
(1)  014712  005046                     CLR     -(SP)
(1)                                     ;;THE NEW SWR
(1)  014714  105777  165224     7$:     TSTB    @$TKS           ;;CHAR THERE?
(1)  014720  100375                     BPL     7$              ;;IF NOT TRY AGAIN
(1)
(1)  014722  117746  165220             MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
(1)  014726  042716  177600             BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)
(1)
(1)
(1)  014732  021627  000025     9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
(1)  014736  001005                     BNE     10$             ;;BRANCH IF NOT
(1)  014740  104401  015342             TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
(1)  014744  062706  000006     20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
(1)  014750  000757                     BR      19$             ;;LET'S TRY IT AGAIN
(1)
(1)
(1)  014752  021627  000015     10$:    CMP     (SP),#15        ;;IS IT A <CR>?
(1)  014756  001022                     BNE     16$             ;;BRANCH IF NO
(1)  014760  005766  000004             TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
(1)  014764  001403                     BEQ     11$             ;;BRANCH IF YES
(1)  014766  016677  000002  165144     MOV     2(SP),@SWR      ;;SAVE NEW SWR
(1)  014774  062706  000006     11$:    ADD     #6,SP           ;;CLEAR UP STACK
```

```
(1)  015000  104401  002165          14$:   TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
(1)  015004  123727  002135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
(1)  015012  001003                          BNE     15$             ;;BRANCH IF NOT
(1)  015014  012777  000100  165122          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
(1)  015022  000002                  15$:    RTI                     ;;RETURN
(1)  015024  004737  013024          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
(1)  015030  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
(1)  015034  002420                          BLT     18$             ;;BRANCH IF YES
(1)  015036  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
(1)  015042  003015                          BGT     18$             ;;BRANCH IF YES
(1)  015044  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  015050  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  015054  001403                          BEQ     17$             ;;BRANCH IF YES
(1)  015056  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  015060  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
(1)  015062  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
(1)  015064  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  015070  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  015074  000707                          BR      7$              ;;GET THE NEXT ONE
(1)  015076  104401  002164          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  015102  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)  015104  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
(1)  015106  016666  000004  000002  MOV     4(SP),2(SP)     ;;SAVE THE PS
(1)  015114  105777  165024          1$:     TSTB    @$TKS           ;;WAIT FOR
(1)  015120  100375                          BPL     1$              ;;A CHARACTER
(1)  015122  117766  165020  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
(1)  015130  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
(1)  015136  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
(1)  015144  001013                          BNE     3$              ;;BRANCH IF NO
(1)  015146  105777  164772          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
(1)  015152  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
(1)  015154  117746  164766                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
(1)  015160  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)  015164  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
(1)  015170  001366                          BNE     2$              ;;IF NOT DISCARD IT
(1)  015172  000750                          BR      1$              ;;YES, RESUME
(1)  015174  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
(1)  015202  002407                          BLT     4$              ;;BRANCH IF YES
(1)  015204  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
(1)  015212  003003                          BGT     4$              ;;BRANCH IF YES
(1)  015214  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
(1)  015222  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)  015224  010346                     $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
(1)  015226  012703  015332             1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
(1)  015232  022703  015342             2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
(1)  015236  101405                             BLOS    4$              ;;BR IF YES
(1)  015240  104410                             RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
(1)  015242  112613                             MOVB    (SP)+,(R3)      ;;GET CHARACTER
(1)  015244  122713  000177             10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
(1)  015250  001003                             BNE     3$              ;;SKIP IF NOT
(1)  015252  104401  002164             4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  015256  000763                             BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  015260  111337  015330             3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  015264  104401  015330                     TYPE    ,9$
(1)  015270  122723  000015                     CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  015274  001356                             BNE     2$              ;;LOOP IF NOT RETURN
(1)  015276  105063  177777                     CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
(1)  015302  104401  002166                     TYPE    ,$LF            ;;TYPE A LINE FEED
(1)  015306  012603                             MOV     (SP)+,R3        ;;RESTORE R3
(1)  015310  011646                             MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  015312  016666  000004  000002             MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  015320  012766  015332  000004             MOV     #$TTYIN,4(SP)
(1)  015326  000002                             RTI                     ;;RETURN
(1)  015330     000             9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  015331     000                     .BYTE   0                       ;;TERMINATOR
(1)  015332  000010             $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
(1)  015342  052536  005015  000  $CNTLU: .ASCIZ  /^U/<15><12>           ;;CONTROL "U"
(1)  015347     136  006507  000012 $CNTLG: .ASCIZ  /^G/<15><12>        ;;CONTROL "G"
(1)  015354  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  015362  020075     000
(1)  015365     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
(1)  015372  036440  000040
7125                                    .SBTTL  POWER DOWN AND UP ROUTINES
(1)
(2)                                     ;;******************************************************************
(1)                                     ;POWER DOWN ROUTINE
(1)  015376  012737  015542  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
(1)  015404  012737  000300  000026        MOV     #PR6,@#PWRVEC+2 ;;PRIO:6
(3)  015412  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  015414  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  015416  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  015420  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  015422  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
(3)  015424  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(3)  015426  017746  164506                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
(1)  015432  010637  015546                MOV     SP,$SAVR6       ;;SAVE SP
(1)  015436  012737  015450  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
(1)  015444  000000                        HALT
(1)  015446  000776                        BR      .-2             ;;HANG UP
(1)
(2)                                     ;;******************************************************************
(1)                                     ;POWER UP ROUTINE
(1)  015450  012737  015542  000024 $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
(1)  015456  013706  015546                MOV     $SAVR6,SP       ;;GET SP
```

```
  (1)  015462  005037  015546            CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
  (1)  015466  005237  015546     1$:    INC     $SAVR6          ;;WAIT FOR THE INC
  (1)  015472  001375                    BNE     1$              ;;OF  WORD
  (3)  015474  012677  164440            MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
  (3)  015500  012605                    MOV     (SP)+,R5        ;;POP STACK INTO R5
  (3)  015502  012604                    MOV     (SP)+,R4        ;;POP STACK INTO R4
  (3)  015504  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
  (3)  015506  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
  (3)  015510  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
  (3)  015512  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
  (1)  015514  012737  015376  000024    MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
  (1)  015522  012737  000300  000026    MOV     #PR6,@#PWRVEC+2 ;;PRIO:6
  (1)  015530  104401                    TYPE                    ;;REPORT THE POWER FAILURE
  (1)  015532  015550            $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
  (1)  015534  012716                    MOV     (PC)+,(SP)      ;;RESTART AT START1
  (1)  015536  003066            $PWRAD: .WORD   START1          ;;RESTART ADDRESS
  (1)  015540  000002                    RTI
  (1)  015542  000000            $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
  (1)  015544  000776                    BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
  (1)  015546  000000            $SAVR6: 0                       ;;PUT THE SP HERE
 7126  015550  005015  042522  052123  PWRMSG: .ASCIZ  <15><12>/RESTARTED FROM PWR FAIL/
       015556  051101  042524  020104
       015564  051106  046517  050040
       015572  051127  043040  044501
       015600  000114
 7127                                   .EVEN
 7128                                   .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)  015602  010046            $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
  (1)  015604  016600  000002            MOV     2(SP),R0        ;;GET TRAP ADDRESS
  (1)  015610  005740                    TST     -(R0)           ;;BACKUP BY 2
  (1)  015612  111000                    MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
  (1)  015614  006300                    ASL     R0              ;;POSITION FOR INDEXING
  (1)  015616  016000  015636            MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
  (1)  015622  000200                    RTS     R0              ;;GO TO ROUTINE
  (1)
  (1)
  (1)                                   ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
  (1)
  (1)  015624  011646            $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
  (1)  015626  016666  000004  000002    MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
  (1)  015634  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)  015636  015624                    $TRPAD: .WORD   $TRAP2
   (3)  015640  012612                            $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
   (3)  015642  013366                            $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
   (3)  015644  013342                            $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
   (3)  015646  013402                            $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
   (3)  015650  013570                            $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
   (1)
   (3)  015652  014672                            $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
   (1)
   (3)  015654  014622                            $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
   (3)  015656  015104                            $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
   (3)  015660  015224                            $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
 7129
 7130                                       .SBTTL  ASCII MESSAGES
 7131                                       ;;GPA TITLED:   .ASCIZ  <15><12>/CNDRDA  DRV11J DIAG TEST PART 2/<15><12>
 7132  015662  041600  042116  042122      TITLED: .ASCIZ  <200>/CNDRDA  DRV11J DIAG TEST PART 2/<200>     ;;GPA
       015670  020101  042040  053122
       015676  030461  020112  044504
       015704  043501  052040  051505
       015712  020124  040520  052122
       015720  031040  000200
 7133                                       ;;GPA TLCABL:   .ASCIZ  <15><12>/DRV11J CABLE REQ'D/<15><12>
 7134  015724  042200  053122  030461      TLCABL: .ASCIZ  <200>/DRV11J CABLE REQ'D/<200>                  ;;GPA
       015732  020112  040503  046102
       015740  020105  042522  023521
       015746  100104     000
 7135
 7136  015751     122  043505  052040      EM1:    .ASCIZ  /REG TIMEOUT ER/
       015756  046511  047505  052125
       015764  042440  000122
 7137  015770  042522  020107  042522      EM2:    .ASCIZ  'REG READ/WRITE ER'
       015776  042101  053457  044522
       016004  042524  042440  000122
 7138  016012  051111  020122  042522      EM3:    .ASCIZ  /IRR REG ER/
       016020  020107  051105     000
 7139  016025     101  051103  051040      EM4:    .ASCIZ  /ACR REG ER/
       016032  043505  042440  000122
 7140  016040  046511  020122  042522      EM5:    .ASCIZ  /IMR REG ER/
       016046  020107  051105     000
 7141  016053     111  051123  051040      EM6:    .ASCIZ  /ISR REG ER/
       016060  043505  042440  000122
 7142  016066  046111  042514  040507      EM7:    .ASCIZ  /ILLEGAL VECTOR MEM ADDR ER/
       016074  020114  042526  052103
       016102  051117  046440  046505
       016110  040440  042104  020122
       016116  051105     000
 7143  016121     103  044510  020120      EM10:   .ASCIZ  /CHIP STAT ER/
       016126  052123  052101  042440
       016134  000122
 7144                                       ;;GPA EM11:     .ASCIZ  /ILLEGAL INTERRUPT RECEIVED/
 7145  016136  046111  042514  040507      EM11:   .ASCIZ  /ILLEGAL INTERRUPT REC'D/                       ;;GPA
       016144  020114  047111  042524
       016152  051122  050125  020124
       016160  042522  023503  000104
 7146  016166  047111  042524  051122      EM12:   .ASCIZ  /INTERRUPT TEST ERROR/
       016174  050125  020124  042524
```

```
                016202  052123  042440  051122
                016210  051117     000
7147                                            ;;GPA EM13:     .ASCIZ  /MULTIPLE INTERRUPTS RECEIVED/
7148    016213     115  046125  044524  EM13:   .ASCIZ  /MULTIPLE INTERRUPTS REC'D/                ;;GPA
        016220  046120  020105  047111
        016226  042524  051122  050125
        016234  051524  051040  041505
        016242  042047     000
7149    016245     126  041505  020124  EM14:   .ASCIZ  /VECT ADDR MEM ER/
        016252  042101  051104  046440
        016260  046505  042440  000122
7150    016266  051105  050122  020103  DH1:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  EXPCT   RCVD/
        016274  020040  051524  047124
        016302  046525  020040  052502
        016310  040523  051104  020040
        016316  054105  041520  020124
        016324  020040  041522  042126
        016332     000
7151    016333     105  051122  041520  DH2:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  VAM     ADDR    EXPCT   RCVD/
        016340  020040  052040  052123
        016346  052516  020115  041040
        016354  051525  042101  020122
        016362  053040  046501  020040
        016370  020040  040440  042104
        016376  020122  020040  042440
        016404  050130  052103  020040
        016412  051040  053103  000104
7152    016420  051105  050122  020103  DH3:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  VAM     ADDR/
        016426  020040  051524  047124
        016434  046525  020040  052502
        016442  040523  051104  020040
        016450  040526  020115  020040
        016456  020040  042101  051104
        016464     000
7153    016465     105  051122  041520  DH4:    .ASCIZ  /ERRPC  TSTNUM  TESTPC  BUSADR  VAM     ADDR/
        016472  052011  052123  052516
        016500  004515  042524  052123
        016506  041520  041011  051525
        016514  042101  020122  053040
        016522  046501  020040  020040
        016530  040440  042104  000122
7154
7155                                            .EVEN
7156    016536  002116  002432  002122  DT1:    $ERRPC,TSTNUM,$BDADR,$GDDAT,$BDDAT,0
        016544  002124  002126  000000
7157    016552  002116  002432  002122  DT2:    $ERRPC,TSTNUM,$BDADR,VECVAL,VECLOC,$GDDAT,$BDDAT,0
        016560  002456  002452  002124
        016566  002126  000000
7158    016572  002116  002432  002122  DT3:    $ERRPC,TSTNUM,$BDADR,VECVAL,VECLOC,0
        016600  002456  002452  000000
7159    016606  002116  002432  002120  DT4:    $ERRPC,TSTNUM,$GDADR,$BDADR,VECVAL,VECLOC,0
        016614  002122  002456  002452
        016622  000000
7160
7161
7162                                            ;;**********************************************************************
```

```
 7163                                   ;'DBUF' IS THE STORAGE AREA FOR SYSMAC LOCATIONS 0-202
 7164                                   ;PRESERVED IN ORDER TO TEST DRV11J VECTORING TO
 7165                                   ;VECTOR SPACE 0-200.
 7166                                   ;;*****************************************************************
 7167  016624  000102                   DBUF:   .BLKW   66.             ;1ST ADRS OF DATA STORAGE AREA
 7168
 7169                                   ;;*****************************************************************
 7170                                   ;VECTOR BUFFER USED FOR VECTOR UNIQUENESS TEST
 7171                                   ;FOR 64 INTERRUPT VECTORS.
 7172                                   ;;*****************************************************************
 7173  017030  000100                   VBUF:   .BLKW   64.
 7174  017230                           ENDBUF:
```

```
 7263                                   ;THE FOLLOWING CALL WAS INCLUDED TO INIT BRKVEC AND LKVEC
 7264            017230                 POINT=.                 ;SAVE POINTER
  (1)            000100                 .=100
  (1)   000100   017230                 $CLKVEC                 ;LKVEC  HANDLER
  (1)   000102   000300                 300                     ;INTERRUPT HANDLER PRI
  (1)            000140                 .=140                   ;BRKVEC
  (1)   000140   170000                 170000                  ;ODT START ADDRESS
  (1)   000142   000300                 300                     ;PRIORITY
  (1)            017230                 .=POINT                 ;RESTORE POINTER
  (1)   017230   104401   017236        $CLKVEC:        TYPE,CLKMES
  (1)   017234   000000                                 HALT
  (1)   017236   005015   045514   042526  CLKMES:  .ASCIZ <15><12>/LKVEC INTERRUPT - DISCONNECT LTC /
  (1)   017244   020103   047111   042524
  (1)   017252   051122   050125   020124
  (1)   017260   020055   044504   041523
  (1)   017266   047117   042516   052103
  (1)   017274   046040   041524   000040
 7265                                   ;THE FOLLOWING FLAG WAS INCLUDED TO INDICATE FALCON
 7266   017302   000001                 KXTFLAG:1
 7267            000001                         .END
```

```
ABASE = 174160          5694#   5743    5820    5821    5822    5823    5824    5825    5826    5827
ACDW1 = 000000          5743
ACDW2 = 000000          5743
ACPUOP= 000000          5743
ACRLOC  002450          5839#
ADDW0 = 000000          5743
ADDW1 = 000000          5743
ADDW10= 000000          5743
ADDW11= 000000          5743
ADDW12= 000000          5743
ADDW13= 000000          5743
ADDW14= 000000          5743
ADDW15= 000000          5743
ADDW2 = 000000          5743
ADDW3 = 000000          5743
ADDW4 = 000000          5743
ADDW5 = 000000          5743
ADDW6 = 000000          5743
ADDW7 = 000000          5743
ADDW8 = 000000          5743
ADDW9 = 000000          5743
ADEVCT= 000000          5743
ADEVM = 000001          5695#   5743
AENV  = 000000          5743
AENVM = 000000          5743
AFATAL= 000000          5743
AMADR1= 000000          5743
AMADR2= 000000          5743
AMADR3= 000000          5743
AMADR4= 000000          5743
AMAMS1= 000000          5743
AMAMS2= 000000          5743
AMAMS3= 000000          5743
AMAMS4= 000000          5743
AMSGAD= 000000          5743
AMSGLG= 000000          5743
AMSGTY= 000000          5743
AMTYP1= 000000          5743
AMTYP2= 000000          5743
AMTYP3= 000000          5743
AMTYP4= 000000          5743
APASS = 000000          5743
APRIOR= 000000          5743
APTCSU= 000040          7109    7110#
APTENV= 000001          7109    7110#   7113
APTSIZ= 000200          5853    7110#
APTSPO= 000100          7109    7110#
APTVEC  011546          6827    6845#
APT200  012020          6905    6912#
ASWREG= 000000          5743
ATESTN= 000000          5743
AUNIT = 000000          5743
AUSWR = 000000          5743
AVECT1= 000000          5743
AVECT2= 000000          5743
BDSAV   002464          5845#
```

```
BGCHP3  012570          5924    6035    6124    6249    7095#
BIT0   = 000001         5693#   6717
BIT00  = 000001         5693#
BIT01  = 000002         5693#
BIT02  = 000004         5693#
BIT03  = 000010         5693#
BIT04  = 000020         5693#
BIT05  = 000040         5693#
BIT06  = 000100         5693#
BIT07  = 000200         5693#
BIT08  = 000400         5693#   7123
BIT09  = 001000         5693#   7113    7123
BIT1   = 000002         5693#   5953    6054    6167    6296    6423    6556
BIT10  = 002000         5693#
BIT11  = 004000         5693#   7123
BIT12  = 010000         5693#
BIT13  = 020000         5693#   7113
BIT14  = 040000         5693#   7123
BIT15  = 100000         5693#   5696
BIT2   = 000004         5693#
BIT3   = 000010         5693#
BIT4   = 000020         5693#
BIT5   = 000040         5693#
BIT6   = 000100         5693#
BIT7   = 000200         5693#
BIT8   = 000400         5693#   5697
BIT9   = 001000         5693#   5698    6009    6042    6134    6257
BPTVEC= 000014          5693#
BRKVEC= 000140          5693#
BYCNT   002470          5847#   6146*   6149*   6270*   6274*   6734*   6745*
BYNUM   002472          5848#   6121*   6146    6168    6227*   6246*   6270    6297    6363*   6866    6927
CAT200  011756          6263    6285    6293    6305    6317    6325    6333    6341    6350    6902#   6934    6941
CHPISR= 000140          5714#   5995    6563    6577
CIMR   = 000040         5704#   6401    6413    6535    6546
CIRMR  = 000020         5701#
CIRR   = 000100         5709#   6567    6581
CISR   = 000160         5715#   6343
CKSWR  = 104407         6228    6364    7113    7120    7123    7128#
CLKMES  017236          7264#
CLRBF   011266          6384    6518    6771#
CLRCSR  010746          5923    6034    6122    6247    6383    6517    6698#
CLRIRR  011026          5926    6037    6135    6258    6389    6523    6714#
CR     = 000015         5693#   7109
CRLF   = 000200         5693#   7109
CSIMR  = 000050         5705#   5925    6036    6126    6251
CSIRMR= 000030          5702#
CSIRR  = 000110         5710#
CSISR  = 000170         5716#   6128
DBUF    016624          6781    6792    7167#
DDISP  = 177570         5693#   5743    5853
DH1     016266          5746    5782    7150#
DH2     016333          5752    5758    5764    5770    5776    5788    5800    5806    7151#
DH3     016420          5794    7152#
DH4     016465          5812    7153#
DIR    = 000400         5697#
DISPLA  002142          5743#   5853*   7113*   7123*
```

```
DISPRE  000174          5732#  5853
DMAP    002434          5833#  5895*  5896*  5899   6682*  6686
DRCSA   002412          5820#  5898*  5902   5918   5920   6029   6031   6117   6119   6242   6244   6394   6528
                        6680   6702   6715   6727
DRCSB   002416          5822#  5919   6030   6118   6243   6396   6530   6728
DRCSC   002422          5824#  6022   6109   6235   6371   6395   6529   6753
DRCSD   002426          5826#  6023   6110   6236   6373   6397   6531   6754
DRDBA   002414          5821#
DRDBB   002420          5823#
DRDBC   002424          5825#
DRDBD   002430          5827#  5905
DSWR  = 177570          5693#  5743   5853
DT1     016536          5747   5783   7156#
DT2     016552          5753   5759   5765   5771   5777   5789   5801   5807   7157#
DT3     016572          5795   7158#
DT4     016606          5813   7159#
EDCHP3  012610          6016   6103   7103#
EMTVEC= 000030          5693#  5853*
EM1     015751          5745   7136#
EM10    016121          5787   7143#
EM11    016136          5793   7145#
EM12    016166          5799   7146#
EM13    016213          5805   7148#
EM14    016245          5781   7149#
EM2     015770          5751   7137#
EM3     016012          5757   7138#
EM4     016025          5763   7139#
EM5     016040          5769   7140#
EM6     016053          5775   7141#
EM7     016066          5811   7142#
ENDBUF  017230          7174#
ENDTRP  011570          6837   6844   6850#
END200  012040          6911   6918#
ERRVEC= 000004          5693#  5853*  7123*
FRMBUF  011326          6283   6291   6303   6315   6323   6331   6339   6348   6352   6377   6789#  6876   6931
                        6939
GNS   = ****** U        5732   7128
GRPCNT  002462          5844#  5922*  6020*  6033*  6107*  6120*  6233*  6245*  6369*
GTSWR = 104406          5893   7128#
HT    = 000011          5693#  7109
IE    = 001000          5698#
IMRLOC  002442          5836#  5925*  5935   6018*  6036*  6051   6105*  6126*  6153   6220*  6251*  6278   6357*
INTBY4  011632          6143   6865#
INTFLG  002436          5834#  5929*  5942   5945*  5948   5951*  5955   6040*  6055   6058*  6061   6064*  6071
                        6145*  6156   6159*  6162   6165*  6170   6269*  6281   6286*  6289   6294*  6299   6402*
                        6407   6410*  6418   6421*  6425   6536*  6541   6544*  6551   6554*  6558   6569*  6572
                        6855*  6857   6865*  6867   6926*  6928   6948*  6949
INTSR1  011576          5927   6038   6855#
INTSR3  012056          6267   6919   6926#
INT0    012170          6760   6959#
INT1    012174          6961#
INT10   012240          6979#
INT11   012244          6981#
INT12   012250          6983#
INT13   012254          6985#
INT14   012260          6987#
```

```
INT15   012264          6989#
INT16   012270          6991#
INT17   012274          6993#
INT18   012300          6995#
INT19   012304          6997#
INT2    012200          6963#
INT20   012310          6999#
INT21   012314          7001#
INT22   012320          7003#
INT23   012324          7005#
INT24   012330          7007#
INT25   012334          7009#
INT26   012340          7011#
INT27   012344          7013#
INT28   012350          7015#
INT29   012354          7017#
INT3    012204          6965#
INT30   012360          7019#
INT31   012364          7021#
INT32   012370          7023#
INT33   012374          7025#
INT34   012400          7027#
INT35   012404          7029#
INT36   012410          7031#
INT37   012414          7033#
INT38   012420          7035#
INT39   012424          7037#
INT4    012210          6967#
INT40   012430          7039#
INT41   012434          7041#
INT42   012440          7043#
INT43   012444          7045#
INT44   012450          7047#
INT45   012454          7049#
INT46   012460          7051#
INT47   012464          7053#
INT48   012470          7055#
INT49   012474          7057#
INT5    012214          6969#
INT50   012500          7059#
INT51   012504          7061#
INT52   012510          7063#
INT53   012514          7065#
INT54   012520          7067#
INT55   012524          7069#
INT56   012530          7071#
INT57   012534          7073#
INT58   012540          7075#
INT59   012544          7077#
INT6    012220          6971#
INT60   012550          7079#
INT61   012554          7081#
INT62   012560          7083#
INT63   012564          7085#
INT7    012224          6973#
INT8    012230          6975#
```

```
INT9    012234          6977#
IOTVEC= 000020          5693#   5853*
IRRLOC  002446          5838#   6127*   6160    6221*   6252*   6287    6358*
ISRLOC  002444          5837#   6128*   6204    6222*
KXTFLA  017302          6211    6502    6666    6737    7266#
LF    = 000012          5693#   7109
LKVEC = 000100          5693#
LMD04 = 000200          5718#
LMD57 = 000240          5719#
LVLCNT  002466          5846#   6125*   6224*   6250*   6360*   6732*   6747*
LVLSAV  002474          5849#   6533*   6565*   6568*   6579*
MACR  = 000254          5725#   5989    6097    6198    6335    6463    6487    6614    6636
MIMR  = 000244          5723#   5977    6085    6186    6319    6452    6476    6604    6625
MIRR  = 000250          5724#   5983    6091    6192    6327    6457    6482    6609    6631
MISR  = 000240          5722#   5971    5996    6079    6180    6205    6311    6344    6447    6471    6599    6620
NEXPAS  003132          5900    5902#   6688
NEXPA1  003136          5903#   5906
NEXPA2  003202          5911    5913#
NXDEV   010564          6680#
NXDEV1  010570          5901    6681#   6687
ODTST = 170000          5693#
PACR  = 000300          5728#   6045    6399    6411
PIMR  = 000260          5729#
PIRQ  = 177772          5693#
PIRQVE= 000240          5693#
POINT = 017230          7264#
PR0   = 000000          5693#   5937*   5952*   6053*   6065*   6154*   6166*   6279*   6295*   6403*   6414*   6422*   6537*
                        6547*   6555*   6570*
PR1   = 000040          5693#
PR2   = 000100          5693#
PR3   = 000140          5693#
PR4   = 000200          5693#
PR5   = 000240          5693#
PR6   = 000300          5693#   5853    5913*   5947*   5959*   6060*   6075*   6161*   6175*   6288*   6306*   6417*   6426*
                        6550*   6559*   6573*   6699*   7125
PR7   = 000340          5693#
PS    = 177776          5693#
PSW   = 177776          5693#
PVMA  = 000340          5730#   5931    6047    6123    6248    6700
PWRMSG  015550          7125    7126#
PWRVEC= 000024          5693#   5853*   7125*
RDCHR = 104410          7124    7128#
RDLIN = 104411          7128#
RDY   = 100000          5696#
RESTRP  011364          6210    6802#
RESVEC= 000010          5693#
RES200  011714          6351    6888#
SETVEC  012142          6393    6527    6946#
SIMR  = 000060          5706#   5934    6050    6152    6277
SIRR  = 000120          5711#   5946    6059    6404    6415    6538    6548    6564    6578
SSIMR = 000070          5707#
SSIRR = 000130          5712#   6127    6252
STACK = 001100          5693#
START   002476          5732    5853#
START1  003066          5892    5894#   6690    7125
STKLMT= 177774          5693#
```

```
STRVEC  011232          6386    6520    6759#
SWR     002140          5743#   5853*   6796*   6804    6829    6903    7113    7123    7124*   7125*
SWRCK   014170          7113    7118#
SWREG   000176          5732#   5853    7124
SWRSAV  002460          5843#   6796    6903*
SW0   = 000001          5693#
SW00  = 000001          5693#
SW01  = 000002          5693#
SW02  = 000004          5693#
SW03  = 000010          5693#
SW04  = 000020          5693#
SW05  = 000040          5693#
SW06  = 000100          5693#
SW07  = 000200          5693#
SW08  = 000400          5693#
SW09  = 001000          5693#
SW1   = 000002          5693#
SW10  = 002000          5693#
SW11  = 004000          5693#
SW12  = 010000          5693#   6804    6829
SW13  = 020000          5693#
SW14  = 040000          5693#
SW15  = 100000          5693#
SW2   = 000004          5693#
SW3   = 000010          5693#
SW4   = 000020          5693#
SW5   = 000040          5693#
SW6   = 000100          5693#
SW7   = 000200          5693#
SW8   = 000400          5693#
SW9   = 001000          5693#
TBITVE= 000014          5693#
TITLED  015662          5887    7132#
TKVEC = 000060          5693#
TLCABL  015724          5888    7134#
TOBUF   011306          5882    6780#
TPVEC = 000064          5693#
TRAPVE= 000034          5693#   5853*
TRPALL  011704          6811    6833    6883#
TRPCAT  011456          5907    6129    6385    6519    6824#
TRPOUT  011666          6813    6846    6875#   6893    6913
TRP200  012122          6890    6907    6938#
TRTVEC= 000014          5693#
TSTNUM  002432          5832#   7118*   7156    7157    7158    7159
TST1    003206          5917#
TST2    004054          6021    6028#
TST3    004554          6108    6116#
TST4    005470          6234    6241#
TST5    006462          6380#
TST6    007430          6514#
TYPDS = 104405          6690    7128#
TYPE  = 104401          5887    5888    6690    7109    7111    7112    7113    7122    7124    7125    7128#   7264
TYPOC = 104402          7122    7124    7128#
TYPON = 104404          7128#
TYPOS = 104403          7128#
VBUF    017030          6390    6494    6524    6658    6771    7173#
```

```
VECFIL  011064            6391    6525    6727#
VECLOC  002452            5840#   6131*   6139    6213    6216    6254*   6262    6353    6701*   6806*   6807    6809*   6816*
                          6896*   7157    7158    7159
VECPAT  002454            5841#   6140*   6141*   6142*   6148    6264*   6265*   6266*   6273    6730*   6735    6736*   6739
                          6741*
VECVAL  002456            5842#   6123*   6147    6223*   6229    6248*   6272    6359*   6365    6388*   6522*   6700*   6731*
                          6733    6749*   7157    7158    7159
XXDP    002440            5835#   5873*   5885
$APTHD  002000            5742#
$ASTAT= ****** U          7110
$ATYC   013120            7110#
$ATY1   013074            7110#
$ATY3   013102            7109    7110#
$ATY4   013112            7110#   7113
$AUTOB  002134            5743#   5877*   5883    5891    7124
$BASE   002244            5743#   5897
$BDADR  002122            5743#   5933*   5961*   5967*   6002*   6049*   6076*   6151*   6176*   6276*   6307*   6398*   6431*
                          6438*   6444*   6470*   6496*   6532*   6583*   6590*   6596*   6619*   6644*   6651*   6660*   7156
                          7157    7158    7159
$BDDAT  002126            5743#   5955*   5956    5962*   5963*   5964    5968*   5972*   5973    5978*   5979    5984*   5985
                          5990*   5991    5997*   5998    6003*   6004*   6005    6010*   6011*   6012    6071*   6072    6077*
                          6080*   6081    6086*   6087    6092*   6093    6098*   6099    6170*   6172    6177*   6181*   6182
                          6187*   6188    6193*   6194    6199*   6200    6206*   6207    6299*   6301    6308*   6312*   6313
                          6320*   6321    6328*   6329    6336*   6337    6345*   6346    6425*   6427    6432*   6433*   6434
                          6439*   6440*   6441    6445*   6448*   6449    6453*   6454    6458*   6459    6464*   6465    6469*
                          6472*   6473    6477*   6478    6483*   6484    6489*   6490    6497*   6498    6558*   6560    6572*
                          6574    6584*   6585*   6586    6591*   6592*   6593    6597*   6600*   6601    6605*   6606    6610*
                          6611    6615*   6616    6621*   6622    6626*   6627    6632*   6633    6637*   6638    6645*   6646*
                          6647    6652*   6653*   6654    6661*   6662    6704*   6857*   6858    6867*   6868    6928*   6929
                          7156    7157
$CDW1   002250            5743#
$CHARC  013070            7109#*
$CKSWR  014622            7124#   7128
$CLKVE  017230            7264#
$CMTAG  002100            5743#   5853
$CM3  =  000000           5743#
$CNTLG  015347            7124#
$CNTLU  015342            7124#
$CPUOP  002216            5743#
$CRLF   002165            5743#   7109    7113    7122    7124
$DBLK   014004            7112#
$DEVCT  002200            5743#   5909*
$DEVM   002246            5743#   5895
$DOAGN  010722            6690#
$DTBL   013774            7112#
$ENDAD  010712            5737    6690#
$ENDCT  010660            6690#
$ENDMG  010731            6690#
$ENULL  010726            6690#
$ENV    002210            5743#   5875    5910    6802    6826    6888    6904    7109    7110    7113
$ENVM   002211            5743#   5853    7109    7110
$EOP    010624            6683    6690#
$EOPCT  010652            6690#
$ERFLG  002103            5743#   7113*   7123*
$ERMAX  002115            5743#   5853*   7123*
$ERROR  014014            5853    7113#
```

```
$ERRPC  002116          5743#  7113*  7122   7156   7157   7158   7159
$ERRTB  002252          5743#  7122
$ERRTY  014206          7119   7122#
$ERTTL  002112          5743#  7113*
$ESCAP  002162          5743#  5853*  7113   7123*
$ETABL  002210          5743#
$ETEND  002252          5742   5743#
$FATAL  002172          5743#  5854*  7110*
$FFLG   013340          7110#*
$FILLC  002156          5743#  7109
$FILLS  002155          5743#  7109
$GDADR  002120          5743#  6875*  6883*  6938*  7159
$GDDAT  002124          5743#  5954*  5956   5960*  5964   5969*  5970*  5973   5976*  5979   5982*  5985   5988*
                        5991   5994*  5998   6001*  6005   6008*  6012   6070*  6072   6078*  6081   6084*  6087
                        6090*  6093   6096*  6099   6168*  6172   6178*  6179*  6182   6185*  6188   6191*  6194
                        6197*  6200   6203*  6207   6297*  6301   6309*  6310*  6313   6318*  6321   6326*  6329
                        6334*  6337   6342*  6346   6424*  6427   6430*  6434   6437*  6441   6446*  6449   6454
                        6459   6462*  6465   6468*  6473   6478   6484   6488*  6490   6495*  6498   6501*  6504
                        6506*  6557*  6560   6571*  6574   6582*  6586   6589*  6593   6598*  6601   6606   6611
                        6616   6622   6627   6633   6638   6643*  6647   6650*  6654   6659*  6662   6665*  6668
                        6670*  6703*  6856*  6858   6866*  6868   6927*  6929   7156   7157
$GET42  010702          6690#
$GTSWR  014672          7124#  7128
$HD   = 000003          5686
$HIBTS  002000          5742#
$ICNT   002104          5743#  7123*
$ILLUP  015542          7125#
$INTAG  002135          5743#  7124
$ITEMB  002114          5743#  7113*  7122
$LF     002166          5743#  7109   7113   7124
$LFLG   013337          7110#*
$LPADR  002106          5743#  5853*  7123*
$LPERR  002110          5743#  5853*  5921*  6032*  6130*  6253*  7113   7123*
$MADR1  002222          5743#
$MADR2  002226          5743#
$MADR3  002232          5743#
$MADR4  002236          5743#
$MAIL   002170          5742   5743#  5853   7109   7113   7123
$MAMS1  002220          5743#
$MAMS2  002224          5743#
$MAMS3  002230          5743#
$MAMS4  002234          5743#
$MBADR  002002          5742#
$MFLG   013336          7110#*
$MNEW   015365          7124#
$MSGAD  002204          5743#  7110*
$MSGLG  002206          5743#  7110*
$MSGTY  002170          5743#  5855*  7110*
$MSWR   015354          7124#
$MTYP1  002221          5743#
$MTYP2  002225          5743#
$MTYP3  002231          5743#
$MTYP4  002235          5743#
$MXCNT  014620          7123#
$NULL   002154          5743#  7109
$NWTST= 000001          5917#  6028#  6116#  6241#  6380#  6514#
```

```
$OCNT   013564          7111#*
$OMODE  013566          7111#*
$OVER   014604          7123#
$PASS   002176          5743#   5853*   6690*   7123
$PASTM  002006          5742#
$PWRAD  015536          7125#
$PWRDN  015376          5853    7125#
$PWRMG  015532          7125#
$PWRUP  015450          7125#
$QUES   002164          5743#   7109    7113    7124
$RDCHR  015104          7124#   7128
$RDDEC= ****** U        7128
$RDLIN  015224          7124#   7128
$RDOCT= ****** U        7128
$RDSZ = 000010          7124#
$RTNAD  010724          6690#
$R2A  = ****** U        7128
$SAVRE= ****** U        7128
$SAVR6  015546          7125#*
$SCOPE  014342          5853    7123#
$SETUP= 000117          5851#   5853    6690    7113    7123    7124
$STUP = 177777          5851#
$SVLAD  014550          7123#
$SVPC = 000106          5737#
$SWR  = 165400          5686#   5690#   5692    5743    5853    5917    6028    6116    6241    6380    6514    6690    7113
                        7123    7125
$SWREG  002212          5743#   5853
$SWRMK= 000000          5692    7123
$TESTN  002174          5743#   5856*   7123*
$TIMES  002160          5743#   5853*   6690*   7123*
$TKB    002146          5743#   7124
$TKS    002144          5743#   7124*
$TN   = 000007          5686#   5691#   5917#   6021    6028#   6108    6116#   6234    6241#   6380#   6514#
$TPB    002152          5743#   7109*
$TPFLG  002157          5743#   7109
$TPS    002150          5743#   7109
$TRAP   015602          5853    7128#
$TRAP2  015624          7128#
$TRP  = 000012          7128#
$TRPAD  015636          7128#
$TSTM   002004          5742#
$TSTNM  002102          5743#   6690*   7113    7118    7123*
$TTYIN  015332          7124#
$TYPBN= ****** U        7128
$TYPDS  013570          7112#   7128
$TYPE   012612          7109#   7110    7128
$TYPEC  013024          7109#   7124
$TYPEX  013072          7109#
$TYPOC  013366          7111#   7128
$TYPON  013402          7111#   7128
$TYPOS  013342          7111#   7128
$UNIT   002202          5743#   5894*   5909    6685*
$UNITM  002010          5742#
$USWR   002214          5743#
$VECT1  002240          5743#
$VECT2  002242          5743#
```

```
$XTSTR  014354          7123#
$$GET4= 000000          6690#
$OFILL  013565          7111#*
$40CAT= ****** U        7113    7123
.     = 017304          5732#   5733#   5737#   5738#   5742#   5743#   5853    6690    7109    7110#   7112#   7113    7122#
                        7123    7124#   7125    7167#   7173#   7264#
.$ASTA= ****** U        7110
.$X   = 002000          5742#
```

```
COMMEN  1566#  5693#
ENDCOM  1578#  5693#
ERROR   5693#  5944  5950  5958  5966  5975  5981  5987  5993  6000  6007  6014  6057  6063  6074
        6083   6089  6095  6101  6158  6164  6174  6184  6190  6196  6202  6209  6284  6292  6304
        6316   6324  6332  6340  6349  6409  6420  6429  6436  6443  6451  6456  6461  6467  6475
        6480   6486  6492  6500  6543  6553  6562  6576  6588  6595  6603  6608  6613  6618  6624
        6629   6635  6640  6649  6656  6664  6860  6870  6877  6884  6932  6940  6951
ESCAPE  1698#  5693#
GETPRI  1312#  5693#
GETSWR  1771#  5693#
MULT    4484#  5693#
NEWTST  1626#  5693#  5917  6028  6116  6241  6380  6514
POP     2153#  5693#  7110  7112  7125
PUSH    2145#  5693#  7110  7112  7125
REPORT  5463#  5693#
SCOPE   5693#  5917  6028  6116  6241  6380  6514
SETPRI  1279#  5693#
SETTRA  7128#
SETUP   1337#  5693#  5853
SKIP    1733#  5693#  6021  6108  6234
SLASH   1517#  5693#
SPACE   5693#
STARS   1485#  5693#  5737  5742  5743  5917  6028  6116  6241  6380  6514  6690  6695  6697  7109
        7110   7111  7112  7113  7114  7117  7122  7124  7125  7128  7162  7166  7169  7172
SWRSU   1453#  5693#  5853#
TRMTRP  7128#
TYPBIN  2088#  5693#
TYPDEC  2058#  5693#  6690
TYPNAM  1826#  5693#
TYPNUM  2025#  5693#
TYPOCS  1978#  5681#  5693#
TYPOCT  1941#  5693#  7122  7124
TYPTXT  1894#  5693#
$$CMRE  5743#
$$CMTM  5743#
$$ESCA  1711#  5693#
$$NEWT  1662#  5693#  5917  6028  6116  6241  6380  6514
$$SET   7128#
$$SETM  5853#
$$SKIP  1746#  5693#  6021  6108  6234
.EQUAT   189#  5681#  5693
.HEADE    61#  5680#  5686
.INIT   5658#  5693#  7264
.SETUP  1213#  5680#  5851
.SWRHI   104#  5681#  5692
.SWRLO  5692#
.$ACT1  5064#  5683#  5737
.$APTB  5109#  5683#  5743#
.$APTH  5370#  5683#  5742
.$APTY  5547#  5683#  7110
.$ASTA  5417#
.$CATC   932#  5681#  5732
.$CMTA  1047#  5681#  5743
.$DB2D  4686#
.$DB2O  4812#
.$DIV   4587#
```

```
.$EOP    2214#   5682#   6690
.$ERRO   2700#   5682#   7113
.$ERRT   2896#   5682#   7122
.$MULT   4523#
.$POWE   4229#   5681#   7125
.$RAND   4307#
.$RDDE   3891#
.$RDOC   3797#   5680#
.$READ   3395#   5682#   7124
.$R2AZ   4958#
.$SAVE   3969#
.$SB2D   4771#
.$SB2O   4874#
.$SCOP   2454#   5682#   7123
.$SIZE   4361#
.$SUPR   4913#
.$TRAP   4073#   5680#   7128
.$TYPB   3287#
.$TYPD   3209#   5682#   7112
.$TYPE   2985#   5682#   7109
.$TYPO   3112#   5681#   7111
.$40CA    972#

. ABS.  017304     000

ERRORS DETECTED:  0

CNDRDA,CNDRDA/CR/NL:TOC=CNMAC2.SML,CNDRDA.P11
RUN-TIME: 11 12 1 SECONDS
RUN-TIME RATIO: 56/25=2.2
CORE USED:  33K  (66 PAGES)
```