VSV01

VIDEO GRAPHIC TEST

MD-11-DZVSE-A

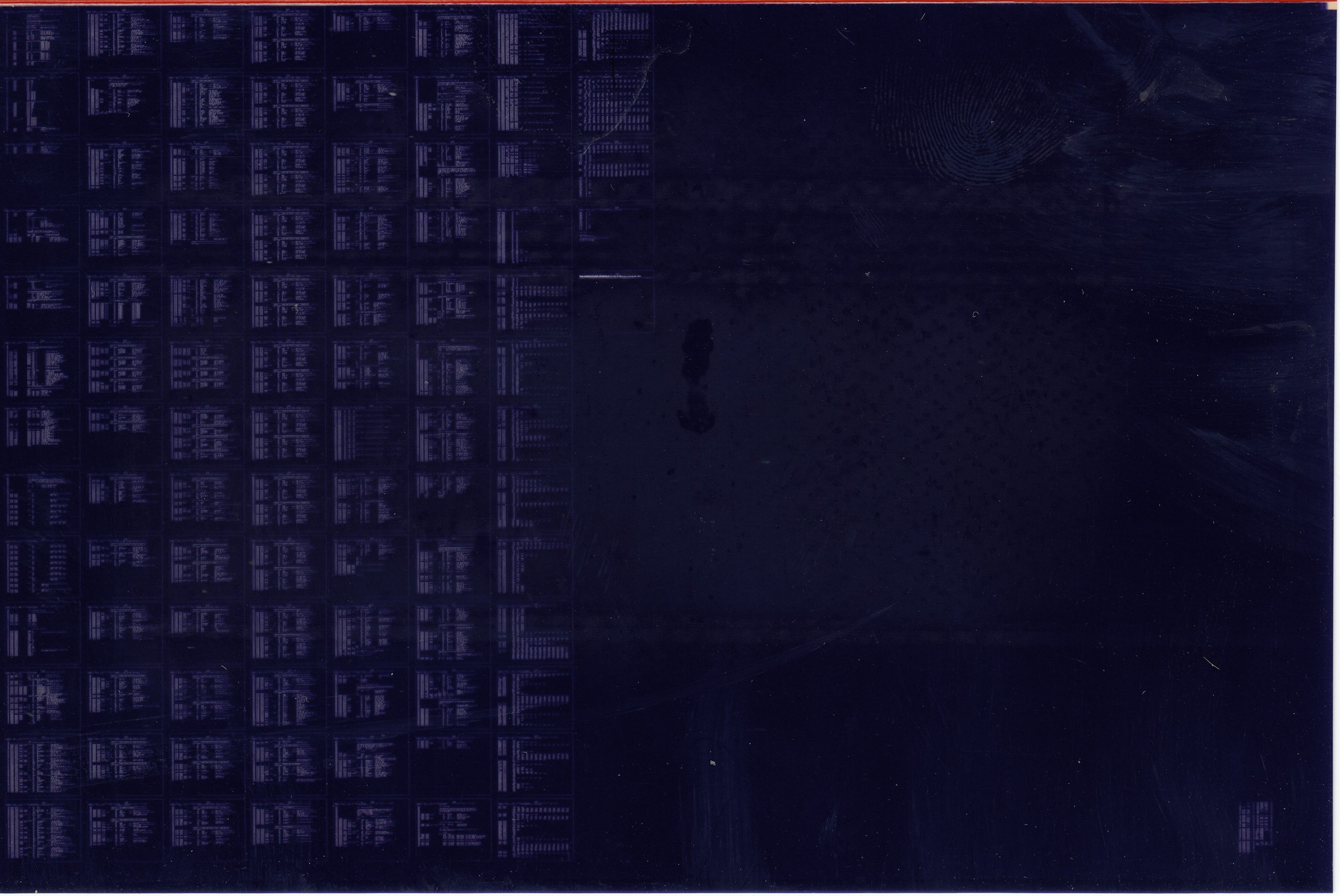
EP-DZVSE-A-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA



```
                        .TITLE  MAINDEC-11-DZVSE-A      VSVC1 VIEDO GRAPHIC TEST PROGRAM
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY RAYMOND SHOOP
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                        ;*
                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
000000 = 001100         STACK=  1100
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

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

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

                        ;*PRIORITY LEVEL DEFINITIONS
000000 = 0              PR0=    0               ;;PRIORITY LEVEL 0
000040 = 40             PR1=    40              ;;PRIORITY LEVEL 1
000100 = 100            PR2=    100             ;;PRIORITY LEVEL 2
000140 = 140            PR3=    140             ;;PRIORITY LEVEL 3
000200 = 200            PR4=    200             ;;PRIORITY LEVEL 4
000240 = 240            PR5=    240             ;;PRIORITY LEVEL 5
000300 = 300            PR6=    300             ;;PRIORITY LEVEL 6
000340 = 340            PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000 = 100000         SW15=   100000
040000 = 40000          SW14=   40000
020000 = 20000          SW13=   20000
010000 = 10000          SW12=   10000
```

```
                                SW11=   4000
                                SW10=   2000
                                SW09=   1000
                                SW08=   400
                                SW07=   200
                                SW06=   100
                                SW05=   40
                                SW04=   20
                                SW03=   10
                                SW02=   4
                                SW01=   2
                                SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                                BIT15=  100000
                                BIT14=  40000
                                BIT13=  20000
                                BIT12=  10000
                                BIT11=  4000
                                BIT10=  2000
                                BIT09=  1000
                                BIT08=  400
                                BIT07=  200
                                BIT06=  100
                                BIT05=  40
                                BIT04=  20
                                BIT03=  10
                                BIT02=  4
                                BIT01=  2
                                BIT00=  1
                                .EQUIV  BIT09,BIT9
                                .EQUIV  BIT08,BIT8
                                .EQUIV  BIT07,BIT7
                                .EQUIV  BIT06,BIT6
                                .EQUIV  BIT05,BIT5
                                .EQUIV  BIT04,BIT4
                                .EQUIV  BIT03,BIT3
                                .EQUIV  BIT02,BIT2
                                .EQUIV  BIT01,BIT1
                                .EQUIV  BIT00,BIT0

                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                          TBITVEC=14              ;;"T" BIT
000014                          TRTVEC= 14              ;;TRACE TRAP
```

```
                        BPTVEC= 14
                        IOTVEC= 20
                        PWRVEC= 24
                        EMTVEC= 30
                        TRAPVEC= 34
                        TKVEC=  60
                        TPVEC=  64
                        PIRQVEC= 240

::BREAKPOINT TRAP (BPT)
::INPUT/OUTPUT TRAP (IOT) **SCOPE**
::POWER FAIL
::EMULATOR TRAP (EMT) **ERROR**
::"TRAP" TRAP
::TTY KEYBOARD VECTOR
::TTY PRINTER VECTOR
::PROGRAM INTERRUPT REQUEST VECTOR
```

```
                        172600          ABASE=172600
                        100360          AVECT1=100360
                        000200          APRIOR=200
                        000003          ADEVM=3

                                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                                        ;*
                                        ;*      SWITCH                  USE
                                        ;*      ------          --------------------
                                        ;*        15            HALT ON ERROR
                                        ;*        14            LOOP ON TEST
                                        ;*        13            INHIBIT ERROR TYPEOUTS
                                        ;*        12            INHIBIT SUB-TEST DELAY
                                        ;*        10            INHIBIT VISUAL SYNC/CHARACTER TEST PATTERNS
                                        ;*         8            LOOP ON TEST IN SWR<7:0>
                                        ;*         9            INHIBIT VISUAL BIT MAP TEST PATTERNS
                                        .SBTTL  TRAP CATCHER

                        000000                  .=0
                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                        000174                  .=174
000174  000000         DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000         SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                        .SBTTL  STARTING ADDRESS(ES)
000200  000137  001572          JMP     @#BEGIN ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  015442          JMP     @#HATCH                 ;DISPLAY CROSS HATCH PATTERN
000210  000137  015642          JMP     @#CROSS                 ;DISPLAY CROSS HAIRS FOR OPERATOR
000214  000137  016322          JMP     FRANKD                  ;DISPLAY CROSS HAIRS WITH TWO MAPS
000220  000137  001E64          JMP     BURNIN                  ;JUMP TO  SET "BURN-IN" MODE FLAG
000224  000137  016416          JMP     BUSOUT                  ;JUMP TO BUS TIME-OUT LOOP
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
                                        $SVPC=.                 ;SAVE PC
                                        .=46
000046  000230                  $ENDAD                          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                                        .=52
000052  000000                  .WORD   0                       ;;2)SET LOC.52 TO ZERO
                                        .=$SVPC                 ;; RESTORE PC
                                        .=1000
                                .SBTTL  APT PARAMETER BLOCK

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

001000                          $APTHD:
001000  000000          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001172          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000000          $TSTM:  .WORD           ;;RUN TIM OF LONGEST TEST
001006  000000          $PASTM: .WORD           ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000000          $UNITM: .WORD           ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000032                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                    001100              .=1100
001100                          $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                          .WORD   0
001102  000                     $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000                     $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000                     $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001                     $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134  000                     $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000                     $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560                  ;;TTY KBD STATUS
001146  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000                     $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002                     $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012                     $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000                     $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000                  $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                                ;;WHICH  ($REGO) WAS OBTAINED
001162  000000                  $REGO:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000                  $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001166  077                     $QUES:  .ASCII  /?/             ;;QUESTION MARK
001167  015                     $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001170  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;**********************************************************************
                                .EVEN
001172                          $MAIL:                          ;;APT MAILBOX
001172  000000                  $MSGTY: .WORD   AMSGTY           ;;MESSAGE TYPE CODE
001174  000000                  $FATAL: .WORD   AFATAL           ;;FATAL ERROR NUMBER
001176  000000                  $TESTN: .WORD   ATESTN           ;;TEST NUMBER
001200  000000                  $PASS:  .WORD   APASS            ;;PASS COUNT
001202  000000                  $DEVCT: .WORD   ADEVCT           ;;DEVICE COUNT
001204  000000                  $UNIT:  .WORD   AUNIT            ;;I/O UNIT NUMBER
```

```
001236  000000  $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
001210  000000  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001212          $ETABLE:                ;;APT ENVIRONMENT TABLE
001212      003 $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001213      000 $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001214  000000  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001216  000000  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001220  000000  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                :*                      BITS 15-11=CPU TYPE
                :*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                :*                              11/70=06,PDQ=07,Q=10
                :*                      BIT 10=REAL TIME CLOCK
                :*                      BIT  9=FLOATING POINT PROCESSOR
                :*                      BIT  8=MEMORY MANAGEMENT
001222      000 $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001223      000 $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                :*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                :*                              900 NSEC CORE=001
                :*                              300 NSEC BIPOLAR=002
                :*                              500 NSEC MOS=003
001224  000000  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                :*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001226      000 $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001227      000 $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001230  000000  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001232      000 $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001233      000 $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001234  000000  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001236      000 $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001237      000 $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001240  000000  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001242  100360  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001244  000000  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001246  172600  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001250  000003  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001252  000000  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
001254  000000  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
001256          $ETEND:
                .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ::POINTS TO THE ERROR MESSAGE
                                ;*      DH              ::POINTS TO THE DATA HEADER
                                ;*      DT              ::POINTS TO THE DATA
                                ;*      DF              ::POINTS TO THE DATA FORMAT

001256                          $ERRTB:

                                ;ITEM   1

001256  023032                          EM1                             ;BUS ERROR TRAP ON CHARACTER ADDRESSES
001260  024004                          DH1                             ;ERRPC  ADR
001262  024226                          DT1                             ;$ERRPC $BDDAT
001264  024322                          DF0

                                ;ITEM   2
001266  023133                          EM2                             ;BUS ERROR TRAP ON BIT MAP ADDRESSES
001270  024004                          DH1                             ;ERRPC  ADR
001272  024226                          DT1                             ;$ERRPC $BDDAT
001274  024322                          DF0

                                ;ITEM   3
001276  023214                          EM3                             ;CHARACTER STATUS REGISTER ERROR
001300  024060                          DH3                             ;ERRPC  ADR     GOOD    BAD
001302  024234                          DT3                             ;$ERRPC VCCSR   $GDDAT  $BDDAT
001304  024322                          DF0

                                ;ITEM   4
001306  023254                          EM4                             ;VERTICAL SYNC ERROR
001310  024105                          DH4                             ;ERRPC  ADR     TIME0   TIME1
001312  024246                          DT4                             ;$ERRPC VCCSR   SAVE0   SAVE2
001314  024322                          DF0

                                ;ITEM   5
001316  023300                          EM5                             ;VERTICAL SYNC ERROR/
001320  024135                          DH5                             ;ERRPC  VCCSR
001322  024260                          DT5                             ;$ERRPC VCCSR
001324  024322                          DF0

                                ;ITEM   6
001326  023336                          EM6                             ;BIT MAP STATUS REGISTER ERROR
001330  024151                          DH6                             ;ERRPC  ADR     GOOD    BAD
001332  024266                          DT6                             ;$ERRPC VGCSR   $GDDAT  $BDDAT
001334  024322                          DF0
```

```
                                        ;ITEM   7
001336  023374                                  EM7                     ;BIT MAP P.C. REGISTER ERROR
001340  024151                                  DH6                     ;ERRPC  VGCSR   GOOD    BAD
001342  024266                                  DT6                     ;$ERRPC VGCSR   $GDDAT  $BDDAT
001344  024322                                  DFD

                                        ;ITEM   10
001346  023430                                  EM10                    ;BIT MAP P.C INCREMENT ERROR
001350  024151                                  DH6                     ;ERRPC  VGCSR   GOOD    BAD
001352  024266                                  DT6                     ;$ERRPC VGCSR   $GDDAT  $BDDAT      :
001354  024322                                  DFD

                                        ;ITEM   11
001356  023465                                  EM11                    ;WORD ERROR ON PIXEL BIT 0
001360  024151                                  DH6                     ;ERRPC  VGCSR   GOOD    BAD
001362  024266                                  DT6                     ;$ERRPC VGCSR   $GDDAT  $BDDAT
001364  024322                                  DFD

                                        ;ITEM   12
001366  023524                                  EM12                    ;BYTE ERROR ON PIXEL BIT 0
001370  024151                                  DH6                     ;ERRPC  VGCSR   GOOD    BAD
001372  024266                                  DT6                     ;$ERRPC VGCSR   $GDDAT  $BDDAT
001374  024322                                  DFD

                                        ;ITEM   13
001376  023563                                  EM13                    ;CHARACTER READY ERROR
001400  024151                                  DH6                     ;ERRPC  VCCSR   GOOD    BAD
001402  024266                                  DT6                     ;$ERRPC VCCSR   $GDDAT  $BDDAT
001404  024322                                  DFD

                                        ;ITEM   14
001406  023611                                  EM14                    ;BIT MAP READY ERROR
001410  024151                                  DH6                     ;ERRPC  VGCSR   GOOD    BAD
001412  024266                                  DT6                     ;$ERRPC VGCSR   $GDDAT  $BDDAT
001414  024322                                  DFD

                                        ;ITEM   15
001416  023611                                  EM14                    ;BIT MAP READY FAILED TO SET
001420  024176                                  DH15                    ;ERRPC  VGCSR
001422  024300                                  DT15                    ;$ERRPC $VGCSR
001424  024322                                  DFD

                                        ;ITEM   16
001426  023635                                  EM16            ;PREVIOUSLY EXISTING BIT MAP DOES NOT EXIST NOW
001430  024212                                  DH16            ;ERRPC  VSV01
001432  024306                                  DT16            ;$ERRPC $BDDAT
001434  024322                                  DFD

                                        ;ITEM   17
001436  023714                                  EM17            ;PREVIOUSLY EXISTING VTV01 CONTROLLER DOES NOT EXIST NOW
001440  024212                                  DH16            ;ERRPC  VSV01
001442  024306                                  DT16            ;$ERRPC $GDADR
001444  024322                                  DFD
```

```
                                        ;VTV01 BUS ADDRESSES AND VECTOR

                                                .SBTTL  VSV01 BUS ADDRESSES AND VECTORS

001446  172600                          VCCSR:  ABASE
001450  172602                          VCHRLO: ABASE+2
001452  172603                          VCHRHI: ABASE+3
001454  172604                          VCPOS:  ABASE+4
001456  172605                          VCPOSH: ABASE+5
001460  172620                          VGCSR:  ABASE+20
001462  172622                          VGPC:   ABASE+22
001464  172624                          VGBUF:  ABASE+24
001466  172630                          VGCOLR: ABASE+30

001470  130360                          VCINT:  AVECT1
001472  130362                          VCINTI: AVECT1+2

                                        ;MISC. LOCATIONS

                                                .SBTTL  MISC. LOCATIONS

001474  000000                          $TEMP:  0
001476  000000                          $TEMP0: 0
001500  000105                          KE:     105             ;ASCII CHARACTER USED IN FULL SCREEN PATTERN
001502  000000                          TEMP1:  0
001504  000000                          TEMP2:  0
001506  000000                          TEMP3:  0
001510  000000                          SAVE0:  0
001512  000000                          SAVE1:  0
001514  000000                          SAVE2:  0
001516  000000                          SAVE3:  0
001520  003100                          TOTALC: 1600.
001522  000100                          WIDTH:  64.
001524  000030                          VH0:    24.
001526  000014                          VH1:    12.
001530  000000                          STCHAR: 0
001532  000137                          LASTCH: 137
001534  000000                          TEMP0:  0
001536  000400                          XHAIR:  256.
001540  000400                          YHAIR:  256.
001542  010000                          NUMPIX: 4096.
001544  000000                          BURN:   0
001546  000001                          COAXSW: 1               ;BIT 0-4 SELECT RELAY CONTACTS IN "BURN-IN" MODE
001550  167772                          COAX:   167772          ;BUS ADDRESS OF COAX SWITCH
001552  000000                          MSAVE0: 0
001554  000000                          MSAVE1: 0
001556  000000                          MSAVE2: 0
001560  000000                          MSAVE3: 0
001562  000000                          MSAVE4: 0
```

```
000439  001564  005237  001544          BURNIN: INC     BURN            ;SET BURN-IN MODE FLAG
000440  001570  000402                          BR      CONT
000441  001572  005037  001544          BEGIN:  CLR     BURN            ;CLEAR "BURN-IN " MODE FLAG
000442  001576                          CONT:
000443                                  .SBTTL  INITIALIZE THE COMMON TAGS
000444                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
000445  001576  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
000446  001602  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
000447  001604  022706  001140                  CMP     #SWR,R6 ;;DONE?
000448  001610  001374                          BNE     .-6             ;;LOOP BACK IF NO
000449  001612  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
000450                                  ;;INITIALIZE A FEW VECTORS
000451  001616  012737  017706  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
000452  001624  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
000453  001632  012737  020036  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
000454  001640  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
000455  001646  012737  022630  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
000456  001654  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
000457  001662  012737  021002  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
000458  001670  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
000459  001676  013737  003252  003244          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
000460  001704  012737  001704  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
000461                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
000462                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
000463  001712  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
000464  001716  012737  001752  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
000465  001724  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
000466  001732  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
000467  001740  022777  177777  177172          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
000468  001746  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
000469                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
000470  001750  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
000471  001752  012716  001760          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
000472  001756  000002                          RTI
000473  001760  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
000474  001766  012737  000174  001142          MOV     #DISPREG,DISPLAY
000475  001774  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
000476
000477  002000  005037  001200                  CLR     $PASS           ;;CLEAR PASS COUNT
000478  002004  132737  000200  001213          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
000479  002012  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
000480  002014  012737  001214  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
000481  002022                          67$:
000482  002022  005037  001552                  CLR     MSAVE0                  ;RESET MAP COUNTERS
000483  002026  005037  001554                  CLR     MSAVE1
000484  002032  005037  001556                  CLR     MSAVE2                  ; AND CONTROLLER
000485  002036  005037  001560                  CLR     MSAVE3
000486  002042  005037  001562                  CLR     MSAVE4                  ; LOCATIONS
000487  002046  005737  000042                  TST     @#42                    ;TEST IF IN CHAIN MODE
000488  002052  001002                          BNE     RESTAT
000489  002054  104401                          TYPE
000490  002056  022712                          TITLE
000491  002060  000005          RESTAT: RESET
```

```
                                        .SBTTL
                                        .SBTTL  DETERMINE THE NUMBER OF CONTROLLERS TO TEST

002062  013737  001246  001120  DETER0: MOV     $BASE,$GDADR            ;LOAD BASE ADDR.
002070  012737  002242  007366          MOV     #4$,RETURN              ;LOAD RETURN ADDR. AFTER TESTING
002076  004737  003340                  JSR     PC,DETECT               ;DETERMINE # OF MAPS
002102  000453                          BR      3$                      ;BR IF CONTROLLER DOESN'T EXIST
002104  004737  003576                  JSR     PC,LODADR               ;FIX THE BUS ADDRESSES
002110  005737  001254                  TST     $CDW2                   ;TEST IF THE OPERATOR HAS SELECTED SOME MAPS
002114  001407                          BEQ     6$                      ;BR IF NOT
002116  013737  001254  001252          MOV     $CDW2,$CDW1             ;LOAD HIS SELECTION
002124  052737  100000  001552          BIS     #BIT15,MSAVE0   ;SET CONTROLLER EXISTS FLAG
002132  000430                          BR      2$                      ;AND TEST THEM
002134  105737  001213          6$:     TSTB    $ENVM                   ;TEST IF "DO NOT SIZE" IS SET
002140  100007                          BPL     5$                      ;BR IF NOT
002142  013737  001250  001252          MOV     $DEVM,$CDW1             ;USE VALUE SUPPLIED BY "APT"
002150  052737  100000  001552          BIS     #BIT15,MSAVE0   ;SET CONTROLLER EXISTS FLAG
002156  000416                          BR      2$
002160  005737  001200          5$:     TST     $PASS                   ;TEST IF FIRST PASS
002164  001006                          BNE     1$                      ;BR IF NOT
002166  013737  001512  001552          MOV     SAVE1,MSAVE0            ;SAVE THE NUMBER OF MAPS
002174  052737  100000  001552          BIS     #BIT15,MSAVE0           ;SET "CONTROLLER EXISTS" FLAG
002202  123737  001512  001552  1$:     CMPB    SAVE1,MSAVE0            ;COMPARE IF ANY CHANGE IN # OF MAPS
002210  001401                          BEQ     2$                      ;BR IF NOT
002212  104016                          ERROR   16              ;PREVIOUSLY EXISTING BIT MAP DOES NOT EXIST NOW
002214  012737  000001  001546  2$:     MOV     #1,COAXSW               ;LOAD SWITCH VALUE
002222  005037  001102                  CLR     $TSTNM                  ;RESET TEST #
002226  000137  003762                  JMP     TST1                    ;AND TEST THIS CONTROLLER
002232  005737  001552          3$:     TST     MSAVE0                  ;TEST IF CONTROLLER EXISTED AT ONE TIME
002236  100001                          BPL     4$                      ;BR IF NO
002240  104017                          ERROR   17      ;PREVIOUSLY EXISTING  CONTROLLER AT 172600 DOES NOT EXIST NOW
002242  000240                  4$:     NOP
002244  005737  001552                  TST     MSAVE0                  ;TEST IF THE FIRST CONTROLLER EXISTS
002250  100413                          BMI     10$                     ;BR IF IT'S THERE
002252  012737  002274  001110          MOV     #7$,$LPERR              ;LOAD LOOP RETURN
002260  012737  002274  001106          MOV     #7$,$LPADR              ;LOAD RETURN ADDRESS
002266  013737  001246  001126          MOV     $BASE,$BDDAT            ;LOAD BASE ADDRESS FOR TYPEOUT
002274  104001                  7$:     ERROR   1       ;** FATAL ERROR **      ;VTV01 AT ADDRESS ($BASE) DOESN'T EXIST
002276  000776                          BR      7$                      ;FATAL ERROR
002300  005737  001544          10$:    TST     BURN                    ;TEST IF "BURN-IN" MODE
002304  001002                          BNE     DETER1                  ;BR AND TEST FOR MORE MAPS
002306  000137  003222                  JMP     DETEC7                  ;REPORT "EOP"
002312  012737  171600  001120  DETER1: MOV     #171600,$GDADR          ;LOAD BUS ADDRESS
002320  012737  002472  007366          MOV     #4$,RETURN              ;LOAD RETURN ADDR. AFTER TESTING
002326  004737  003340                  JSR     PC,DETECT               ;DETERMINE # OF MAPS
002332  000453                          BR      3$                      ;BR IF CONTROLLER DOESN'T EXIST
002334  004737  003576                  JSR     PC,LODADR               ;FIX THE BUS ADDRESSES
002340  005737  001254                  TST     $CDW2                   ;TEST IF THE OPERATOR HAS SELECTED SOME MAPS
002344  001407                          BEQ     6$                      ;BR IF NOT
002346  013737  001254  001252          MOV     $CDW2,$CDW1             ;LOAD HIS SELECTION
002354  052737  100000  001554          BIS     #BIT15,MSAVE1   ;SET CONTROLLER EXISTS FLAG
002362  000430                          BR      2$                      ;AND TEST THEM
002364  105737  001213          6$:     TSTB    $ENVM                   ;TEST IF "DO NOT SIZE" IS SET
002370  100007                          BPL     5$                      ;BR IF NOT
002372  013737  001250  001252          MOV     $DEVM,$CDW1             ;USE VALUE SUPPLIED BY "APT"
002400  052737  100000  001554          BIS     #BIT15,MSAVE1   ;SET CONTROLLER EXISTS FLAG
```

```
548  002406  000416                          BR      2$
549  002410  005737  001200          5$:     TST     $PASS                   ;TEST IF FIRST PASS
550  002414  001006                          BNE     1$                      ;BR IF NOT
551  002416  013737  001512  001554          MOV     SAVE1,MSAVE1            ;SAVE THE NUMBER OF MAPS
552  002424  052737  100000  001554          BIS     #BIT15,MSAVE1           ;SET "CONTROLLER EXISTS" FLAG
553  002432  123737  001512  001554  1$:     CMPB    SAVE1,MSAVE1            ;COMPARE IF ANY CHANGE IN # OF MAPS
554  002440  001401                          BEQ     2$                      ;BR IF NOT
555  002442  104016                          ERROR   16              ;PREVIOUSLY EXISTING BIT MAP DOES NOT EXIST NOW
556  002444  012737  000002  001546  2$:     MOV     #2,COAXSW               ;LOAD SWITCH VALUE
557  002452  005037  001102                  CLR     $TSTNM                  ;RESET TEST #
558  002456  000137  003762                  JMP     TST1                    ;AND TEST THIS CONTROLLER
559  002462  005737  001554          3$:     TST     MSAVE1                  ;TEST IF CONTROLLER EXISTED AT ONE TIME
560  002466  100001                          BPL     4$                      ;BR IF NO
561  002470  104017                          ERROR   17      ;PREVIOUSLY EXISTING  CONTROLLER AT 171600 DOES NOT EXIST NOW
562  002472  000240                  4$:     NOP
563  002474  012737  171300  001120  DETER2: MOV     #171300,$GDADR          ;LOAD BUS ADDRESS
564  002502  012737  002654  007366          MOV     #4$,RETURN              ;LOAD RETURN ADDR. AFTER TESTING
565  002510  004737  003340                  JSR     PC,DETECT               ;DETERMINE # OF MAPS
566  002514  000453                          BR      3$                      ;BR IF CONTROLLER DOESN'T EXIST
567  002518  004737  003576                  JSR     PC,LODADR               ;FIX THE BUS ADDRESSES
568  002522  005737  001254                  TST     $CDW2                   ;TEST IF THE OPERATOR HAS SELECTED SOME MAPS
569  002526  001407                          BEQ     6$                      ;BR IF NOT
570  002530  013737  001254  001252          MOV     $CDW2,$CDW1             ;LOAD HIS SELECTION
571  002536  052737  100000  001556          BIS     #BIT15,MSAVE2   ;SET CONTROLLER EXISTS FLAG
572  002544  000430                          BR      2$                      ;AND TEST THEM
573  002546  105737  001213          6$:     TSTB    $ENVM                   ;TEST IF "DO NOT SIZE" IS SET
574  002552  100007                          BPL     5$                      ;BR IF NOT
575  002554  013737  001250  001252          MOV     $DEVM,$CDW1             ;USE VALUE SUPPLIED BY "APT"
576  002562  052737  100000  001556          BIS     #BIT15,MSAVE2   ;SET CONTROLLER EXISTS FLAG
577  002570  000416                          BR      2$
578  002572  005737  001200          5$:     TST     $PASS                   ;TEST IF FIRST PASS
579  002576  001006                          BNE     1$                      ;BR IF NOT
580  002600  013737  001512  001556          MOV     SAVE1,MSAVE2            ;SAVE THE NUMBER OF MAPS
581  002606  052737  100000  001556          BIS     #BIT15,MSAVE2           ;SET "CONTROLLER EXISTS" FLAG
582  002614  123737  001512  001556  1$:     CMPB    SAVE1,MSAVE2            ;COMPARE IF ANY CHANGE IN # OF MAPS
583  002622  001401                          BEQ     2$                      ;BR IF NOT
584  002624  104016                          ERROR   16              ;PREVIOUSLY EXISTING BIT MAP DOES NOT EXIST NOW
585  002626  012737  000004  001546  2$:     MOV     #4,COAXSW               ;LOAD SWITCH VALUE
586  002634  005037  001102                  CLR     $TSTNM                  ;RESET TEST #
587  002640  000137  003762                  JMP     TST1                    ;AND TEST THIS CONTROLLER
588  002644  005737  001556          3$:     TST     MSAVE2                  ;TEST IF CONTROLLER EXISTED AT ONE TIME
589  002650  100001                          BPL     4$                      ;BR IF NO
590  002652  104017                          ERROR   17      ;PREVIOUSLY EXISTING  CONTROLLER AT 171300 DOES NOT EXIST NOW
591  002654  000240                  4$:     NOP
592  002656  012737  172300  001120  DETER3: MOV     #172300,$GDADR          ;LOAD BUS ADDRESS
593  002664  012737  003036  007366          MOV     #4$,RETURN              ;LOAD RETURN ADDR. AFTER TESTING
594  002672  004737  003340                  JSR     PC,DETECT               ;DETERMINE # OF MAPS
595  002676  000453                          BR      3$                      ;BR IF CONTROLLER DOESN'T EXIST
596  002700  004737  003576                  JSR     PC,LODADR               ;FIX THE BUS ADDRESSES
597  002704  005737  001254                  TST     $CDW2                   ;TEST IF THE OPERATOR HAS SELECTED SOME MAPS
598  002710  001407                          BEQ     6$                      ;BR IF NOT
599  002712  013737  001254  001252          MOV     $CDW2,$CDW1             ;LOAD HIS SELECTION
600  002720  052737  100000  001560          BIS     #BIT15,MSAVE3   ;SET CONTROLLER EXISTS FLAG
601  002726  000430                          BR      2$                      ;AND TEST THEM
602  002730  105737  001213          6$:     TSTB    $ENVM                   ;TEST IF "DO NOT SIZE" IS SET
603  002734  100007                          BPL     5$                      ;BR IF NOT
```

```
002736  013737  001250  001252             MOV     $DEVM,$CDW1             ;USE VALUE SUPPLIED BY "APT"
002744  052737  100000  001560             BIS     #BIT15,MSAVE3   ;SET CONTROLLER EXISTS FLAG
002752  000416                             BR      2$
002754  005737  001200          5$:        TST     $PASS                   ;TEST IF FIRST PASS
002760  001006                             BNE     1$                      ;BR IF NOT
002762  013737  001512  001560             MOV     SAVE1,MSAVE3            ;SAVE THE NUMBER OF MAPS
002770  052737  100000  001560             BIS     #BIT15,MSAVE3           ;SET "CONTROLLER EXISTS" FLAG
002776  123737  001512  001560  1$:        CMPB    SAVE1,MSAVE3            ;COMPARE IF ANY CHANGE IN # OF MAPS
003004  001401                             BEQ     2$                      ;BR IF NOT
003006  104016                             ERROR   16              ;PREVIOUSLY EXISTING BIT MAP DOES NOT EXIST NOW
003010  012737  000010  001546  2$:        MOV     #10,COAXSW              ;LOAD SWITCH VALUE
003016  005037  001102                     CLR     $TSTNM                  ;RESET TEST #
003022  000137  003762                     JMP     TST1                    ;AND TEST THIS CONTROLLER
003026  005737  001560          3$:        TST     MSAVE3                  ;TEST IF CONTROLLER EXISTED AT ONE TIME
003032  100001                             BPL     4$                      ;BR IF NO
003034  104017                             ERROR   17      ;PREVIOUSLY EXISTING  CONTROLLER AT 172300 DOES NOT EXIST NOW
003036  000240                  4$:        NOP
003040  012737  173200  001120  DETER4:    MOV     #173200,$GDADR          ;LOAD BUS ADDRESS
003046  012737  003220  007366             MOV     #4$,RETURN              ;LOAD RETURN ADDR. AFTER TESTING
003054  004737  003340                     JSR     PC,DETECT               ;DETERMINE # OF MAPS
003060  000453                             BR      3$                      ;BR IF CONTROLLER DOESN'T EXIST
003062  004737  003576                     JSR     PC,LODADR               ;FIX THE BUS ADDRESSES
003066  005737  001254                     TST     $CDW2                   ;TEST IF THE OPERATOR HAS SELECTED SOME MAPS
003072  001407                             BEQ     6$                      ;BR IF NOT
003074  013737  001254  001252             MOV     $CDW2,$CDW1             ;LOAD HIS SELECTION
003102  052737  100000  001562             BIS     #BIT15,MSAVE4   ;SET CONTROLLER EXISTS FLAG
003110  000430                             BR      2$                      ;AND TEST THEM
003112  105737  001213          6$:        TSTB    $ENVM                   ;TEST IF "DO NOT SIZE" IS SET
003116  100007                             BPL     5$                      ;BR IF NOT
003120  013737  001250  001252             MOV     $DEVM,$CDW1             ;USE VALUE SUPPLIED BY "APT"
003126  052737  100000  001562             BIS     #BIT15,MSAVE4   ;SET CONTROLLER EXISTS FLAG
003134  000416                             BR      2$
003136  005737  001200          5$:        TST     $PASS                   ;TEST IF FIRST PASS
003142  001006                             BNE     1$                      ;BR IF NOT
003144  013737  001512  001562             MOV     SAVE1,MSAVE4            ;SAVE THE NUMBER OF MAPS
003152  052737  100000  001562             BIS     #BIT15,MSAVE4           ;SET "CONTROLLER EXISTS" FLAG
003160  123737  001512  001562  1$:        CMPB    SAVE1,MSAVE4            ;COMPARE IF ANY CHANGE IN # OF MAPS
003166  001401                             BEQ     2$                      ;BR IF NOT
003170  104016                             ERROR   16              ;PREVIOUSLY EXISTING BIT MAP DOES NOT EXIST NOW
003172  012737  000020  001546  2$:        MOV     #20,COAXSW              ;LOAD SWITCH VALUE
003200  005037  001102                     CLR     $TSTNM                  ;RESET TEST #
003204  000137  003762                     JMP     TST1                    ;AND TEST THIS CONTROLLER
003210  005737  001562          3$:        TST     MSAVE4                  ;TEST IF CONTROLLER EXISTED AT ONE TIME
003214  100001                             BPL     4$                      ;BR IF NO
003216  104017                             ERROR   17      ;PREVIOUSLY EXISTING  CONTROLLER AT 173200 DOES NOT EXIST NOW
003220  000240                  4$:        NOP
```

```
003222                          DETEC7:
                                .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 RESTAT

003222                          $EOP:
003222  000240                          NOP
003224  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
003230  005237  001200                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
003234  042737  100000  001200          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
003242  005327                          DEC     (PC)+           ;;LOOP?
003244  000001                  $EOPCT: .WORD   1
003246  003022                          BGT     $DOAGN          ;;YES
003250  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
003252  000001                  $ENDCT: .WORD   1
003254  003244                          $EOPCT
003256  104401  003323                  TYPE    ,$ENDMG         ;;TYPE "END PASS #"
003262  013746  001200                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
003266  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
003270  104401  003320                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
003274  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
003300  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
003302  000005                          RESET                   ;;CLEAR THE WORLD
003304  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
003306  000240                          NOP                     ;;SAVE ROOM
003310  000240                          NOP                     ;;FOR
003312  000240                          NOP                     ;;ACT11
003314                          $DOAGN:
003314  000137                          JMP     @(PC)+          ;;RETURN
003316  002060                  $RTNAD: .WORD   RESTAT
003320     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
003323     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
003330  050040  051501  020123
003336  000043
```

```
                                        .SBTTL  DETERMINE THE NUMBER OF BIT MAP'S ON A VTV01
                                        .SBTTL

003340  013737  001120  001124  DETECT: MOV     $GDADR,$GDDAT           ;GET BASE ADDRESS
003346  012737  003570  000004          MOV     #7$,ERRVEC              ;LOAD RETURN
003354  005777  175544                  TST     @$GDDAT                 ;TEST THE CHARACTER STATUS ADDRESS
003360  005037  001504                  CLR     TEMP2                   ;THE CONTROLLER EXISTS NOW TEST THE MAPS
003364  005037  001506                  CLR     TEMP3                   ;
003370  012737  003434  000004          MOV     #2$,ERRVEC              ;LOAD BUS TRAP RETURN
003376  013737  001124  001126          MOV     $GDDAT,$BDDAT           ;GET THE ADDRESS
003404  062737  000020  001126  1$:     ADD     #20,$BDDAT              ;UPDATE THE ADDRESS
003412  005777  175510                  TST     @$BDDAT                 ;TEST FOR A MAP
003416  005237  001504                  INC     TEMP2                   ;YES IT EXISTS UPDATE THE COUNT
003422  022737  000006  001504          CMP     #6,TEMP2                ;TEST FO LAST
003430  001402                          BEQ     3$
003432  000764                          BR      1$
003434  022626                  2$:     CMP     (SP)+,(SP)+             ;POP THE STACK
003436  012737  000006  000004  3$:     MOV     #ERRVEC+2,ERRVEC                ;RESET BUS TRAP
003444  005037  000006                  CLR     ERRVEC+2
003450  005737  000042                  TST     @#42                    ;TEST IF "CHAIN MODE"
003454  001020                          BNE     4$                      ;BR IF CHAIN MODE
003456  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS
003462  001015                          BNE     4$                      ;BR IF NOT
003464  104401  022750                  TYPE    ,VTHDR1                 ;REPORT # OF MAPS
003470  013746  001124                  MOV     $GDDAT,-(SP)            ;ON THE CONTROLLER ADDRESS
003474  104402                          TYPOC
003476  104401  022774                  TYPE    ,VTHDR2                 ; "HAS"
003502  013746  001504                  MOV     TEMP2,-(SP)             ;#N
003506  104403                          TYPOS
003510     001     000                  .BYTE   1,0                     ;ONE DIGIT
003512  104401  023002                  TYPE    ,VTHDR3                 ;"MAPS INSTALLED"
003516  013737  001504  001512  4$:     MOV     TEMP2,SAVE1             ;SAVE # OF MAPS
003524  001413                          BEQ     6$                      ;BR IF NONE
003526  012737  000001  001506          MOV     #1,TEMP3                ;LOAD INDICATOR
003534  005337  001504          5$:     DEC     TEMP2                   ;CONVERT FOR LATER
003540  001405                          BEQ     6$
003542  006337  001506                  ASL     TEMP3                   ;MOVE
003546  005237  001506                  INC     TEMP3
003552  000770                          BR      5$
003554  013737  001506  001252  6$:     MOV     TEMP3,$CDW1             ;LOAD THE DETECTED MAP BIT'S
003562  062716  000002                  ADD     #2,(SP)                 ;UPDATE EXIT ADDRESS
003566  000207                          RTS     PC                      ;RETURN

003570  022626                  7$:     CMP     (SP)+,(SP)+
003572  000240                          NOP
003574  000207                          RTS     PC                      ;RETURN
```

```
                                LODADR: MOV     #VCCSR,R0               ;LOAD ADDRESS POINTER
                                        MOV     $GDADR,R1               ;LOAD ADDRESS VALUE
                                10$:    MOV     R1,(R0)+                ;LOAD THE ADDRESSES
003610  022703  001470                  CMP     #VCINT,R0               ;TEST FOR END
003614  001374                          BNE     10$
003616  062737  000002  001450          ADD     #2,VCHRLO               ;ADJUST THE ADDRESSES
003624  062737  000003  001452          ADD     #3,VCHRHI
003632  062737  000004  001454          ADD     #4,VCPOS
003640  062737  000005  001456          ADD     #5,VCPOSH
003646  062737  000020  001460          ADD     #20,VGCSR
003654  062737  000022  001462          ADD     #22,VGPC
003662  062737  000024  001464          ADD     #24,VGBUF
003670  062737  000030  001466          ADD     #30,VGCOLR
003676  013737  001242  001470          MOV     $VECT1,VCINT            ;LOAD BASIC INTR. VECTOR
003704  042737  160000  001470          BIC     #160000,VCINT           ;MASK OFF BR LEVEL BITS
003712  013737  001470  001472          MOV     VCINT,VCINT1
003720  062737  000002  001472          ADD     #2,VCINT1
003726  000207                          RTS     PC                      ;EXIT

003730  062737  000020  001460  MOVADR: ADD     #20,VGCSR               ;UPDATE BUS ADDRESSES TO NEXT MAP
003736  062737  000020  001462          ADD     #20,VGPC
003744  062737  000020  001464          ADD     #20,VGBUF
003752  062737  000020  001466          ADD     #20,VGCOLR
003760  000207                          RTS     PC                      ;EXIT

                        ;;**************************************************************
                        ;*TEST 1        VERIFY ALL BUS ADDRESSES OF THE CHARACTER GEN.
                        ;;**************************************************************
003762  000004          TST1:   SCOPE
003764  012737  177777  001204          MOV     #-1,$UNIT               ;INDIACTE A SYNC/ CHARACTER GENERATOR LOGIC ERRO
003772  012737  004036  001110          MOV     #1$,$LPERR              ;LOAD ERROR RETURN
004000  013737  001120  001124          MOV     $GDADR,$GDDAT           ;LOAD BASE ADDR.
004006  062737  000006  001124          ADD     #6,$GDDAT               ;ADJUST TO LAST ADDRESS
004014  013737  001120  001126          MOV     $GDADR,$BDDAT           ;LOAD TEST ADDRESS
004022  012737  004054  000004          MOV     #2$,@#ERRVEC            ;LOAD BUS TRAP RETURN
004030  005777  175072          1$:     TST     @$BDDAT                 ;TEST IF ADDRESS EXISTS
004034  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST IF FINISHED ALL ADDR.
004042  001407                          BEQ     3$                      ;;BR IF ALL DONE
004044  062737  000002  001126          ADD     #2,$BDDAT               ;ADJUST ADDRESS POINTER
004052  000766                          BR      1$                      ;TRY AGAIN

004054  022626          2$:     CMP     (SP)+,(SP)+             ;CLEAN STACK
004056  104001                  ERROR   1                       ;BUS ERROR WITH CHARACTER ADDRESS
004060  000763                  BR      1$

004062  012737  000006  000004  3$:     MOV     #ERRVEC+2,@#ERRVEC      ;RESET BUS TRAP
004070  005037  000006                  CLR     @#ERRVEC+2
```

```
                                ;;*****************************************************************
                                ;*TEST 2        TEST ALL EXISTANT BUS ADDRESSES FOR THE SELECTED BIT MAPS
                                ;;*****************************************************************
004074  000004                TST2:   SCOPE
004076  012737  004150  001110        MOV     #1$,$LPERR              ;LOAD RETURN IF ERROR
004104  013737  001120  001502        MOV     $GDIOR,TEMP1            ;GET BASE ADDRESS
004112  062737  000020  001502        ADD     #20,TEMP1               ;UPDATE TO THE MAP ADDRESS
004120  013737  001252  004360        MOV     $CDW1,DEVTMP            ;GET THE SELECTED MAPS
004126  042737  177700  004360        BIC     #177700,DEVTMP          ;MASK TO LOWER SIX BITS
004134  001426                        BEQ     3$                      ;BR IF NO MAPS SELECTED
004136  005037  001204                CLR     $UNIT                   ;CLEAR MAP INDICATOR
004142  012737  000001  001506        MOV     #BIT0,TEMP3             ;LOAD TESTING BIT
004150  033737  001506  004360  1$:   BIT     TEMP3,DEVTMP            ;TEST IF THIS BIT MAP IS SELECTED
004156  001402                        BEQ     2$                      ;BR IF NOT
004160  000137  004214                JMP     10$                     ;SELECTED - TEST THIS MAP
004164  062737  000020  001502  2$:   ADD     #20,TEMP1               ;UPDATE TO NEXT BUS ADDRESS
004172  005237  001204                INC     $UNIT                   ;UPDATE MAP INDICATOR
004176  006337  001506                ASL     TEMP3                   ;MOVE THE TEST BIT
004202  022737  000100  001506        CMP     #100,TEMP3              ;TEST IF ANY MORE POSSIBLE MAPS
004210  001357                        BNE     1$                      ;BR IF MORE
004212                          3$:
004212  000463                        BR      TST3            ;;BR TO NEXT TEST

                                ;THE BIT MAP ADDRESS IS IN "TEMP1" - LOAD THE ADDRESS AND
                                ;       START TESTING THAT BIT MAP

004214  012737  004352  000004  10$:   MOV     #20$,@#4                ;LOAD FOR BUS TRAP
004222  013737  001502  001126        MOV     TEMP1,$BDDAT            ;LOAD ADDRESS POINTER
004230  005777  174672                TST     @$BDDAT                 ;TEST AN ADDRESS
004234  062737  000002  001126        ADD     #2,$BDDAT               ;UPDATE ADDRESS
004242  005777  174660                TST     @$BDDAT                 ;TEST AN ADDRESS
004246  062737  000002  001126        ADD     #2,$BDDAT               ;UPDATE ADDRESS
004254  005777  174646                TST     @$BDDAT                 ;TEST AN ADDRESS
004260  062737  000002  001126        ADD     #2,$BDDAT               ;UPDATE ADDRESS
004266  005777  174634                TST     @$BDDAT                 ;TEST AN ADDRESS
004272  062737  000002  001126        ADD     #2,$BDDAT               ;UPDATE ADDRESS
004300  005777  174622                TST     @$BDDAT                 ;TEST AN ADDRESS
004304  062737  000002  001126        ADD     #2,$BDDAT               ;UPDATE ADDRESS
004312  005777  174610                TST     @$BDDAT                 ;TEST AN ADDRESS
004316  062737  000002  001126        ADD     #2,$BDDAT               ;UPDATE ADDRESS
004324  005777  174576                TST     @$BDDAT                 ;TEST AN ADDRESS
004330  062737  000002  001126        ADD     #2,$BDDAT               ;UPDATE ADDRESS
004336  005777  174564                TST     @$BDDAT                 ;TEST AN ADDRESS
004342  062737  000002  001126        ADD     #2,$BDDAT               ;UPDATE ADDRESS
004350  000705                        BR      2$                      ;RETURN

004352  022626                20$:    CMP     (SP)+,(SP)+             ;POP STACK
004354  104002                        ERROR   2                       ;REPORT BUS ERROR TRAP ON A SELECTED BIT MAP ADD
004356  000702                        BR      2$                      ;RETURN


004360  000003                DEVTMP: ADEVM                           ;BITS SET INDICATE THE MAPS TO BE TESTED
```

```
                                        ;;*****************************************************************
                                        ;*TEST 3         TEST THE CHAR CURSOR DISABLE BIT
                                        ;;*****************************************************************
004362  000004                  TST3:   SCOPE
004364  012737  177777  001204          MOV     #-1,$UNIT               ;INDICATE A CONTROLLER ERROR
004372  012737  002000  001124          MOV     #BIT10,$GDDAT           ;LOAD EXPECTED VALUE
004400  013777  001124  175040          MOV     $GDDAT,@VCCSR           ;LOAD THE REGISTER
004406  017737  175034  001126          MOV     @VCCSR,$BDDAT           ;READ THE REGISTER
004414  042737  140000  001126          BIC     #140000,$BDDAT          ;MASK TO BITS 15-14
004422  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE FOR SAME
004430  001401                          BEQ     TST4            ;;BR IF EXPECTED
004432  104003                          ERROR   3                       ;'CHAR CURSOR DISABLE' FAILED TO SET

                                        ;;*****************************************************************
                                        ;*TEST 4         TEST THE Y CROSS HAIR ENABLE BIT
                                        ;;*****************************************************************
004434  000004                  TST4:   SCOPE
004436  012737  004000  001124          MOV     #BIT11,$GDDAT           ;LOAD EXPECTED VALUE
004444  013777  001124  174774          MOV     $GDDAT,@VCCSR           ;LOAD THE REGISTER
004452  017737  174770  001126          MOV     @VCCSR,$BDDAT           ;READ THE REGISTER
004460  042737  140000  001126          BIC     #140000,$BDDAT          ;MASK TO BITS 15-14
004466  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE FOR SAME
004474  001401                          BEQ     TST5            ;;BR IF EXPECTED
004476  104003                          ERROR   3                       ;'Y CROSS HAIR ENABLE' FAILED TO SET

                                        ;;*****************************************************************
                                        ;*TEST 5         TEST THE X CROSS HAIR ENABLE BIT
                                        ;;*****************************************************************
004500  000004                  TST5:   SCOPE
004502  012737  010000  001124          MOV     #BIT12,$GDDAT           ;LOAD EXPECTED VALUE
004510  013777  001124  174730          MOV     $GDDAT,@VCCSR           ;LOAD THE REGISTER
004516  017737  174724  001126          MOV     @VCCSR,$BDDAT           ;READ THE REGISTER
004524  042737  140000  001126          BIC     #140000,$BDDAT          ;MASK TO BITS 15-14
004532  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE FOR SAME
004540  001401                          BEQ     TST6            ;;BR IF EXPECTED
004542  104003                          ERROR   3                       ;'X CROSS HAIR ENABLE' FAILED TO SET

                                        ;;*****************************************************************
                                        ;*TEST 6         TEST THE VERT SYNC INTR. ENABLE BIT
                                        ;;*****************************************************************
004544  000004                  TST6:   SCOPE
004546  012746  000340                  MOV     #340,-(SP)              ;RAISE PRIORITY
004552  012746  004560                  MOV     #1$,-(SP)
004556  000002                          RTI
004560  012737  020000  001124  1$:     MOV     #BIT13,$GDDAT           ;LOAD EXPECTED VALUE
004566  013777  001124  174652          MOV     $GDDAT,@VCCSR           ;LOAD THE REGISTER
004574  017737  174646  001126          MOV     @VCCSR,$BDDAT           ;READ THE REGISTER
004602  042737  140000  001126          BIC     #140000,$BDDAT          ;MASK TO BITS 15-14
004610  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE FOR SAME
004616  001401                          BEQ     2$              ;;BR IF EXPECTED
004620  104003                          ERROR   3                       ;'VERT SYNC INTR. ENABLE' FAILED TO SET
004622  042777  020000  174616  2$:     BIC     #BIT13,@VCCSR           ;DISABLE INTR.
```

H02


```
                                ;;**************************************************************
                                ;*TEST 7         TEST THAT "RESET" CLEARS CHARACTER STATUS REG.
                                ;;**************************************************************
004630  000004                  TST7:   SCOPE
004632  005037  001124                  CLR     $GDDAT                          ;CLEAR EXPETED
004636  012777  036000  174602          MOV     #BIT13!BIT12!BIT11!BIT10,@VCCSR ;LOAD THE REGISTER
004644  000005                          RESET                                   ;CLEAR THE WORLD
004646  017737  174574  001126          MOV     @VCCSR,$BDDAT                   ;READ CHARACTER STATUS
004654  042737  141777  001126          BIC     #141777,$BDDAT                  ;MASK TO OTHER BITS
004662  001401                          BEQ     TST10           ;;BR IF ALL BITS CLEARED
004664  104003                          ERROR   3                               ;RESET FAILED TO CLEAR CHARACTER STATUS REGISTER
                                ;;**************************************************************
                                ;*TEST 10        TEST THAT 'CLEAR LOW BYTE' DOES NOT CLEAR HIGH BYTE
                                ;;**************************************************************
004666  000004                  TST10:  SCOPE
004670  012737  016000  001124          MOV     #16000,$GDDAT                   ;LOAD EXPECTED READ/WRITE RESULT
004676  013777  001124  174542          MOV     $GDDAT,@VCCSR                   ;LOAD THE REGISTER BITS
004704  105077  174536                  CLRB    @VCCSR                          ;CLEAR LOW BYTE
004710  017737  174532  001126          MOV     @VCCSR,$BDDAT                   ;READ THE RESULT OF 'CLRB'
004716  042737  140000  001126          BIC     #BIT15!BIT14,$BDDAT             ;MASK TO TOP TWO BITS
004724  023737  001124  001126          CMP     $GDDAT,$BDDAT                   ;COMPARE FOR SAME
004732  001401                          BEQ     TST11           ;;BR IF EXPECTED
004734  104003                          ERROR   3                               ;CLR LOW BYTE CHANGED THE
                                                                                ;HIGH BYTE OF THE CHAR STATUS REG.
```

```
                                ;;*****************************************************************
                                ;*TEST 11       TEST THAT 'VERT SYNC' FLAG CHANGES STATES
                                ;;*****************************************************************
004736  000004                  T11:    SCOPE
004740  005077  174502                  CLR     @VCCSR                  ;ENSURE CLEAR REGISTER
004744  004737  015672                  JSR     PC,COUNTA               ;GO WAIT FOR A TRANSITION OF THE VERT SYNC FLAG
                                        ;NOW TEST THE RELATIVE TIME DIFFERENCE BETWEEN 'OFF' AND 'ON' STATES
004750  004737  015676                  JSR     PC,COUNT
004754  010137  001510                  MOV     R1,SAVE0                ;SAVE THE FIRST TIME
004760  004737  015676                  JSR     PC,COUNT
004764  010137  001514                  MOV     R1,SAVE2                ;SAVE THE SECOND TIME
004770  013737  001514  001126          MOV     SAVE2,$BDDAT            ;LOAD THE BIGGER TIME
004776  163737  001510  001126          SUB     SAVE0,$BDDAT            ;SUB THE FIRST TIME
005004  001001                          BNE     1$              ;;BR IF NOT SAME
005006  104004                          ERROR   4                       ;NO TIME DIFFERENCE BETWEEN
005010  013737  001514  001126  1$:     MOV     SAVE2,$BDDAT            ;LOAD BIGGER TIME
005016  006237  001126                  ASR     $BDDAT                  ;DIVIDE BY 8
005022  006237  001126                  ASR     $BDDAT
005026  006237  001126                  ASR     $BDDAT
005032  023737  001126  001510          CMP     $BDDAT,SAVE0            ;TEST THAT MAGITUDE OF AT LEAST 8 BETWEEN THE TW
005040  101016                          BHI     2$              ;;BR IF GREATER
005042  013737  001510  001126          MOV     SAVE0,$BDDAT            ;LOAD FIRST VALUE
005050  006237  001126                  ASR     $BDDAT                  ;ADJUST VALUE
005054  006237  001126                  ASR     $BDDAT
005060  006237  001126                  ASR     $BDDAT
005064  023737  001126  001514          CMP     $BDDAT,SAVE2            ;COMPARE TO SECOND READ
005072  101001                          BHI     2$                      ;BR IF MAGITUDE IS GREATER
005074  104004                          ERROR   4                       ;TIME DIFFERENCE BETWEEN OFF AND ON STATE WRONG
005076  000240                  2$:     NOP
```

```
                        ;;*****************************************************************
                        ;*TEST 12       TEST THE 'CHAR READY' FLAG USING THE "HOME" CODE
                        ;;*****************************************************************
005100  000004          TST12:  SCOPE
005102  005077  174340          CLR     @VCCSR                  ;CLEAR THE CHAR REG
005106  012737  100000  001124  MOV     #BIT15,$GDDAT           ;LOAD EXPECTED READ VALUE
005114  017737  174326  001126  MOV     @VCCSR,$BDDAT           ;READ THE STATUS REGISTER
005122  100401                  BMI     1$              ;;BR IF 'CHAR READY' IS SET
005124  104003                  ERROR   3                       ;'CHAR READY' FAILED TO SET
                                                                ; BY LOADING A 'O' CHARACTER
                        ;NOW TEST THAT LOADING A "HOME" CHARACTER WILL CAUSE
                        ;"CHAR READY" TO CLEAR AND THEN SET AFTER A DELAY
005126  005037  001124  1$:     CLR     $GDDAT                  ;CLEAR EXPECTED READ VALVE
005132  112777  000035  174306  MOVB    #35,@VCCSR              ;LOAD A "HOME" CHARACTER
005140  017737  174302  001126  MOV     @VCCSR,$BDDAT           ;READ THE REGISTER
005146  100001                  BPL     2$              ;;BR IF 'CHAR READY' CLEARED
005150  104003                  ERROR   3                       ;'CHAR READY' FAILED TO CLEAR WHEN LOADED
                                                                ;WITH A "HOME" ASCII CHARACTER
005152  012737  100000  001124  2$:     MOV     #BIT15,$GDDAT
005160  012700  100000          MOV     #BIT15,R0                       ;LOAD A DELAY FACTOR
005164  017737  174256  001126  3$:     MOV     @VCCSR,$BDDAT           ;READ THE REGISTER
005172  100403                  BMI     TST13           ;;BR IF THE BIT FINALLY SET
005174  005300                  DEC     R0                      ;DELAY
005176  001372                  BNE     3$                      ;BR IF NOT DONE
005200  104003                  ERROR   3                       ;'CHAR READY' FAILED TO SET AFTER LOADING
                                                                ;A "PRINTABLE" CHARACTER.
```

```
                                        ;;*****************************************************************
                                        ;*TEST 13       TEST FOR 'VERT SYNC' INTERRUPT
                                        ;;*****************************************************************
005202  000004                  TST13:  SCOPE
005204  012746  000340                  MOV     #340,-(SP)              ;LOAD PSW RETURN TO LEVEL 7
005210  012746  005216                  MOV     #10$,-(SP)              ;LOAD RETURN P.C.
005214  000002                          RTI                             ;DO THIS TO BE LSI-11 COMPAT.
005216  012777  020000  174222  10$:    MOV     #BIT13,@VCCSR           ;SET INTR ENABLE
005224  004737  015672                  JSR     PC,COUNTA               ;WAIT FOR A COMPLETE'VERT SYNC' CYCLE
005230  012777  005274  174232          MOV     #2$,@VCINT              ;LOAD INTR RETURN VECTOR
005236  012777  000340  174226          MOV     #340,@VCINT1            ;
005244  012700  000010                  MOV     #10,R0                  ;LOAD A SHORT DELAY COUNTER
005250  005046                          CLR     -(SP)                   ;LOAD PSW RETURN TO LEVEL 0
005252  012746  005260                  MOV     #1$,-(SP)               ;LOAD RETURN P.C.
005256  000002                          RTI
005260  005300                  1$:     DEC     R0                      ;DELAY
005262  001376                          BNE     1$                      ;BR IF NOT DONE
005264  005077  174156                  CLR     @VCCSR
005270  104005                          ERROR   5                       ;'VERT SYNC' FAILED TO CAUSE AN INTR.
005272  000421                          BR      TST14           ;;BR OVER INTR HANDLER
                                        ;RETURN HERE WITH PSW=7, LOWER PRIORITY AGAIN AND TEST FOR NO INTERRUPT AGAIN
005274  022626                  2$:     CMP     (SP)+,(SP)+             ;ADJUST THE STACK BY 4
005276  012777  005332  174166          MOV     #4$,@VCINT1             ;RELOAD INTR VECTOR TO FALSE INTR HANDLER
005304  012700  000100                  MOV     #100,R0                 ;LOAD DELAY COUNTER
005310  005046                          CLR     -(SP)                   ;LOAD PSW RETURN LEVEL TO 0
005312  012746  005320                  MOV     #3$,-(SP)               ;LOAD RETURN P.C.
005316  000002                          RTI
005320  005300                  3$:     DEC     R0                      ;DELAY
005322  001376                          BNE     3$
005324  005077  174116                  CLR     @VCCSR                  ;CLEAR INTR ENABLE
005330  000402                          BR      TST14           ;;
005332  022626                  4$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
005334  104005                          ERROR   5                       ;INTR DONE FAILED TO CLEAR INTR REQUEST
```

```
1002                                    ;;*****************************************************************
1003                                    ;*TEST 14       INTERRUPT LEVEL TEST FOR 'VERT SYNC' INTERRUPT
1004                                    ;;*****************************************************************
1005  005336  000004                    TST14:  SCOPE
1006  005340  012746  000340                    MOV     #340,-(SP)              ;LOAD PSW RETURN TO LEVEL 7
1007  005344  012746  005352                    MOV     #10$,-(SP)              ;LOAD RETURN P.C.
1008  005350  000002                            RTI                             ;DO THIS TO BE LSI-11 COMPAT.
1009  005352  012777  020000  174066    10$:    MOV     #BIT13,@VCCSR           ;SET INTR ENABLE
1010  005360  004737  015672                    JSR     PC,COUNTA               ;WAIT FOR A COMPLETE 'VERT SYNC' CYCLE
1011  005364  012777  005444  174076            MOV     #2$,@VCINT              ;LOAD INTR RETURN VECTOR
1012  005372  012777  000340  174072            MOV     #340,@VCINT1            ;
1013  005400  012700  000010                    MOV     #10,R0                  ;LOAD A SHORT DELAY COUNTER
1014  005404  113737  001243  001516            MOVB    $VECT1+1,SAVE3          ;GET BR LEVEL
1015  005412  042737  177437  001516            BIC     #177437,SAVE3           ;MASK OUT BITS
1016  005420  013746  001516                    MOV     SAVE3,-(SP)             ;LOAD BR LEVEL
1017  005424  012746  005432                    MOV     #1$,-(SP)               ;LOAD RETURN P.C.
1018  005430  000002                            RTI
1019  005432  005300                    1$:     DEC     R0                      ;DELAY
1020  005434  001376                            BNE     1$                      ;BR IF NOT DONE
1021  005436  005077  174004                    CLR     @VCCSR
1022  005442  000404                            BR      5$              ;;BR OVER INTR HANDLER
1023                                    ;RETURN HERE IF AN INTERRUPT OCCURRED  **  ERROR **
1024  005444  022626                    2$:     CMP     (SP)+,(SP)+             ;ADJUST THE STACK BY 4
1025  005446  005077  173774                    CLR     @VCCSR
1026  005452  104005                            ERROR   5                       ;VERT SYNC INTERRUPTED IN ERROR
1027                                                                            ;IS BR LEVEL CORRECT ?
1028  005454  012777  005512  174006    5$:     MOV     #4$,@VCINT              ;RELOAD INTR VECTOR
1029  005462  012700  000100                    MOV     #100,R0                 ;LOAD DELAY COUNTER
1030  005466  005046                            CLR     -(SP)                   ;LOAD PSW RETURN LEVEL TO 0
1031  005470  012746  005476                    MOV     #3$,-(SP)               ;LOAD RETURN P.C.
1032  005474  000002                            RTI
1033  005476  005300                    3$:     DEC     R0                      ;DELAY
1034  005500  001376                            BNE     3$
1035  005502  005077  173740                    CLR     @VCCSR                  ;CLEAR INTR ENABLE
1036  005506  104005                            ERROR   5                       ;LOWERING PRIORITY FAILED TO CAUSE AN INTERRUPT
1037  005510  000401                            BR      6$              ;;
1038  005512  022626                    4$:     CMP     (SP)+,(SP)+             ;ADJUST STACK
1039  005514  005046                    6$:     CLR     -(SP)
1040  005516  012746  005524                    MOV     #7$,-(SP)
1041  005522  000002                            RTI
1042  005524  000240                    7$:     NOP
```

```
1043                                    ;;*****************************************************************
1044                                    ;*TEST 15       DYNAMIC TESTING OF THE X CROSS HAIR POSITION REGISTERS
1045                                    ;;*****************************************************************
1046  005526  000004                    TST15:  SCOPE
1047  005530  005737  001544                    TST     BURN                    ;TEST IF RUNNING IN "BURN-IN" MODE
1048  005534  001406                            BEQ     3$                      ;BR IF NOT
1049  005536  013777  001546  174004            MOV     COAXSW,@COAX            ;LOAD THE RELAY
1050  005544  004537  016666                    JSR     R5,DELAY                ;LET THE RELAY SETTLE
1051  005550  000012                            10.
1052  005552                            3$:
1053  005552  005737  001200                    TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
1054  005556  001427                            BEQ     TST16           ;;BR IF FIRST PASS
1055  005560  032777  002000  173352            BIT     #SW10,@SWR              ;TEST IF INHIBIT VISUAL CHAR. PATTERNS
1056  005566  001023                            BNE     TST16           ;;BR IF INHIBIT IS SET
1057  005570  012777  010000  173650  2$:       MOV     #BIT12,@VCCSR           ;ENABLE X CROSS HAIRS
1058  005576  005037  001124                    CLR     $GDDAT                  ;CLEAR EXPECTED VALUE
1059  005602  005077  173642                    CLR     @VCHRLO                 ;CLEAR POS.
1060  005606  113777  001124  173634  1$:       MOVB    $GDDAT,@VCHRLO          ;LOAD THE LOW BYTE (X POSITION)
1061  005614  004537  016666                    JSR     R5,DELAY
1062  005620  000001                            1
1063  005622  005237  001124                    INC     $GDDAT                  ;UPDATE X POSITION
1064  005626  023737  001124  001536            CMP     $GDDAT,XHAIR            ;TEST IF MORE LINES?
1065  005634  001364                            BNE     1$
1066                                    ;;*****************************************************************
1067                                    ;*TEST 16       DYNAMIC TESTING OF THE Y CROSS HAIR POSITION REGISTERS
1068                                    ;;*****************************************************************
1069  005636  000004                    TST16:  SCOPE
1070  005640  005737  001200                    TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
1071  005644  001431                            BEQ     TST17           ;;BR IF FIRST PASS
1072  005646  032777  002000  173264            BIT     #SW10,@SWR              ;TEST IF INHIBIT VISUAL CHAR. PATTERNS
1073  005654  001025                            BNE     TST17           ;;BR IF INHIBIT IS SET
1074  005656  012777  004000  173562            MOV     #BIT11,@VCCSR           ;CNABLE Y CROSS HAIRS
1075  005664  005037  001124                    CLR     $GDDAT
1076  005670  005077  173554                    CLR     @VCHRLO                 ;CLEAR POSITION
1077  005674  113777  001124  173550  1$:       MOVB    $GDDAT,@VCHRHI
1078  005702  004537  016666                    JSR     R5,DELAY
1079  005706  000001                            1
1080  005710  105237  001124                    INCB    $GDDAT
1081  005714  123737  001540  001124            CMPB    YHAIR,$GDDAT
1082  005722  001364                            BNE     1$
1083  005724  005077  173516                    CLR     @VCCSR                  ;DISABLE CROSS HAIRS
1084
```

```
1085                                    ;;*****************************************************************
1086                                    ;*TEST 17       FULL SCREEN OF A CHARACTER
1087                                    ;;*****************************************************************
1088  005730  000004                    TST17:  SCOPE
1089  005732  005737  001200                    TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
1090  005736  001413                            BEQ     TST20           ;;BR IF FIRST PASS
1091  005740  032777  002000  173172            BIT     #SW10,@SWR              ;TEST IF INHIBIT VISUAL CHAR. PATTERNS
1092  005746  001007                            BNE     TST20           ;;BR IF INHIBIT IS SET
1093  005750  004737  017256                    JSR     PC,HOME                 ;HOME AND ERASE THE SCREEN
1094  005754  004737  016076                    JSR     PC,FILLWC               ;FILL SCREEN WITH A 'E'S
1095  005760  004537  016666                    JSR     R5,DELAY
1096  005764  000144                            100.
1097
1098                                    ;;*****************************************************************
1099                                    ;*TEST 20       SINGLE CHARACTER ACROSS ALL COLS. <ALL CHARACTERS>
1100                                    ;;*****************************************************************
1101  005766  000004                    TST20:  SCOPE
1102  005770  005737  001200                    TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
1103  005774  001446                            BEQ     TST21           ;;BR IF FIRST PASS
1104  005776  032777  002000  173134            BIT     #SW10,@SWR              ;TEST IF INHIBIT VISUAL CHAR. PATTERNS
1105  006004  001042                            BNE     TST21           ;;BR IF INHIBIT IS SET
1106  006006  004737  017256                    JSR     PC,HOME                 ;HOME AND ERASE THE SCREEN
1107  006012  012737  000040  001530    3$:     MOV     #40,STCHAR              ;SET-UP STARTING CHARACTER
1108
1109  006020  013737  001524  001534            MOV     VHO,TEMPO               ;LOAD COUNT
1110  006026  013701  001530            1$:     MOV     STCHAR,R1               ;LOAD R1= TO CHARACTER
1111  006032  004737  016132                    JSR     PC,FILBUF               ;LOAD A BUFFER WITH THAT CHARACTER
1112
1113  006036  004737  016032                    JSR     PC,XPRNT                ;DISPLAY A FULL LINE FROM THE BUFFER
1114
1115  006042  005337  001534                    DEC     TEMPO                   ;DONE ?
1116  006046  100010                            BPL     2$                      ;FINISHED
1117  006050  004537  016666                    JSR     R5,DELAY
1118  006054  000144                            100.
1119  006056  004737  017256                    JSR     PC,HOME                 ;HOME AND ERASE THE SCREEN
1120  006062  013737  001524  001534            MOV     VHO,TEMPO
1121  006070  005237  001530            2$:     INC     STCHAR                  ;UPDATE THE CHARACTER
1122  006074  023737  001532  001530            CMP     LASTCH,STCHAR           ;TEST FOR FINAL CHARACTER
1123  006102  001351                            BNE     1$                      ;BRANCH IF NOT COMPLETED
1124
1125  006104  004537  016666                    JSR     R5,DELAY
1126  006110  000144                            100.
1127
```

```
                                ;;**********************************************************************
                                ;*TEST 21      ROTATING CHARACTERS ACROSS ALL COLS. (ALL CHARACTERS)
                                ;;**********************************************************************
006112  000004                *ST21:  SCOPE
006114  005737  001200                TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
006120  001451                        BEQ     TST22           ;;BR IF FIRST PASS
006122  032777  002000  173010        BIT     #SW10,@SWR              ;TEST IF INHIBIT VISUAL CHAR. PATTERNS
006130  001045                        BNE     TST22           ;;BR IF INHIBIT IS SET
006132  004737  017256                JSR     PC,HOME                 ;HOME AND ERASE THE SCREEN
006136  012737  000040  001530  3$:   MOV     #40,STCHAR              ;SET-UP STARTING CHARACTER

006144  004737  017256                JSR     PC,HOME                 ;HOME AND ERASE THE SCREEN
006150  013737  001524  001534        MOV     VHO,TEMPO               ;LOAD TEMP
006156  013701  001530          1$:   MOV     STCHAR,R1               ;LOAD R1=TO CHARACTER
006162  004537  016166                JSR     R5,LIC                  ;LOAD A BUFFER STARTING WITH
006166  001522                        WIDTH                           ; THAT CHARACTER AND WIDTH (BYTE)

006170  004737  016032                JSR     PC,XPRNT                ;DISPLAY A FULL LINE FROM THE BUFFER

006174  005337  001534                DEC     TEMPO                   ;DONE ?
006200  100010                        BPL     2$                      ;BR IF YES
006202  004537  016666                JSR     R5,DELAY
006206  000144                        100.
006210  004737  017256                JSR     PC,HOME                 ;HOME AND ERASE THE SCREEN
006214  013737  001524  001534        MOV     VHO,TEMPO
006222  005237  001530          2$:   INC     STCHAR                  ;UPDATE THE STARTING CHARACTER
006226  023737  001532  001530        CMP     LASTCH,STCHAR           ;TEST FOR FINAL CHARACTER
006234  001350                        BNE     1$                      ;BRANCH IF NOT COMPLETED

006236  004537  016666                JSR     R5,DELAY
006242  000144                        100.
```

```
                                ;;**********************************************************************
                                ;*TEST 22       DIRECT CURSOR ADDRESS TEST
                                ;;**********************************************************************
006244  000004                  TST22:  SCOPE
006246  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
006252  001530                          BEQ     TST23           ;;BR IF FIRST PASS
006254  032777  002000  172656          BIT     #SW10,@SWR              ;TEST IF INHIBIT VISUAL CHAR. PATTERNS
006262  001124                          BNE     TST23           ;;BR IF INHIBIT IS SET
006264  004737  017256                  JSR     PC,HOME                 ;HOME AND ERASE THE SCREEN
006270  012737  123456  017704  DCATST: MOV     #123456,$LONUM  ;PRIME RANDOM NUMBER GENERATOR
006276  012737  176543  017702          MOV     #176543,$HINUM
006304  013737  001520  017152          MOV     TOTALC,OVRAL    ;LOAD CHAR COUNT
006312  013737  001524  017150          MOV     VHO,SET         ;LOAD LINE COUNT
006320  012700  024350                  MOV     #BUFFER+20,R0   ;LOAD DESTINATION BUFFER
006324  012701  017154          2$:     MOV     #MSGTXT,R1      ;LOAD MESSAGE POINTER
006330  012120                  1$:     MOV     (R1)+,(R0)+     ;LOAD 2 CHARACTERS
006332  022701  017254                  CMP     #MSGTND,R1      ;TEST FOR LAST CHAR
006336  001374                          BNE     1$              ;BR UNTIL DONE 1 LINE
006340  005337  017150                  DEC     SET             ;FINISHED ALL LINES
006344  100367                          BPL     2$              ;BR IF NOT
006346  012737  177777  024330          MOV     #-1,BUFFER      ;LOAD MESSAGE TERMINATOR
006354  004737  017604          GENER:  JSR     PC,$RAND        ;GENERATE RANDOM NUMBER
006360  013700  017702                  MOV     $HINUM,R0       ;GET RANDOM NUMBER
006364  042700  177740                  BIC     #177740,R0      ;RANDOM NO. MUST BE TWO DIGITS
006370  020037  001524                  CMP     R0,VHO          ;NO. MUST NOT BE GREATER THAN  VHO
006374  101367                          BHI     GENER           ;GREATER, REGENERATION
006376  010037  017144                  MOV     R0,YADDS        ;STORE RANDOM Y COORDINATE
006402  012737  024350  017150          MOV     #BUFFER+20,SET  ;LOAD BASE POINTER
006410  010001                          MOV     R0,R1
006412  001405                          BEQ     GENRX           ;RESULT, MINIMUM Y COORDINATE
006414  063737  001522  017150  1$:     ADD     WIDTH,SET       ;SETUP  Y INDEX LOCATION FOR PRINTOUT
006422  005301                          DEC     R1
006424  001373                          BNE     1$              ;Y COORDINATE IS SET
006426  004737  017604          GENRX:  JSR     PC,$RAND        ;GENERATE RANDOM NUMBER
006432  013700  017702                  MOV     $HINUM,R0       ;GET A RANDOM NUMBER
006436  042700  177700                  BIC     #177700,R0      ;RANDOM NO. MAY BE LESS THAN 200
006442  010037  017146                  MOV     R0,XADDS        ;STORE RANDOM X COORDINATE
006446  060037  017150                  ADD     R0,SET          ;SETUP X COOR, FOR PNTOUT.
006452  013701  017150                  MOV     SET,R1          ;SETUP CHECK
006456  105711                          TSTB    (R1)            ;HAS CURRENT CHAR, ALREADY BEEN USED?
006460  100735                          BMI     GENER           ;YES, REGENERATE
006462  113777  017144  172766          MOVB    YADDS,@VCPOSH   ;LOAD Y COORDINATE
006470  113777  017146  172756          MOVB    XADDS,@VCPOS    ;LOAD X COORDINATE
006476  111137  024330                  MOVB    (R1),BUFFER     ;LOAD CHARACTER TO BE PRINTED
006502  152711  000200                  BISB    #200,(R1)       ;INDICATE USE OF CURSOR POSITION
006506  004737  016032                  JSR     PC,XPRNT        ;EXECUTE AND PRINT CHARACTER
006512  104407                          CKSWR                   ;TEST FOR CTRL G
006514  005337  017152                  DEC     OVRAL           ;MAXIMUM NO. OF COORDINATES
006520  001315                          BNE     GENER           ;BR BACK UNTIL DONE
006522  004537  016666                  JSR     R5,DELAY
006526  000144                          100.
006530  004737  017256                  JSR     PC,HOME                 ;HOME AND ERASE THE SCREEN
```

```
                                        ;;*****************************************************************
                                        ;*TEST 23       CURSOR MOTION TEST
                                        ;;*****************************************************************
006534  000004                  TST23:  SCOPE
006536  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
006542  001562                          BEQ     TST24           ;;BR IF FIRST PASS
006544  032777  002000  172366          BIT     #SW10,@SWR              ;TEST IF INHIBIT VISUAL CHAR. PATTERNS
006552  001156                          BNE     TST24           ;;BR IF INHIBIT IS SET
006554  004737  017256                  JSR     PC,HOME                 ;HOME AND ERASE THE SCREEN
006560  042777  002000  172660          BIC     #BIT10,@VCCSR           ;ENABLE CURSOR

                                        ;ROUTINE TO LOAD REFERENCE LINES FOR CURSOR MOTION TEST

006566  012700  024330          LBMT:   MOV     #BUFFER,R0      ;LOAD POINTER
006572  013701  001526                  MOV     VH1,R1          ;LOAD R1
006576  005301                          DEC     R1
006600  004737  006676                  JSR     PC,MOVDN1       ;MOVE CURSOR DOWN
006604  004737  006714                  JSR     PC,MOVRIG       ;MOVE RIGHT
006610  112720  000062                  MOVB    #62,(R0)+       ;LOAD #2
006614  004737  006714                  JSR     PC,MOVRIG       ;MOVE RIGHT
006620  004737  006714                  JSR     PC,MOVRIG       ;MOVE RIGHT
006624  112720  000040                  MOVB    #40,(R0)+
006630  112720  000061                  MOVB    #61,(R0)+       ;LOAD #1
006634  004737  006672                  JSR     PC,MOVDWN       ;MOVE DOWN
006640  004737  006714                  JSR     PC,MOVRIG       ;MOVE RIGHT
006644  112720  000063                  MOVB    #63,(R0)+       ;LOAD #3
006650  004737  006714                  JSR     PC,MOVRIG
006654  004737  006714                  JSR     PC,MOVRIG       ;MOVE RIGHT
006660  112720  000064                  MOVB    #64,(R0)+       ;LOAD #4
006664  112710  000377                  MOVB    #377,(R0)       ;TERM
006670  000420                          BR      LBMT1           ;;BR TO NEXT PART

006672  013701  001526          MOVDWN: MOV     VH1,R1
006676  112720  000015          MOVDN1: MOVB    #15,(R0)+       ;LOAD CR
006702  112720  000012                  MOVB    #12,(R0)+       ;LOAD LF
006706  005301                          DEC     R1
006710  001372                          BNE     MOVDN1          ;LOOP UNTIL DONE
006712  000207                          RTS     PC              ;EXIT

006714  012701  000016          MOVRIG: MOV     #16,R1          ;LOAD R1
006720  112720  000040          1$:     MOVB    #40,(R0)+       ;LOAD SPACES
006724  005301                          DEC     R1
006726  100374                          BPL     1$              ;LOOP UNTIL DONE
006730  000207                          RTS     PC              ;EXIT

006732  004737  016032          LBMT1:  JSR     PC,XPRNT        ;DISPLAY THIS LINE
006736  004537  016666                  JSR     R5,DELAY
006742  000144                          100.
```

```
                                        ;CURSOR MOTION SUBROUTINE
001526                          CUM:    MOV     VH1,R1          ;LOAD COUNT
024330                                  MOV     #BUFFER,R0      ;LOAD BUFFER POINTER
000032                          1$:     MOVB    #32.,(R0)+      ;LOAD A CURSOR UP
                                        DEC     R1
                                        BNE     1$              ;LOOP UNTIL DONE
000130                                  MOVB    #130,(R0)+      ;LOAD 'X'
000040                                  MOV     #40,R1          ;LOAD COUNT
000010                          2$:     MOVB    #10,(R0)+       ;LOAD ^BACKSPACE (CURSOR LEFT)
                                        DEC     R1              ;DONE ALL ?
                                        BPL     2$              ;BR IF NOT
000130                                  MOVB    #130,(R0)+      ;LOAD 'X'
000010                                  MOVB    #10,(R0)+       ;LOAD BACKSPACE
001526                                  MOV     VH1,R1          ;LOAD COUNT
000013                          3$:     MOVB    #13,(R0)+       ;LOAD CURSOR DOWN
                                        DEC     R1              ;DONE ?
                                        BNE     3$              ;BR IF NOT
000130                                  MOVB    #130,(R0)+      ;LOAD 'X'
000010                                  MOVB    #10,(R0)+       ;LOAD BACKSPACE
000036                                  MOV     #36,R1          ;LOAD COUNT
000030                          4$:     MOVB    #30,(R0)+       ;LOAD 'C' CURSOR RIGHT
                                        DEC     R1
                                        BPL     4$              ;LOOP UNTIL DONE
000130                                  MOVB    #130,(R0)+      ;LOAD 'X'
000377                                  MOVB    #377,(R0)+
016632                                  JSR     PC,XPRNT        ;DISPLAY THIS LINE
016666                                  JSR     R5,DELAY
                                        100.
                                        JSR     PC,HOME                 ;HOME AND ERASE THE SCREEN
002000  172336                          BIS     #$ITID,@VCCSR           ;DISABLE CHAR. CURSOR


                                ;;*****************************************************************
                                ;*TEST 24
                                ;;*****************************************************************
                                *T24:   SCOPE
```

```
                                NOCHAR: NOP
007112  000240                          MOV     $GDADR,TEMP1            ;GET BASE ADDRESS
007114  013737  001120  001502          ADD     #20,TEMP1               ;UPDATE TO THE MAP ADDRESS
007122  062737  000020  001502          MOV     $CDWI,DEVTMP            ;GET THE NUMBER OF MAPS SELECTED
007130  013737  001252  004360          BIC     #177700,DEVTMP          ;MASK TO LOWER SIX BITS
007136  042737  177700  004360          BEQ     MAPRTA                  ;BR IF NO MAPS SELECTED
007144  001506                          CLR     $UNIT                   ;CLEAR MAP INDICATOR
007146  005037  001204                  MOV     #BIT0,TEMP3             ;LOAD TESTING BIT
007152  012737  000001  001506          BIT     TEMP3,DEVTMP            ;TEST IF THIS BIT MAP IS SELECTED
007160  033737  001506  004360  MAPRTB: BEQ     MAPRET                  ;BR IF NOT
007166  001402                          JMP     NUMBR                   ;SELECTED - TEST THIS MAP
007170  000137  007370                  ADD     #20,TEMP1               ;UPDATE TO NEXT BUS ADDRESS
007174  062737  000020  001502  MAPRET: INC     $UNIT                   ;UPDATE MAP INDICATOR
007202  005237  001204                  ASL     TEMP3                   ;MOVE THE TEST BIT
007206  006337  001506                  CMP     #100,TEMP3              ;TEST IF ANY MORE POSSIBLE MAPS
007212  022737  000100  001506          BNE     MAPRTB                  ;BR IF MORE
007220  001357                          .SBTTL  MULTIPLE MAP PATTERN
                                        TST     $PASS                   ;TEST IF 1ST PASS
007222  005737  001200                  BEQ     MAPRTA                  ;BR IF FIRST PASS
007226  001455                          JSR     PC,LODADR               ;LOAD THE BUS ADDRESS
007230  004737  003576                  MOV     $CDWI,10$               ;GET THE NUMBER OF MAPS
007234  013737  001252  007342          BIC     #177700,10$             ;MASK
007242  042737  177700  007342          MOV     #DUALPL,R0              ;LOAD POINTER
007250  012700  007346                  TST     10$                     ;TEST IF ANY EXIST
007254  005737  007342          1$:     BEQ     4$                      ;BR IF DONE
007260  001424                          ASR     10$                     ;TEST FOR NEXT MAP
007262  006237  007342                  BCC     2$                      ;BR IF NEXT NOT SELECTED
007266  103015                          JSR     PC,CLRMAP               ;CLEAR THE MAP
007270  004737  016234                  MOVB    (R0)+,11$               ;GET COLOR VALUE
007274  112037  007344                  MOV     11$,@VGCOLR             ;LOAD THE COLOR FOR THIS MAP
007300  013777  007344  172160          MOVB    (R0)+,@VGCSR            ;LOAD THE ORGIN FOR THIS MAP
007306  112077  172146                  BIS     #BIT8,@VGCSR            ;ENABLE MAP
007312  052777  000400  172140          BR      3$
007320  000401                  2$:     TST     (R0)+                   ;ADJUST POINTER
007322  005720                  3$:     JSR     PC,MOVADR               ;LOAD ADR OF NEXT MAP
007324  004737  003730                  BR      1$
007330  000751                  4$:     JSR     R5,DELAY
007332  004537  016666                  100.
007336  000144                          BR      MAPRTA
007340  000410                  10$:    0
007342  000000                  11$:    0
007344  000000
                                DUALPL: .BYTE   3,5                     ;MAP #0 COLOR AND ORGIN
007346      003     005                 .BYTE   14,6                    ;MAP #1 COLOR AND ORGIN
007350      014     006                 .BYTE   60,1                    ;MAP #2
007352      060     001                 .BYTE   33,2                    ;MAP #3
007354      033     002                 .BYTE   17,15                   ;MAP #4
007356      017     015                 .BYTE   63,16                   ;MAP #5
007360      063     016

007362  000177  000000          MAPRTA: JMP     @RETURN

007366  000000                  RETURN: 0                               ;RETURN CONTROL
```

```
                                ;THE BIT MAP ADDRESS IS IN "TEMP1" - LOAD THE ADDRESS AND
                                ;       START TESTING THAT BIT MAP

001460  012737  001502  001460  NUMBR:  MOV     TEMP1,@VGCSR            ;LOAD BUS ADDRESSES
001462  012737  001502  001462          MOV     TEMP1,@VGPC
001464  012737  001502  001464          MOV     TEMP1,@VGBUF
001466  012737  001502  001466          MOV     TEMP1,@VGCOLR
001462  062737  000002  001462          ADD     #2,VGPC
001464  062737  000004  001464          ADD     #4,VGBUF
001466  062737  000010  001466          ADD     #10,VGCOLR
001102  012737  000024  001102          MOV     #STN-1,$TSTNM           ;LOAD TEST NUMBER

                                ;;*****************************************************************
                                ;*TEST 25       TEST THAT 'EXPAND' BIT CAN BE SET
                                ;;*****************************************************************
007744                          ST25:   SCOPE
001124  012737  004000  001124          MOV     #BIT11,$GDDAT           ;LOAD THE EXPECTED
001126  013777  001124  171662          MOV     $GDDAT,@VGCSR           ;LOAD THE REGISTER
001130  017737  171656  001126          MOV     @VGCSR,$BDDAT           ;READ THE REGISTER
001134  042737  171360  001126          BIC     #171360,$BDDAT          ;MASK TO BITS
001140  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE EXPECTED TO RECVD
001144  001401                          BEQ     TST26           ;;BR IF SET
007516  104006                          ERROR   6                       ;'EXPAND' BIT FAILED TO SET

                                ;;*****************************************************************
                                ;*TEST 26       TEST THAT 'MONO' BIT CAN BE SET
                                ;;*****************************************************************
007514  000004                  ST26:   SCOPE
007516  012737  002000  001124          MOV     #BIT10,$GDDAT           ;LOAD THE EXPECTED
007524  013777  001124  171726          MOV     $GDDAT,@VGCSR           ;LOAD THE REGISTER
007532  017737  171722  001126          MOV     @VGCSR,$BDDAT           ;READ THE REGISTER
007540  042737  171360  001126          BIC     #171360,$BDDAT          ;MASK TO BITS
007546  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE EXPECTED TO RECVD
007554  001401                          BEQ     TST27           ;;BR IF SET
007556  104006                          ERROR   6                       ;'MONO' BIT FAILED TO SET

                                ;;*****************************************************************
                                ;*TEST 27       TEST THAT 'MAP ENABLE' BIT CAN BE SET
                                ;;*****************************************************************
007560  000004                  TST27:  SCOPE
007562  012737  000400  001124          MOV     #BIT8,$GDDAT            ;LOAD THE EXPECTED
007570  013777  001124  171662          MOV     $GDDAT,@VGCSR           ;LOAD THE REGISTER
007576  017737  171656  001126          MOV     @VGCSR,$BDDAT           ;READ THE REGISTER
007604  042737  171360  001126          BIC     #171360,$BDDAT          ;MASK TO BITS
007612  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE EXPECTED TO RECVD
007620  001401                          BEQ     TST30           ;;BR IF SET
007622  104006                          ERROR   6                       ;MAP ENABLE FAILED TO SET
```

```
                                        ;;*****************************************************************
                                        ;*TEST 30       TEST THAT 'ORGIN POINT' BITS CAN BE SET
                                        ;;*****************************************************************
007624  000004                  TST30:  SCOPE
007626  012737  007642  001110          MOV     #1$,$LPERR
007634  012737  000001  001124          MOV     #BIT0,$GDDAT            ;LOAD THE EXPECTED
007642  013777  001124  171610  1$:     MOV     $GDDAT,@VGCSR           ;LOAD THE REGISTER
007650  017737  171604  001126          MOV     @VGCSR,$BDDAT           ;READ THE REGISTER
007656  042737  171360  001126          BIC     #171360,$BDDAT          ;MASK TO BITS
007664  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE THE EXPECTED TO RECVD
007672  001401                          BEQ     2$              ;;BR IF SET
007674  104006                          ERROR   6                       ;'ORGIN POINT' BIT FAILED TO SET
007676  005237  001124          2$:     INC     $GDDAT                  ;UPDATE EXPECTED VALUE OF 'ORGIN POINT'
007702  023737  000020  001124          CMP     #20,$GDDAT              ;TEST IF VALID VALUE FOR 'ORGIN'
007710  001354                          BNE     1$                      ;BR IF MORE TO TEST
                                        ;;*****************************************************************
                                        ;*TEST 31       TEST THAT CLEAR LOW BYTE DOES NOT CLEAR HIGH BYTE
                                        ;;*****************************************************************
007712  000004                  TST31:  SCOPE
007714  012737  006400  001124          MOV     #BIT11!BIT10!BIT8,$GDDAT
007722  012777  006417  171530          MOV     #BIT11!BIT10!BIT8+17,@VGCSR
007730  105077  171524                  CLRB    @VGCSR                  ;CLEAR LOW BYTE
007734  017737  171520  001126          MOV     @VGCSR,$BDDAT           ;READ THE REGISTER
007742  042737  170360  001126          BIC     #170360,$BDDAT          ;MASK TO BITS
007750  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST IF SAME
007756  001401                          BEQ     TST32           ;;BR IF SAME
007760  104006                          ERROR   6                       ;CLEARING LOW BYTE OF STATUS CHANGED THE HIGH BY
                                        ;;*****************************************************************
                                        ;*TEST 32       TEST THAT CLEAR HIGH BYTE DOES NOT CLEAR LOW BYTE
                                        ;;*****************************************************************
007762  000004                  TST32:  SCOPE
007764  012737  000017  001124          MOV     #17,$GDDAT              ;LOAD EXPECTED READ VALUE
007772  012777  006417  171460          MOV     #BIT11!BIT10!BIT8+17,@VGCSR             ;LOAD STATUS REGISTER
010000  013700  001460                  MOV     VGCSR,R0                ;MAKE ADDRESS
010004  005200                          INC     R0                      ;OF HIGH BYTE
010006  105010                          CLRB    (R0)                    ;CLEAR HIGH BYTE
010010  017737  171444  001126          MOV     @VGCSR,$BDDAT           ;READ REGISTER
010016  042737  170360  001126          BIC     #170360,$BDDAT          ;MASK OUT BITS
010024  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;COMPARE EXPECT TO RCVD
010032  001401                          BEQ     TST33           ;;BR IF SAME
010034  104006                          ERROR   6                       ;CLEARING THE HIGH BYTE OF STATUS CHANGED
                                                                        ;THE LOW BYTE
                                        ;;*****************************************************************
                                        ;*TEST 33       TEST THAT "RESET" CLEARS THE BIT MAP STATUS REGISTER
                                        ;;*****************************************************************
010036  000004                  TST33:  SCOPE
010040  005037  001124                  CLR     $GDDAT                  ;LOAD EXPECTED
010044  012777  006417  171406          MOV     #BIT11!BIT10!BIT8!17,@VGCSR     ;LOAD THE STATUS REGISTER
010052  000005                          RESET                           ;RESET THE WORLD
010054  017737  171400  001126          MOV     @VGCSR,$BDDAT           ;READ THE REGISTER
010062  042737  171360  001126          BIC     #171360,$BDDAT          ;MASK TO OTHER BITS
010070  001401                          BEQ     TST34           ;;BR IF CLEARED
010072  104006                          ERROR   6                       ;RESET FAILED TO CLEAR BIT MAP STATUS REGISTER
```

```
                                        ;;*****************************************************************
                                        ;*TEST 34        TEST MAP READY AND CLEAR MAP BITS
                                        ;;*****************************************************************
010074  000034                          *ST34:  SCOPE
                                        ;TEST THAT WRITTING BIT 9 (CLEAR MAP) CLEARS MAP READY (BIT 7)
010076  005037  001124                          CLR     $GDDAT                  ;CLEAR EXPECTED
010102  005037  001126                          CLR     $BDDAT                  ;CLEAR RESULTS
010106  004537  016666                          JSR     R5,DELAY                ;WAIT FOR VERT SYNC
010112  100001                                  BIT15!1
010114  052777  001000  171336                  BIS     #BIT9,@VGCSR            ;SET "CLEAR MAP" BIT
010122  117737  171332  001126                  MOVB    @VGCSR,$BDDAT           ;READ VG STATUS REG
010130  100001                                  BPL     1$              ;;BR IF CLEARED
010132  104014                                  ERROR   14                      ;BIT MAP READY NOT SET

                                        ;TEST "CLEAR MAP" BIT SETS

010134  012737  001000  001124          1$:     MOV     #BIT9,$GDDAT            ;LOAD EXPECTED
010142  004537  016665                          JSR     R5,DELAY                ;WAIT FOR VERT SYNC
010146  100001                                  BIT15!BIT0
010150  017737  171334  001126                  MOV     @VGCSR,$BDDAT           ;READ MAP STATUS REG.
010156  033737  001124  001126                  BIT     $GDDAT,$BDDAT           ;TEST IF BIT IS SET
010164  001001                                  BNE     2$              ;;BR IF SET
010166  104014                                  ERROR   14                      ;CLEAR MAP BIT FAILED TO SET

                                        ;TEST THAT "MAP READY" CLEARED

010170  005037  001124          2$:     CLR     $GDDAT                  ;CLEARED EXPECTED
010174  105737  001126                  TSTB    $BDDAT                  ;TEST BIT 7
010200  100001                                  BPL     3$              ;;BR IF CLEARED
010202  104014                                  ERROR   14                      ;SETTING "CLEAR MAP" BIT FAILED TO CLEAR "MAP RE

                                        ;TEST THAT MAP READY RETURNS AFTER 2 VERT SYNC TIMES

010204  004537  016666          3$:     JSR     R5,DELAY                        ;WAIT FOR VERT SYNC
010210  100002                                  BIT15!2
010212  012737  000200  001124                  MOV     #BIT7,$GDDAT            ;LOAD EXPECTED
010220  017737  171234  001126                  MOV     @VGCSR,$BDDAT           ;READ STATUS
010226  033737  001124  001126                  BIT     $GDDAT,$BDDAT           ;TEST IF BIT IS SET
010234  001001                                  BNE     4$              ;;BR IS SET
010236  104014                                  ERROR   14                      ;MAP READY FAILED TO SET AFTER CLEAR MAP

                                        ;TEST THAT "CLEAR MAP" BIT WAS CLEARED

010240  005037  001124          4$:     CLR     $GDDAT                  ;CLEAR EXPECTED
010244  032737  001000  001126                  BIT     #BIT9,$BDDAT            ;TEST FOR CLEARED BIT
010252  001401                                  BEQ     TST35           ;;BR IF CLEARED
010254  104014                                  ERROR   14                      ;MAP READY SET BUT "CLEAR MAP" BIT FAILED TO CLE
```

```
                                        ;;*****************************************************************
                                        ;*TEST 35        READ-WRITE TEST OF THE LOW 3 BITS OF THE MAP P.C.
                                        ;;*****************************************************************
010256  000004                  ↑ST35:  SCOPE
010260  012737  010276  001110          MOV     #1$,$LPERR
010266  005037  001124                  CLR     $GDDAT                  ;CLEAR EXPECTED READ VALUE
010272  005037  001474                  CLR     $TEMP                   ;CLEAR WRITE VALUE
010276  013777  001474  171156  1$:     MOV     $TEMP,@VGPC             ;LOAD BIT MAP P.C.
010304  017737  171150  001126          MOV     @VGCSR,$BDDAT           ;READ STATUS (BIT MAP)
010312  042737  177617  001126          BIC     #177617,$BDDAT          ;MASK TO UNUSED BITS
010320  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST IF SAME
010326  001401                          BEQ     2$              ::BR IF SAME
010330  104007                          ERROR   7                       ;BIT MAP P.C. DID NOT LOAD
                                                                        ;CORRECTLY
010332  062737  000020  001124  2$:     ADD     #20,$GDDAT              ;UPDATE EXPECTED READ VALUE
010340  005237  001474                  INC     $TEMP                   ;UPDATE EXPECTED WRITE VALUE
010344  022737  000010  001474          CMP     #10,$TEMP               ;TEST FOR FIRST NON-VALID
010352  001351                          BNE     1$                      ;BRANCH IF STILL OK VALUES.

                                        ;;*****************************************************************
                                        ;*TEST 36        STATIC BIT MAP P.C. INCREMENT
                                        ;;*****************************************************************
010354  000004                  ↑ST36:  SCOPE
010356  012737  010376  001110          MOV     #1$,$LPERR
010364  012737  000020  001124          MOV     #20,$GDDAT              ;LOAD EXPECTED READ VALUE
010372  005077  171064                  CLR     @VGPC                   ;CLEAR MAP P.C.
010376  012777  177777  171060  1$:     MOV     #-1,@VGBUF              ;LOAD A PIXEL
010404  012700  000100                  MOV     #BIT6,R0                ;LOAD DELAY COUNT
010410  105777  171044          3$:     TSTB    @VGCSR                  ;TEST IF MAP IS READY
010414  100403                          BMI     4$                      ;BR IF YES
010416  005300                          DEC     R0                      ;DELAY
010420  001373                          BNE     3$                      ;BR IN NOT EXCEEDED
010422  104014                          ERROR   14                      ;MAP READY FAILED TO SET
010424  017737  171030  001126  4$:     MOV     @VGCSR,$BDDAT           ;READ RESULT VALUE FROM BITS 4-6 OF CSR
010432  042737  177617  001126          BIC     #177617,$BDDAT          ;MASK TO UNUSED BITS
010440  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST IF SAME
010446  001401                          BEQ     2$              ::BR IF SAME
010450  104010                          ERROR   10                      ;BIT MAP P.C. FAILED TO INCREMENT
                                                                        ;BY LOADING BUFFER WITHOUT LOADING P.C.
010452  062737  000020  001124  2$:     ADD     #BIT4,$GDDAT            ;UPDATE EXPECTED READ VALUE
010460  022737  000200  001124          CMP     #BIT7,$GDDAT            ;TEST IF MORE BITS TO INCREMENT
010466  001343                          BNE     1$              ::BR IF MORE BITS
```

```
                                        ;;*****************************************************************
                                        ;*TEST 37       READ-WRITE WORD TEST OF PIXEL 0,1,2 AND 3 -- BIT 0
                                        ;;*****************************************************************
010470  000004                  T$T37:  SCOPE
010472  012737  010514  001110          MOV     #1$,$LPERR
010500  012737  010000  001124          MOV     #BIT12,$GDDAT           ;LOAD EXPECTED READ VALUE
010506  012737  000001  001474          MOV     #BIT0,$TEMP             ;LOAD EXPECTED WRITE VALUE
010514  005077  170742          1$:     CLR     @VGPC                   ;RESET P.C.
010520  005077  170740                  CLR     @VGBUF                  ;CLEAR DATA
010524  005077  170732                  CLR     @VGPC                   ;RESET PC
010530  105777  170724          3$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
010534  100375                          BPL     3$
010536  013777  001474  170720          MOV     $TEMP,@VGBUF            ;LOAD WRITE VLUE
010544  105777  170710          4$:     TSTB    @VGCSR                  ;WAIT FOR READY
010550  100375                          BPL     4$
010552  017737  170702  001126          MOV     @VGCSR,$BDDAT           ;READ RESULT VALUE
010560  042737  007777  001126          BIC     #7777,$BDDAT            ;MASK TO UNWANTED BITS
010566  023737  001124  001126          CMP     $GDDAT,$BDDAT           ;TEST IF SAME
010574  001401                          BEQ     2$              ;;BR IF SAME
010576  104011                          ERROR   11                      ;BIT0 OF A PIXEL FAILED TO SET
010600  006337  001474          2$:     ASL     $TEMP                   ;ADJUST
010604  006337  001474                  ASL     $TEMP                   ;  THE
010610  006337  001474                  ASL     $TEMP                   ;    WRITE
010614  006337  001474                  ASL     $TEMP                   ;      VALUE TO THE NEXT PIXEL
010620  006337  001124                  ASL     $GDDAT                  ;ADJUST THE EXPECTED READ VALUE
010624  001333                          BNE     1$                      ;BR IF; MORE BITS TO TEST
```

```
1568
1569                                    ;;*****************************************************************
1570                                    ;*TEST 40        BASIC LOOK UP TABLE (L.U.T.) TEST
1571                                    ;;*****************************************************************
1572  010626  000004                    TST40:  SCOPE
1573  010630  005737  001544                    TST     $BURN                   ;TEST IF IN "BURN-IN" MODE
1574  010634  001406                            BEQ     2$                      ;BR IF NOT
1575  010636  013777  001546  170704            MOV     COAXSW,@COAX            ;CHANGE THE RELAY
1576  010644  004537  016666                    JSR     R5,DELAY                ;WAIT FOR RELAY TO SETTLE
1577  010650  000012                            10.
1578  010652                            2$:
1579  010652  005737  001200                    TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
1580  010656  001422                            BEQ     TST41           ;;BR IF FIRST PASS
1581  010660  032777  001000  170252            BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
1582  010666  001016                            BNE     TST41           ;;BR IF INHIBIT IS SET
1583  010670                            1$:
1584  010670  004737  017306                    JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
1585
1586  010674  004737  016234                    JSR     PC,CLRMAP               ;CLEAR THE BIT MAP
1587
1588  010700  004537  017062                    JSR     R5,LODSTA               ;LOAD THE SEQUENCE TABLE
1589  010704  000000                            0                               ;LOC 0 WITH 14 - LOC 1-17 WITH 63
1590  010706  016506                            TABLE3
1591
1592  010710  012777  002400  170542            MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
1593
1594  010716  004537  016666                    JSR     R5,DELAY
1595  010722  000144                            100.
1596
1597                                    ;;*****************************************************************
1598                                    ;*TEST 41        STATIC INTENSITY LEVEL TEST (L.U.T. 0)
1599                                    ;;*****************************************************************
1600  010724  000004                    TST41:  SCOPE
1601  010726  005737  001200                    TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
1602  010732  001427                            BEQ     TST42           ;;BR IF FIRST PASS
1603  010734  032777  001000  170176            BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
1604  010742  001023                            BNE     TST42           ;;BR IF INHIBIT IS SET
1605  010744  004737  017306                    JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
1606
1607  010750  004737  016234                    JSR     PC,CLRMAP               ;ENSURE CLEAR MAP
1608
1609  010754  012777  002400  170476            MOV     #BIT10!BIT8,@VGCSR      ;ENABLE 'MONO' AND 'MAP'
1610  010762  012737  000017  001474            MOV     #17,$TEMP
1611
1612  010770  013777  001474  170470  1$:       MOV     $TEMP,@VGCOLR           ;LOAD LOC. 0 WITH A INTENSITY LEVEL
1613  010776  004537  016666                    JSR     R5,DELAY
1614  011002  000012                            10.
1615  011004  005337  001474                    DEC     $TEMP                   ;REDUCE THE INTENSITY LEVEL
1616  011010  001367                            BNE     1$              ;;BR IF MORE LEVELS TO TEST
1617
```

```
1618                                        ;;*****************************************************************
1619                                        ;*TEST 42       DYNAMIC WORD TESTING OF PIXEL 0  (FIRST MAP LOCATION)
1620                                        ;;*****************************************************************
1621  011012  000004                       TST42:  SCOPE
1622  011014  005737  001200                       TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
1623  011020  001432                               BEQ     TST43           ;;BR IF FIRST PASS
1624  011022  032777  001000  170110               BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
1625  011030  001026                               BNE     TST43           ;;BR IF INHIBIT IS SET
1626  011032  004737  017306                       JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
1627
1628  011036  004737  016234                       JSR     PC,CLRMAP               ;CLEAR THE MAP
1629  011042  012777  002000  170410               MOV     #BIT10,@VGCSR           ;SET 'MONO'
1630
1631  011050  004537  017062                       JSR     R5,LODSTA               ;LOAD 4 INTO L.U.T. 0 AND
1632  011054  000000                               0                               ; LOAD #17 INTO L.U.T. 1
1633  011056  016526                               TABLE4
1634  011060  005077  170376                       CLR     @VGPC                   ;CLEAR MAP P.C.
1635  011064  012777  000001  170372               MOV     #1,@VGBUF               ;LOAD PIXEL0 BIT0
1636  011072  052777  000400  170360               BIS     #BIT8,@VGCSR            ;ENABLE THE MAP
1637  011100  004537  016666                       JSR     R5,DELAY
1638  011104  000144                               100.
1639
1640                                        ;;*****************************************************************
1641                                        ;*TEST 43       DYNAMIC WORD TESTING OF PIXEL 1  (FIRST MAP LOCATION)
1642                                        ;;*****************************************************************
1643  011106  000004                       TST43:  SCOPE
1644  011110  005737  001200                       TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
1645  011114  001432                               BEQ     TST44           ;;BR IF FIRST PASS
1646  011116  032777  001000  170014               BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
1647  011124  001026                               BNE     TST44           ;;BR IF INHIBIT IS SET
1648  011126  004737  017306                       JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
1649
1650  011132  004737  016234                       JSR     PC,CLRMAP               ;CLEAR THE MAP
1651  011136  012777  002000  170314               MOV     #BIT10,@VGCSR           ;SET 'MONO'
1652
1653  011144  004537  017062                       JSR     R5,LODSTA               ;LOAD 4 INTO L.U.T. 0 AND
1654  011150  000000                               0                               ; LOAD #17 INTO L.U.T. 1
1655  011152  016526                               TABLE4
1656  011154  005077  170302                       CLR     @VGPC                   ;CLEAR MAP P.C.
1657  011160  012777  000020  170276               MOV     #20,@VGBUF              ;LOAD PIXEL0 BIT0
1658  011166  052777  000400  170264               BIS     #BIT8,@VGCSR            ;ENABLE THE MAP
1659  011174  004537  016666                       JSR     R5,DELAY
1660  011200  000144                               100.
1661
```

```
                                ;;*****************************************************************
                                ;*TEST 44       DYNAMIC WORD TESTING OF PIXEL 2  (FIRST MAP LOCATION)
                                ;;*****************************************************************
011202  000004                  TST44:  SCOPE
011204  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
011210  001432                          BEQ     TST45           ;;BR IF FIRST PASS
011212  032777  001000  167720          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
011220  001026                          BNE     TST45           ;;BR IF INHIBIT IS SET
011222  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #

011226  004737  016234                  JSR     PC,CLRMAP               ;CLEAR THE MAP
011232  012777  002000  170220          MOV     #BIT10,@VGCSR           ;SET 'MONO'

011240  004537  017062                  JSR     R5,LODSTA               ;LOAD 4 INTO L.U.T. 0 AND
011244  000000                          0                               ; LOAD #17 INTO L.U.T. 1
011246  016526                          TABLE4
011250  005077  170206                  CLR     @VGPC                   ;CLEAR MAP P.C.
011254  012777  000400  170202          MOV     #400,@VGBUF             ;LOAD PIXEL2 BIT0
011262  052777  000400  170170          BIS     #BIT8,@VGCSR            ;ENABLE THE MAP
011270  004537  016666                  JSR     R5,DELAY
011274  000144                          100.

                                ;;*****************************************************************
                                ;*TEST 45       DYNAMIC WORD TESTING OF PIXEL 3  (FIRST MAP LOCATION)
                                ;;*****************************************************************
011276  000004                  TST45:  SCOPE
011300  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
011304  001432                          BEQ     TST46           ;;BR IF FIRST PASS
011306  032777  001000  167624          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
011314  001026                          BNE     TST46           ;;BR IF INHIBIT IS SET
011316  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #

011322  004737  016234                  JSR     PC,CLRMAP               ;CLEAR THE MAP
011326  012777  002000  170124          MOV     #BIT10,@VGCSR           ;SET 'MONO'

011334  004537  017062                  JSR     R5,LODSTA               ;LOAD 4 INTO L.U.T. 0 AND
011340  000000                          0                               ; LOAD #17 INTO L.U.T. 1
011342  016526                          TABLE4
011344  005077  170112                  CLR     @VGPC                   ;CLEAR MAP P.C.
011350  012777  010000  170106          MOV     #10000,@VGBUF           ;LOAD PIXEL3 BIT0
011356  052777  000400  170074          BIS     #BIT8,@VGCSR            ;ENABLE THE MAP
011364  004537  016666                  JSR     R5,DELAY
011370  000144                          100.
```

```
                                ;;*****************************************************************
                                ;*TEST 46       DYNAMIC WORD TESTING OF 'PIXEL0' (INCREMENT MAP P.C.) TEST ...
                                ;;*****************************************************************
011372  000004                TST46:  SCOPE
011374  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
011400  001441                          BEQ     TST47           ;;BR IF FIRST PASS
011402  032777  001000  167530          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
011410  001035                          BNE     TST47           ;;BR IF INHIBIT IS SET
011412  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
011416  004737  016234                  JSR     PC,CLRMAP
011422  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
011426  000000                          0                               ;LOC 0 AND 2 THRU 17 WITH 0
011430  016466                          TABLE2                          ;LOC 1 WITH A 17

011432  005077  170024                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
011436  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
011442  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
011446  105777  170006          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
011452  100375                          BPL     1$
011454  012710  000001                  MOV     #1,(R0)                 ;LOAD BIT0 OF PIXEL0
011460  005301                          DEC     R1                      ;FINISHED ALL PIXEL0'S
011462  001371                          BNE     1$                      ;BR IF NOT
011464  012777  002400  167766          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
011472  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
011476  004537  016666                  JSR     R5,DELAY
011502  000144                          100.


                                ;;*****************************************************************
                                ;*TEST 47       DYNAMIC WORD TESTING OF 'PIXEL1' (INCREMENT MAP P.C.) TEST ...
                                ;;*****************************************************************
011504  000004                TST47:  SCOPE
011506  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
011512  001441                          BEQ     TST50           ;;BR IF FIRST PASS
011514  032777  001000  167416          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
011522  001035                          BNE     TST50           ;;BR IF INHIBIT IS SET
011524  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
011530  004737  016234                  JSR     PC,CLRMAP
011534  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
011540  000000                          0                               ;LOC 0 AND 2 THRU 17 WITH 0
011542  016466                          TABLE2                          ;LOC 1 WITH A 17

011544  005077  167712                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
011550  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
011554  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
011560  105777  167674          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
011564  100375                          BPL     1$
011566  012710  000020                  MOV     #20,(R0)                ;LOAD BIT0 OF PIXEL1
011572  005301                          DEC     R1                      ;FINISHED ALL PIXEL1'S
011574  001371                          BNE     1$                      ;BR IF NOT
011576  012777  002400  167654          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
011604  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
011610  004537  016666                  JSR     R5,DELAY
011614  000144                          100.
```

```
                                ;;*****************************************************************
                                ;*TEST 50       DYNAMIC WORD TESTING OF 'PIXEL2' (INCREMENT MAP P.C.) TEST L.U.T. 1
                                ;;*****************************************************************
011616  000004                TST50:  SCOPE
011620  005737  001200                TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
011624  001441                        BEQ     TST51           ;;BR IF FIRST PASS
011626  032777  001000  167304        BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
011634  001035                        BNE     TST51           ;;BR IF INHIBIT IS SET
011636  004737  017306                JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
011642  004737  016234                JSR     PC,CLRMAP
011646  004537  017062                JSR     R5,LODSTA               ;LOAD COLOR TABLE
011652  000000                        0                               ;LOC 0 AND 2 THRU 17 WITH 0
011654  016466                        TABLE2                          ;LOC 1 WITH A 17

011656  005077  167600                CLR     @VGPC                   ;LOAD MAP P.C. TO 0
011662  013700  001464                MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
011666  013701  001542                MOV     NUMPIX,R1               ;LOAD # OF PIXELS
011672  105777  167562        1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
011676  100375                        BPL     1$
011700  012710  000400                MOV     #400,(R0)               ;LOAD BIT0 OF PIXEL2
011704  005301                        DEC     R1                      ;FINISHED ALL PIXEL2'S
011706  001371                        BNE     1$                      ;BR IF NOT
011710  012777  002400  167542        MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
011716  004737  017554                JSR     PC,LDLIN                ;LOAD REFERENCE LINE
011722  004537  016666                JSR     R5,DELAY
011726  000144                        100.


                                ;;*****************************************************************
                                ;*TEST 51       DYNAMIC WORD TESTING OF 'PIXEL3' (INCREMENT MAP P.C.) TEST L.U.T. 1
                                ;;*****************************************************************
011730  000004                TST51:  SCOPE
011732  005737  001200                TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
011736  001441                        BEQ     TST52           ;;BR IF FIRST PASS
011740  032777  001000  167172        BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
011746  001035                        BNE     TST52           ;;BR IF INHIBIT IS SET
011750  004737  017306                JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
011754  004737  016234                JSR     PC,CLRMAP
011760  004537  017062                JSR     R5,LODSTA               ;LOAD COLOR TABLE
011764  000000                        0                               ;LOC 0 AND 2 THRU 17 WITH 0
011766  016466                        TABLE2                          ;LOC 1 WITH A 17

011770  005077  167466                CLR     @VGPC                   ;LOAD MAP P.C. TO 0
011774  013700  001464                MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
012000  013701  001542                MOV     NUMPIX,R1               ;LOAD # OF PIXELS
012004  105777  167450        1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
012010  100375                        BPL     1$
012012  012710  010000                MOV     #10000,(R0)             ;LOAD BIT0 OF PIXEL3
012016  005301                        DEC     R1                      ;FINISHED ALL PIXEL3'S
012020  001371                        BNE     1$                      ;BR IF NOT
012022  012777  002400  167430        MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
012030  004737  017554                JSR     PC,LDLIN                ;LOAD REFERENCE LINE
012034  004537  016666                JSR     R5,DELAY
012040  000144                        100.
```

```
                                ;;*****************************************************************
                                ;*TEST 52       DYNAMIC WORD TESTING OF 'PIXEL0' (INCREMENT MAP P.C.) TEST L.U.T. 2
                                ;;*****************************************************************
012042  000004                TST52:  SCOPE
012044  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
012050  001441                          BEQ     TST53           ;;BR IF FIRST PASS
012052  032777  001000  167760          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
012060  001035                          BNE     TST53           ;;BR IF INHIBIT IS SET
012062  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
012066  004737  016234                  JSR     PC,CLRMAP
012072  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
012076  000000                          0                               ;LOC 0,1 AND 3 THRU 17 WITH 0
012100  016606                          TABLE6                          ;LOC 2 WITH A 17
                                
012102  005077  167354                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
012106  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
012112  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
012116  105777  167336          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
012122  100375                          BPL     1$
012124  012710  000002                  MOV     #2,(R0)                 ;LOAD BIT1 OF PIXEL0
012130  005301                          DEC     R1                      ;FINISHED ALL PIXEL0'S
012132  001371                          BNE     1$                      ;BR IF NOT
012134  012777  002400  167316          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
012142  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
012146  004537  016666                  JSR     R5,DELAY
012152  000144                          100.
                                
                                ;;*****************************************************************
                                ;*TEST 53       DYNAMIC WORD TESTING OF 'PIXEL1' (INCREMENT MAP P.C.) TEST L.U.T. 2
                                ;;*****************************************************************
012154  000004                TST53:  SCOPE
012156  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
012162  001441                          BEQ     TST54           ;;BR IF FIRST PASS
012164  032777  001000  166746          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
012172  001035                          BNE     TST54           ;;BR IF INHIBIT IS SET
012174  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
012200  004737  016234                  JSR     PC,CLRMAP
012204  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
012210  000000                          0                               ;LOC 0,1 AND 3 THRU 17 WITH 0
012212  016606                          TABLE6                          ;LOC 2 WITH A 17
                                
012214  005077  167242                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
012220  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
012224  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
012230  105777  167224          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
012234  100375                          BPL     1$
012236  012710  000040                  MOV     #40,(R0)                ;LOAD BIT1 OF PIXEL1
012242  005301                          DEC     R1                      ;FINISHED ALL PIXEL1'S
012244  001371                          BNE     1$                      ;BR IF NOT
012246  012777  002400  167204          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
012254  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
012260  004537  016666                  JSR     R5,DELAY
012264  000144                          100.
```

```
                                        ;;*****************************************************************
                                        ;*TEST 54       DYNAMIC WORD TESTING OF 'PIXEL2' (INCREMENT MAP P.C.) TEST L.U.T. 2
                                        ;;*****************************************************************
012266  000004                TST54:  SCOPE
012270  005737  001200                        TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
012274  001441                        BEQ     TST55           ;;BR IF FIRST PASS
012276  032777  001000  166634        BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
012304  001035                        BNE     TST55           ;;BR IF INHIBIT IS SET
012306  004737  017306                JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
012312  004737  016234                JSR     PC,CLRMAP
012316  004537  017062                JSR     R5,LODSTA               ;LOAD COLOR TABLE
012322  000000                        0                               ;LOC 0,1 AND 3 THRU 17 WITH 0
012324  016606                        TABLE6                          ;LOC 2 WITH A 17

012326  005077  167130                CLR     @VGPC                   ;LOAD MAP P.C. TO 0
012332  013700  001464                MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
012336  013701  001542                MOV     NUMPIX,R1               ;LOAD # OF PIXELS
012342  105777  167112        1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
012346  100375                        BPL     1$
012350  012710  001000                MOV     #1000,(R0)              ;LOAD BIT1 OF PIXEL2
012354  005301                        DEC     R1                      ;FINISHED ALL PIXEL2'S
012356  001371                        BNE     1$                      ;BR IF NOT
012360  012777  002400  167072        MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
012366  004737  017554                JSR     PC,LDLIN                ;LOAD REFERENCE LINE
012372  004537  016666                JSR     R5,DELAY
012376  000144                        100.

                                        ;;*****************************************************************
                                        ;*TEST 55       DYNAMIC WORD TESTING OF 'PIXEL3' (INCREMENT MAP P.C.) TEST L.U.T. 2
                                        ;;*****************************************************************
012400  000004                TST55:  SCOPE
012402  005737  001200                        TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
012406  001441                        BEQ     TST56           ;;BR IF FIRST PASS
012410  032777  001000  166522        BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
012416  001035                        BNE     TST56           ;;BR IF INHIBIT IS SET
012420  004737  017306                JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
012424  004737  016234                JSR     PC,CLRMAP
012430  004537  017062                JSR     R5,LODSTA               ;LOAD COLOR TABLE
012434  000000                        0                               ;LOC 0,1 AND 3 THRU 17 WITH 0
012436  016606                        TABLE6                          ;LOC 2 WITH A 17

012440  005077  167016                CLR     @VGPC                   ;LOAD MAP P.C. TO 0
012444  013700  001464                MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
012450  013701  001542                MOV     NUMPIX,R1               ;LOAD # OF PIXELS
012454  105777  167000        1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
012460  100375                        BPL     1$
012462  012710  020000                MOV     #20000,(R0)             ;LOAD BIT1 OF PIXEL3
012466  005301                        DEC     R1                      ;FINISHED ALL PIXEL3'S
012470  001371                        BNE     1$                      ;BR IF NOT
012472  012777  002400  166760        MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
012500  004737  017554                JSR     PC,LDLIN                ;LOAD REFERENCE LINE
012504  004537  016666                JSR     R5,DELAY
012510  000144                        100.
```

```
                                ;;*****************************************************************
                                ;*TEST 56       DYNAMIC WORD TESTING OF 'PIXEL0' (INCREMENT MAP P.C.) TEST L.U.T. 4
                                ;;*****************************************************************
012512  000004                  TST56:  SCOPE
012514  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
012520  001441                          BEQ     TST57           ;;BR IF FIRST PASS
012522  032777  001000  166410          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
012530  001035                          BNE     TST57           ;;BR IF INHIBIT IS SET
012532  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
012536  004737  016234                  JSR     PC,CLRMAP
012542  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
012546  000000                          0                               ;LOC 0,1,2,3 AND 5 THRU 17 WITH 0
012550  016626                          TABLE7                          ;LOC 4 WITH A 17
                                
012552  005077  166704                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
012556  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
012562  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
012566  105777  166666          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
012572  100375                          BPL     1$
012574  012710  000004                  MOV     #4,(R0)                 ;LOAD BIT2 OF PIXEL0
012600  005301                          DEC     R1                      ;FINISHED ALL PIXEL0'S
012602  001371                          BNE     1$                      ;BR IF NOT
012604  012777  002400  166646          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
012612  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
012616  004537  016666                  JSR     R5,DELAY
012622  000144                          100.
                                
                                ;;*****************************************************************
                                ;*TEST 57       DYNAMIC WORD TESTING OF 'PIXEL1' (INCREMENT MAP P.C.) TEST L.U.T. 4
                                ;;*****************************************************************
012624  000004                  TST57:  SCOPE
012626  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
012632  001441                          BEQ     TST60           ;;BR IF FIRST PASS
012634  032777  001000  166276          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
012642  001035                          BNE     TST60           ;;BR IF INHIBIT IS SET
012644  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
012650  004737  016234                  JSR     PC,CLRMAP
012654  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
012660  000000                          0                               ;LOC 0,1,2,3 AND 5 THRU 17 WITH 0
012662  016626                          TABLE7                          ;LOC 4 WITH A 17
                                
012664  005077  166572                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
012670  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
012674  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
012700  105777  166554          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
012704  100375                          BPL     1$
012706  012710  000100                  MOV     #100,(R0)               ;LOAD BIT2 OF PIXEL1
012712  005301                          DEC     R1                      ;FINISHED ALL PIXEL1'S
012714  001371                          BNE     1$                      ;BR IF NOT
012716  012777  002400  166534          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
012724  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
012730  004537  016666                  JSR     R5,DELAY
012734  000144                          100.
```

```
                                ;;*****************************************************************
                                ;*TEST 60       DYNAMIC WORD TESTING OF 'PIXEL2' (INCREMENT MAP P.C.) TEST L.U.T. 4
                                ;;*****************************************************************
013736  000004                TST60:  SCOPE
012740  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
012744  001441                          BEQ     TST61           ;;BR IF FIRST PASS
012746  032777  001000  166164          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
012754  001035                          BNE     TST61           ;;BR IF INHIBIT IS SET
012756  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
012762  004737  016234                  JSR     PC,CLRMAP
012766  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
012772  000000                          0                               ;LOC 0,1,2,3 AND 5 THRL 17 WITH C
012774  016626                          TABLE7                          ;LOC 41 WITH A 17
                                
012776  005077  166460                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
013002  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
013006  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
013012  105777  166442          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
013016  100375                          BPL     1$
013020  012710  002000                  MOV     #2000,(R0)              ;LOAD BIT2 OF PIXEL2
013024  005301                          DEC     R1                      ;FINISHED ALL PIXEL2'S
013026  001371                          BNE     1$                      ;BR IF NOT
013030  012777  002400  166422          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
013036  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
013042  004537  016666                  JSR     R5,DELAY
013046  000144                          100.
                                
                                ;;*****************************************************************
                                ;*TEST 61       DYNAMIC WORD TESTING OF 'PIXEL3' (INCREMENT MAP P.C.) TEST L.U.T. 4
                                ;;*****************************************************************
013050  000004                TST61:  SCOPE
013052  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
013056  001441                          BEQ     TST62           ;;BR IF FIRST PASS
013060  032777  001000  166052          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
013066  001035                          BNE     TST62           ;;BR IF INHIBIT IS SET
013070  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
013074  004737  016234                  JSR     PC,CLRMAP
013100  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
013104  000000                          0                               ;LOC 0,1,2,3 AND 5 THRU 17 WITH 0
013106  016626                          TABLE7                          ;LOC 4 WITH A 17
                                
013110  005077  166346                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
013114  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
013120  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
013124  105777  166330          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
013130  100375                          BPL     1$
013132  012710  040000                  MOV     #40000,(R0)             ;LOAD BIT2 OF PIXEL3
013136  005301                          DEC     R1                      ;FINISHED ALL PIXEL3'S
013140  001371                          BNE     1$                      ;BR IF NOT
013142  012777  002400  166310          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
013150  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
013154  004537  016666                  JSR     R5,DELAY
013160  000144                          100.
```

```
                        ;;*****************************************************************
                        ;*TEST 62       DYNAMIC WORD TESTING OF 'PIXEL0' (INCREMENT MAP P.C.) TEST L.U.T. 10
                        ;;*****************************************************************
013162  000004          TST62:  SCOPE
013164  005737  001200          TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
013170  001441                  BEQ     TST63           ;;BR IF FIRST PASS
013172  032777  001000  165740  BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
013200  001035                  BNE     TST63           ;;BR IF INHIBIT IS SET
013202  004737  017306          JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
013206  004737  016234          JSR     PC,CLRMAP
013212  004537  017062          JSR     R5,LODSTA               ;LOAD COLOR TABLE
013216  000000                  0                               ;LOC 0,1,2,3,4,5,6,7 AND 11 THRU 17 WITH 0
013220  016646                  TABLE8                          ;LOC 10 WITH A 17

013222  005077  166234          CLR     @VGPC                   ;LOAD MAP P.C. TO 0
013226  013700  001464          MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
013232  013701  001542          MOV     NUMPIX,R1               ;LOAD # OF PIXELS
013236  105777  166216  1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
013242  100375                  BPL     1$
013244  012710  000010          MOV     #10,(R0)                ;LOAD BIT3 OF PIXEL0
013250  005301                  DEC     R1                      ;FINISHED ALL PIXEL0'S
013252  001371                  BNE     1$                      ;BR IF NOT
013254  012777  002400  166176  MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
013262  004737  017554          JSR     PC,LDLIN                ;LOAD REFERENCE LINE
013266  004537  016666          JSR     R5,DELAY
013272  000144                  100.

                        ;;*****************************************************************
                        ;*TEST 63       DYNAMIC WORD TESTING OF 'PIXEL1' (INCREMENT MAP P.C.) TEST L.L.T. 10
                        ;;*****************************************************************
013274  000004          TST63:  SCOPE
013276  005737  001200          TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
013302  001441                  BEQ     TST64           ;;BR IF FIRST PASS
013304  032777  001000  165626  BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
013312  001035                  BNE     TST64           ;;BR IF INHIBIT IS SET
013314  004737  017306          JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
013320  004737  016234          JSR     PC,CLRMAP
013324  004537  017062          JSR     R5,LODSTA               ;LOAD COLOR TABLE
013330  000000                  0                               ;LOC 0,1,2,3,4,5,6,7 AND 11 THRU 17 WITH 0
013332  016646                  TABLE8                          ;LOC 10 WITH A 17

013334  005077  166122          CLR     @VGPC                   ;LOAD MAP P.C. TO 0
013340  013700  001464          MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
013344  013701  001542          MOV     NUMPIX,R1               ;LOAD # OF PIXELS
013350  105777  166104  1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
013354  100375                  BPL     1$
013356  012710  000200          MOV     #200,(R0)               ;LOAD BIT3 OF PIXEL1
013362  005301                  DEC     R1                      ;FINISHED ALL PIXEL1'S
013364  001371                  BNE     1$                      ;BR IF NOT
013366  012777  002400  166064  MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
013374  004737  017554          JSR     PC,LDLIN                ;LOAD REFERENCE LINE
013400  004537  016666          JSR     R5,DELAY
013404  000144                  100.
```

```
2081                                    ;;*****************************************************************
2082                                    ;*TEST 64       DYNAMIC WORD TESTING OF 'PIXEL2' (INCREMENT MAP P.C.) TEST L.U.T. 10
2083                                    ;;*****************************************************************
2084  013406  000004                    TST64:  SCOPE
2085  013410  005737  001200                    TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
2086  013414  001441                            BEQ     TST65           ;;BR IF FIRST PASS
2087  013416  032777  001000  165514            BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
2088  013424  001035                            BNE     TST65           ;;BR IF INHIBIT IS SET
2089  013426  004737  017306                    JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
2090  013432  004737  016234                    JSR     PC,CLRMAP
2091  013436  004537  017062                    JSR     R5,LODSTA               ;LOAD COLOR TABLE
2092  013442  000000                            0                               ;LOC 0,1,2,3,4,5,6,7 AND 11 THRU 17 WITH 0
2093  013444  016646                            TABLE8                          ;LOC 10 WITH A 17
2094
2095  013446  005077  166010                    CLR     @VGPC                   ;LOAD MAP P.C. TO 0
2096  013452  013700  001464                    MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
2097  013456  013701  001542                    MOV     NUMPIX,R1               ;LOAD # OF PIXELS
2098  013462  105777  165772            1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
2099  013466  100375                            BPL     1$
2100  013470  012710  004000                    MOV     #4000,(R0)              ;LOAD BIT3 OF PIXEL2
2101  013474  005301                            DEC     R1                      ;FINISHED ALL PIXEL2'S
2102  013476  001371                            BNE     1$                      ;BR IF NOT
2103  013500  012777  002400  165752            MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
2104  013506  004737  017554                    JSR     PC,LDLIN                ;LOAD REFERENCE LINE
2105  013512  004537  016666                    JSR     R5,DELAY
2106  013516  000144                            100.
2107
2108                                    ;;*****************************************************************
2109                                    ;*TEST 65       DYNAMIC WORD TESTING OF 'PIXEL3' (INCREMENT MAP P.C.) TEST L.U.T. 10
2110                                    ;;*****************************************************************
2111  013520  000004                    TST65:  SCOPE
2112  013522  005737  001200                    TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
2113  013526  001441                            BEQ     TST66           ;;BR IF FIRST PASS
2114  013530  032777  001000  165402            BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
2115  013536  001035                            BNE     TST66           ;;BR IF INHIBIT IS SET
2116  013540  004737  017306                    JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
2117  013544  004737  016234                    JSR     PC,CLRMAP
2118  013550  004537  017062                    JSR     R5,LODSTA               ;LOAD COLOR TABLE
2119  013554  000000                            0                               ;LOC 0,1,2,3,4,5,6,7 AND 11 THRU 17 WITH 0
2120  013556  016646                            TABLE8                          ;LOC 10 WITH A 17
2121
2122  013560  005077  165676                    CLR     @VGPC                   ;LOAD MAP P.C. TO 0
2123  013564  013700  001464                    MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
2124  013570  013701  001542                    MOV     NUMPIX,R1               ;LOAD # OF PIXELS
2125  013574  105777  165660            1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
2126  013600  100375                            BPL     1$
2127  013602  012710  100000                    MOV     #100000,(R0)            ;LOAD BIT3 OF PIXEL3
2128  013606  005301                            DEC     R1                      ;FINISHED ALL PIXEL3'S
2129  013610  001371                            BNE     1$                      ;BR IF NOT
2130  013612  012777  002400  165640            MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
2131  013620  004737  017554                    JSR     PC,LDLIN                ;LOAD REFERENCE LINE
2132  013624  004537  016666                    JSR     R5,DELAY
2133  013630  000144                            100.
```

```
                                ;;*****************************************************************
                                ;*TEST 66       DYNAMIC BYTE TESTING OF 'PIXEL0'
                                ;;*****************************************************************
013632  000004                  TST66:  SCOPE
013634  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
013640  001441                          BEQ     TST67           ;;BR IF FIRST PASS
013642  032777  001000  165270          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
013650  001035                          BNE     TST67           ;;BR IF INHIBIT IS SET
013652  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
013656  004737  016234                  JSR     PC,CLRMAP
013662  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
013666  000000                          0                               ;LOC 0 AND 2 THRU 17 WITH 17
013670  016466                          TABLE2                          ;LOC 1 WITH A 4
013672  005077  165564                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
013676  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
013702  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
013706  105777  165546          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
013712  100375                          BPL     1$
013714  112710  000001                  MOVB    #1,(R0)                 ;LOAD BIT0 OF PIXEL0
013720  005301                          DEC     R1                      ;FINISHED ALL PIXEL 0'S
013722  001371                          BNE     1$                      ;BR IF NOT
013724  012777  002400  165526          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
013732  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
013736  004537  016666                  JSR     R5,DELAY
013742  000144                          100.
                                ;;*****************************************************************
                                ;*TEST 67       DYNAMIC BYTE TESTING OF 'PIXEL1'
                                ;;*****************************************************************
013744  000004                  TST67:  SCOPE
013746  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
013752  001442                          BEQ     TST70           ;;BR IF FIRST PASS
013754  032777  001000  165156          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
013762  001036                          BNE     TST70           ;;BR IF INHIBIT IS SET
013764  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
013770  004737  016234                  JSR     PC,CLRMAP
013774  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
014000  000000                          0                               ;LOC 0 AND 2 THRU 17 WITH 17
014002  016466                          TABLE2                          ;LOC 1 WITH A 4
014004  005077  165452                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
014010  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
014014  005200                          INC     R0                      ;MAKE PIXEL 1 BYTE ADDRESS
014016  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
014022  105777  165432          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
014026  100375                          BPL     1$
014030  112710  000001                  MOVB    #1,(R0)                 ;LOAD BIT0 OF PIXEL 1
014034  005301                          DEC     R1                      ;FINISHED ALL PIXEL 1'S
014036  001371                          BNE     1$                      ;BR IF NOT
014040  012777  002400  165412          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
014046  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
014052  004537  016666                  JSR     R5,DELAY
014056  000144                          100.
```

```
                                        ;;*****************************************************************
                                        ;*TEST 70        DYNAMIC BYTE TESTING OF 'PIXEL2'
                                        ;;*****************************************************************
014060  000004                  TST70:  SCOPE
014062  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
014066  001442                          BEQ     TST71           ;;BR IF FIRST PASS
014070  032777  001000  165042          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
014076  001036                          BNE     TST71           ;;BR IF INHIBIT IS SET
014100  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
014104  004737  016234                  JSR     PC,CLRMAP
014110  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
014114  000000                          0                               ;LOC 0 AND 2 THRU 17 WITH 17
014116  016466                          TABLE2                          ;LOC 1 WITH A 4
014120  005077  165336                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
014124  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
014130  005720                          TST     (R0)+                   ;MAKE PIXEL 2 BYTE ADDRESS
014132  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
014136  105777  165316          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
014142  100375                          BPL     1$
014144  112710  000001                  MOVB    #1,(R0)                 ;LOAD BIT0 OF PIXEL 2
014150  005301                          DEC     R1                      ;FINISHED ALL PIXEL 2'S
014152  001371                          BNE     1$                      ;BR IF NOT
014154  012777  002400  165276          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
014162  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
014166  004537  016666                  JSR     R5,DELAY
014172  000144                          100.

                                        ;;*****************************************************************
                                        ;*TEST 71        DYNAMIC BYTE TESTING OF 'PIXEL3'
                                        ;;*****************************************************************
014174  000004                  TST71:  SCOPE
014176  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
014202  001446                          BEQ     TST72           ;;BR IF FIRST PASS
014204  032777  001000  164726          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
014212  001042                          BNE     TST72           ;;BR IF INHIBIT IS SET
014214  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
014220  004737  016234                  JSR     PC,CLRMAP
014224  004537  017062                  JSR     R5,LODSTA               ;LOAD COLOR TABLE
014230  000000                          0                               ;LOC 0 AND 2 THRU 17 WITH 17
014232  016466                          TABLE2                          ;LOC 1 WITH A 4
014234  005077  165222                  CLR     @VGPC                   ;LOAD MAP P.C. TO 0
014240  013700  001464                  MOV     VGBUF,R0                ;LOAD PIXEL ADDRESS
014244  062700  000003                  ADD     #3,R0                   ;MAKE PIXEL 3 BYTE ADDRESS
014250  013701  001542                  MOV     NUMPIX,R1               ;LOAD # OF PIXELS
014254  105777  165200          1$:     TSTB    @VGCSR                  ;WAIT FOR MAP READY
014260  100375                          BPL     1$
014262  112710  000001                  MOVB    #1,(R0)                 ;LOAD BIT0 OF PIXEL 3
014266  005301                          DEC     R1                      ;FINISHED ALL PIXEL 3'S
014270  001371                          BNE     1$                      ;BR IF NOT
014272  012777  002400  165160          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE MAP
014300  004737  017554                  JSR     PC,LDLIN                ;LOAD REFERENCE LINE
014304  004537  016666                  JSR     R5,DELAY
014310  000144                          100.
014312  042777  010000  165126          BIC     #BIT12,@VGCSR           ;DISABLE REF. LINE
```

```
                                ;;*****************************************************************
                                ;*TEST 72        TEST THE 'EXPAND' FUNCTION
                                ;;*****************************************************************
014320  000004                  TST72:  SCOPE
014322  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
014326  001427                          BEQ     TST73           ;;BR IF FIRST PASS
014330  032777  001000  164602          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
014336  001023                          BNE     TST73           ;;BR IF INHIBIT IS SET
014340  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
014344  004737  016234                  JSR     PC,CLRMAP
014350  012777  000017  165110          MOV     #17,@VGCOLR             ;LOAD MAX INTENSITY INTO LOCATION C
014356  012777  002400  165074          MOV     #BIT10!BIT8,@VGCSR      ;SET "MONO" AND ENABLE BIT MAP
014364  004537  016666                  JSR     R5,DELAY
014370  000144                          100.
014372  052777  004000  165060          BIS     #BIT11,@VGCSR           ;SET THE 'EXPAND' BIT
014400  004537  016666                  JSR     R5,DELAY
014404  000144                          100.
                                ;;*****************************************************************
                                ;*TEST 73        DRAW A BOX UP ON VSVD1 SCREEN
                                ;;*****************************************************************
014406  000004                  TST73:  SCOPE
014410  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
014414  001476                          BEQ     TST74           ;;BR IF FIRST PASS
014416  032777  001000  164514          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
014424  001072                          BNE     TST74           ;;BR IF INHIBIT IS SET
014426  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
014432  005000                          CLR     R0                      ;SET UP BIT MAP ADRS - TOP LINE
014434  012701  073567                  MOV     #73567,R1               ;SET UP PIXEL DATA IN R1 - INT 7
014440  004737  015770                  JSR     PC,DISPY                ;GO LOAD BIT MAP
014444  005200                  1$:     INC     R0                      ;ADVANCE BIT MAP ADRS
014446  022700  000040                  CMP     #40,R0                  ;TOP LINE DONE?
014452  001403                          BEQ     2$                      ;BR IF SO
014454  004737  016002                  JSR     PC,DCONT                ;LOAD NEXT PIXEL
014460  000771                          BR      1$                      ;NEXT MAP LOAD
014462  012700  007740          2$:     MOV     #7740,R0                ;SET UP BIT MAP ADRS - BOTTOM LINE
014466  004737  015770                  JSR     PC,DISPY                ;GO LOAD BIT MAP
014472  005200                  3$:     INC     R0                      ;ADVANCE ADRS
014474  022700  010000                  CMP     #10000,R0               ;BOTTOM LINE DONE?
014500  001403                          BEQ     4$                      ;BR IF SO
014502  004737  016002                  JSR     PC,DCONT                ;LOAD NEXT PIXEL WORD
014506  000771                          BR      3$                      ;NEXT MAP LOAD
014510  012700  000040          4$:     MOV     #40,R0                  ;SET UP BIT MAP ADRS SIDE LINES
014514  012701  000007          5$:     MOV     #7,R1                   ;SET UP PIXEL 0 DATA IN R1 - INT 7
014520  004737  015770                  JSR     PC,DISPY                ;GO LOAD BIT MAP
014524  012701  070000                  MOV     #70000,R1               ;SET UP PIXEL 3 DATA IN R1 - INT 7
014530  062700  000037                  ADD     #37,R0                  ;OFFSET TO RIGHT SIDE LINE
014534  004737  015770                  JSR     PC,DISPY                ;GO LOAD BIT MAP
014540  005200                          INC     R0                      ;GO TO NEXT ROW ON LEFT
014542  022700  007740                  CMP     #7740,R0                ;TO BOTTOM YET?
014546  001362                          BNE     5$                      ;BR IF NOT
014550  005077  164712                  CLR     @VGCOLR                 ;LOAD L.U.T. 0
014554  012777  003417  164704          MOV     #3417,@VGCOLR           ;LOAD L.U.T. 7
                                ;NOW DISPLAY NORMAL SIZE BOX
014562  012777  002400  164670          MOV     #BIT10!BIT8,@VGCSR      ;ENABLE MAP
014570  004537  016666                  JSR     R5,DELAY                ;DELAY FOR OPERATOR
014574  000144                          100.
```

```
2296                                    ;NOW DISPLAY EXPANDED SIZE BOX
2297  014576  052777  004000  164654            BIS     #BIT11,@VGCSR           ;EXPAND MAP
2298  014604  004537  016666                    JSR     R5,DELAY                ;DELAY FOR OPERATOR
2299  014610  000144                            100.
2300                                    ;;*****************************************************************
2301                                    ;*TEST 74       TEST THE 'ORIGIN' FUNCTION
2302                                    ;;*****************************************************************
2303  014612  000004                    TST74:  SCOPE
2304  014614  005737  001200                    TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
2305  014620  001434                            BEQ     TST75           ;;BR IF FIRST PASS
2306  014622  032777  001000  164310            BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
2307  014630  001030                            BNE     TST75           ;;BR IF INHIBIT IS SET
2308  014632  004737  017306                    JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
2309  014636  004737  016234                    JSR     PC,CLRMAP               ;CLEAR THE MAP
2310  014642  012777  000017  164616            MOV     #17,@VGCOLR             ;LOAD MAX INTENSITY INTO TABLE LOC. 0
2311  014650  012777  002400  164602            MOV     #BIT10!BIT8,@VGCSR      ;ENABLE THE BIT MAP
2312  014656  005037  001474                    CLR     $TEMP
2313  014662  113777  001474  164570  1$:       MOVB    $TEMP,@VGCSR            ;LOAD ORIGIN REGISTER
2314  014670  004537  016666                    JSR     R5,DELAY
2315  014674  000144                            100.
2316  014676  105237  001474                    INCB    $TEMP                   ;UPDATE THE ORIGIN POSITION
2317  014702  122737  000020  001474            CMPB    #20,$TEMP               ;TEST FOR A NON-VALID ORIGIN
2318  014710  001364                            BNE     1$                      ;BR IF A VALID ORIGIN
2319
2320                                    ;;*****************************************************************
2321                                    ;*TEST 75       INTENSITY LEVEL TEST-MONOCROME
2322                                    ;;*****************************************************************
2323  014712  000004                    TST75:  SCOPE
2324  014714  005737  001200                    TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
2325  014720  001435                            BEQ     TST76           ;;BR IF FIRST PASS
2326  014722  032777  001000  164210            BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
2327  014730  001031                            BNE     TST76           ;;BR IF INHIBIT IS SET
2328  014732  004737  017306                    JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
2329  014736  004737  016234                    JSR     PC,CLRMAP               ;CLEAR THE BIT MAP
2330  014742  004737  016774                    JSR     PC,LODSEQ               ;LOAD TWO WORDS PER LOOK TABLE LOCATION
2331  014746  005037  014764                    CLR     2$                      ;CLEAR STARTING INTENSITY LEVEL
2332  014752  012777  006400  164500            MOV     #BIT11!BIT10!BIT8,@VGCSR        ;SET MONOCROME + ENABLE MAP
2333  014760  004537  017062          1$:       JSR     R5,LODSTA               ;LOAD SEQUENCE
2334  014764  000000                  2$:       0                               ;INDEX INTO DATA VALVE TABLE
2335  014766  016446                            TABLE1
2336  014770  004537  016666                    JSR     R5,DELAY
2337  014774  000144                            100.
2338  014776  062737  000001  014764            ADD     #1,2$                   ;UPDATE INDEX INTO TABLE FOR LOCATION 0'S VALUE
2339  015004  022737  000020  014764            CMP     #20,2$                  ;TEST IF LAST VLUE FOR MONOCROME
2340  015012  001362                            BNE     1$                      ;BR IF MORE LEVELS FOR LOCATION 0
```

```
                                ;;*****************************************************************
                                ;*TEST 76        TEST THE GREEN BITS
                                ;;*****************************************************************
015014  000004                  TST76:  SCOPE
015016  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
015022  001435                          BEQ     TST77           ;;BR IF FIRST PASS
015024  032777  001000  164106          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
015032  001031                          BNE     TST77           ;;BR IF INHIBIT IS SET
015034  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
015040  004737  016234                  JSR     PC,CLRMAP               ;CLEAR THE MAP
015044  012777  004400  164406          MOV     #BIT11!BIT8,@VGCSR      ;ENABLE MAP AND EXPAND
015052  012777  000004  164406          MOV     #4,@VGCOLR              ;LOAD TABLE POINTER0 - LOW GREEN LEVEL
015060  004537  016666                  JSR     R5,DELAY
015064  000144                          100.
015066  012777  000010  164372          MOV     #10,@VGCOLR             ;SET THE MEDIUM GREEN LEVEL
015074  004537  016666                  JSR     R5,DELAY
015100  000144                          100.
015102  012777  000014  164356          MOV     #14,@VGCOLR             ;SET THE BRIGHTEST BREEN LEVEL
015110  004537  016666                  JSR     R5,DELAY
015114  000144                          100.

                                ;;*****************************************************************
                                ;*TEST 77        TEST THE RED BITS
                                ;;*****************************************************************
015116  000004                  TST77:  SCOPE
015120  005737  001200                  TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
015124  001435                          BEQ     TST100          ;;BR IF FIRST PASS
015126  032777  001000  164004          BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
015134  001031                          BNE     TST100          ;;BR IF INHIBIT IS SET
015136  004737  017306                  JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
015142  004737  016234                  JSR     PC,CLRMAP
015146  012777  004400  164304          MOV     #BIT11!BIT8,@VGCSR      ;ENABLE MAP AND EXPAND
015154  012777  000001  164304          MOV     #1,@VGCOLR              ;SET THE DIMMEST RED LEVEL
015162  004537  016666                  JSR     R5,DELAY
015166  000144                          100.
015170  012777  000002  164270          MOV     #2,@VGCOLR              ;SET THE MEDIUM RED LEVEL
015176  004537  016666                  JSR     R5,DELAY
015202  000144                          100.
015204  012777  000003  164254          MOV     #3,@VGCOLR              ;SET THE BRIGHTEST RED LEVEL
015212  004537  016666                  JSR     R5,DELAY
015216  000144                          100.
```

B05


```
                                    ;;*************************************************************
                                    ;;*TEST 100      TEST THE BLUE COLOR BITS AT TABLE ADDRESS 0
                                    ;;*************************************************************
015220  000004                      TST100: SCOPE
015222  005737  001200                      TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
015226  001435                              BEQ     TST101          ;;BR IF FIRST PASS
015230  032737  001000  163702              BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
015236  001031                              BNE     TST101          ;;BR IF INHIBIT IS SET
015240  004737  017306                      JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
015244  004737  016234                      JSR     PC,CLRMAP
015250  012737  004400  164202              MOV     #BIT11!BIT8,@VGCSR      ;ENABLE MAP AND EXPAND
015256  012737  000020  164202              MOV     #20,@VGCOLR             ;SET THE DIMMEST BLUE LEVEL
015264  004537  016666                      JSR     R5,DELAY
015270  000144                              100.
015272  012737  000040  164166              MOV     #40,@VGCOLR             ;SET THE MEDIUM BLUE LEVEL
015300  004537  016666                      JSR     R5,DELAY
015304  000144                              100.
015306  012737  000060  164152              MOV     #60,@VGCOLR             ;SET THE BRIGHTEST BLUE LEVEL
015314  004537  016666                      JSR     R5,DELAY
015320  000144                              100.
```

```
                                        ;;*****************************************************************
                                        ;*TEST 101      COLOR LEVEL TEST - COLORCROME
                                        ;;*****************************************************************
015322  000004                          TST101: SCOPE
015324  005737  001200                          TST     $PASS                   ;TEST IF FIRST PASS OF PROGRAM
015330  001441                                  BEQ     TST102          ;;BR IF FIRST PASS
015332  032777  001000  163600                  BIT     #SW09,@SWR              ;TEST IF INHIBIT VISUAL MAP PATTERNS
015340  001035                                  BNE     TST102          ;;BR IF INHIBIT IS SET
015342  004737  017306                          JSR     PC,TESTID               ;DISPLAY MAP # AND TEST #
015346  004737  016234                          JSR     PC,CLRMAP               ;ENSURE CLEAR MAP
015352  004737  016774                          JSR     PC,LODSEQ               ;LOAD TWO WORDS PER LOOKUP TABLE LOC.
015356  005037  015372                          CLR     2$                      ;CLEAR INDEX POINTER
015362  005077  164072                          CLR     @VGCSR                  ;CLEAR STATUS
015366  004537  017362          1$:             JSR     R5,LODSTA               ;LOAD DATA INTO TABLE
015372  000000                  2$:             0
015374  016546                                  TABLE5
015376  012777  004400  164054                  MOV     #BIT11!BIT8,@VGCSR      ;ENABLE MAP
015404  004537  016666                          JSR     R5,DELAY
015410  000144                                  100.
015412  062737  000001  015372                  ADD     #1,2$                   ;UPDATE LOC 0 INDEX INTO TABLE
015420  022737  000020  015372                  CMP     #20,2$                  ;TEST IF FINISHED FOR LOC 0
015426  001357                                  BNE     1$                      ;
015430  005077  164024                          CLR     @VGCSR                  ;DISABLE MAP

                                        ;;*****************************************************************
                                        ;*TEST 102      LOGICAL END OF BIT MAP TEST
                                        ;;*****************************************************************
015434  000004                          TST102: SCOPE
015436  000137  007174                          JMP     MAPRET                  ;RETURN AND TEST IF MORE MAPS

                                                .SBTTL
                                                .SBTTL  OPERATOR INTERVENTION AND INSPECTION TESTS
                                                .SBTTL
```

```
2438                                                    .SBTTL  CROSS HATCH TEST PATTERN
2439
2440  015442  012706  001100          HATCH:  MOV     #STACK,SP               ;LOAD SP
2441  015446  004737  003576                  JSR     PC,LODADR               ;LOAD BUS ADDRESSES FOR MAP #1
2442  015452  004737  015510                  JSR     PC,HATCHO               ;LOAD MAP #0
2443  015456  012777  004400  163774          MOV     #BIT11!BIT8,@VGCSR      ;ENABLE MAP EXPAND AND ORGIN =0
2444  015464  004737  003730                  JSR     PC,MOVADR               ;ADJUST BUS ADDRESSES TO MAP #1
2445  015470  004737  015510                  JSR     PC,HATCHO               ;LOAD MAP #1
2446  015474  012777  004402  163756          MOV     #BIT11!BIT8!BIT1,@VGCSR ;ENABLE MAP, EXPAND AND ORIGIN =2
2447  015502  000000                          HALT
2448  015504  000137  001572                  JMP     BEGIN
2449
2450
2451                                          ;ACTUAL LOADING OF CROSS HATCH INTO BIT MAP SUB-ROUTINE
2452  015510  004737  016234          HATCHO: JSR     PC,CLRMAP               ;ENSURE CLEAR MAP
2453  015514  012737  000022  015636          MOV     #22,10$                 ;LOAD VERT. COUNTER
2454  015522  005077  163734                  CLR     @VGPC                   ;CLEAR MAP P.C.
2455  015526  012701  000040          1$:     MOV     #32.,R1                 ;LOAD WIDTH COUNTER
2456  015532  105777  163722          2$:     TSTB    @VGCSR                  ;TEST IF MAP READY SET
2457  015536  100375                          BPL     2$                      ;BR IF NOT READY
2458  015540  012777  177777  163716          MOV     #-1,@VGBUF              ;LOAD #17 INTO ALL PIXELS
2459  015546  005301                          DEC     R1
2460  015550  001370                          BNE     2$                      ;BR UNTIL FINISHED
2461  015552  012737  000140  015640          MOV     #140,11$                ;LOAD VERT COUNT
2462  015560  105777  163674          3$:     TSTB    @VGCSR                  ;TEST IF MAP READY IS SET
2463  015564  100375                          BPL     3$                      ;BR IF NOT
2464  015566  012777  000017  163670          MOV     #17,@VGBUF              ;LOAD VERT LINE PIXEL
2465  015574  105777  163660          4$:     TSTB    @VGCSR                  ;TEST IF READY
2466  015600  100375                          BPL     4$                      ;WAIT IF NOT
2467  015602  005077  163656                  CLR     @VGBUF                  ;CLEAR PIXELS
2468
2469  015606  005337  015640                  DEC     11$
2470  015612  001362                          BNE     3$                      ;BR UNTIL FUNISHED SEVERAL ROWS OF PIXELS
2471
2472  015614  005337  015636                  DEC     10$                     ;TEST IF FINISHED THE SCREEN
2473  015620  001342                          BNE     1$                      ;BR IF NOT
2474
2475  015622  005077  163640                  CLR     @VGCOLR                 ;CLEAR L.U.T. 0
2476  015626  012777  007477  163632          MOV     #7477,@VGCOLR           ;LOAD "WHITE" INTO L.U.T. #17
2477  015634  000207                          RTS     PC                      ;EXIT
2478
2479  015636  000000          10$:    0
2480  015640  000000          11$:    0
2481
2482                                                  .SBTTL  SWR CONTROL OF X AND Y CROSS HAIRS LOOP
2483  015642  012706  001100          CROSS:  MOV     #STACK,SP               ;LOAD STACK
2484  015646  004737  003576                  JSR     PC,LODADR               ;LOAD DEVICE ADDRESS
2485  015652  012777  014000  163566          MOV     #BIT12!BIT11,@VCCSR     ;ENABLE CROSS HAIRS
2486  015660  104407          1$:     CKSWR                                   ;TEST FOR CTRL G
2487  015662  017777  163252  163560          MOV     @SWR,@VCHRLO            ;LOAD THE CROSS HAIR POSITION REGISTERS
2488  015670  000773                          BR      1$                      ;LOOP BACK
2489
```

```
                                                .SBTTL
                                                .SBTTL  MISC SUBROUTINES
                                                .SBTTL
                                        .SBTTL  SUBROUTINE TO TEST THAT 'VERT SYNC' BIT CHANGES, REPORT ERROR IF
                                                ;BIT DOES NOT CHANGE STATES (0 TO 1 OR 1 TO 0)
015672  005237  015766          COUNTA: INC     CNTSAV                  ;SET EXIT FLAG
015676  012702  040000          COUNT:  MOV     #BIT14,R2               ;LOAD BIT 14
015702  005001                          CLR     R1                      ;CLEAR COUNT LOC.
015704  013700  001446                  MOV     VCCSR,R0                ;LOAD BUS ADDRESS
015710  030210                          BIT     R2,(R0)                 ;TEST IF BIT IS SET
015712  001411                          BEQ     3$                      ;BR IF SET
                                        ;COME HERE IF BIT 14 WAS INITIALLY SET - NOW WAIT FOR IT TO CLEAR
015714  005201                  1$:     INC     R1                      ;COUNT TIME
015716  001002                          BNE     2$                      ;BR IF NOT OVERFLOW
015720  104005                          ERROR   5                       ;VERTICAL SYNC FLAG ERROR
015722  000207                          RTS     PC                      ;EXIT
015724  030210                  2$:     BIT     R2,(R0)                 ;TEST IF BIT 14 IS CLEARED
015726  001372                          BNE     1$                      ;BR IF STILL SET
015730  005037  015766                  CLR     CNTSAV                  ;CLEAR EXIT FLAG
015734  000207                          RTS     PC                      ;EXIT
                                        ;COME HERE IF BIT 14 WAS INITIALLY CLEARED - NOW WAIT FOR IT TO SET
015736  005201                  3$:     INC     R1                      ;COUNT TIME
015740  001002                          BNE     4$                      ;BR IF NOT OVERFLOW
015742  104005                          ERROR   5                       ;VERTICAL SYNC FLAG ERROR
015744  000207                          RTS     PC                      ;EXIT
015746  030210                  4$:     BIT     R2,(R0)                 ;TEST IF BIT 14 IS SET
015750  001772                          BEQ     3$                      ;BR IF CLEARED
015752  005737  015766                  TST     CNTSAV                  ;TEST IF EXIT FLAG IS SET
015756  001347                          BNE     COUNT                   ;BR IF SET
015760  005037  015766                  CLR     CNTSAV                  ;CLEAR AGAIN
015764  000207                          RTS     PC                      ;EXIT
015766  000000          CNTSAV: 0

015770  042777  000400  163462  DISPY:  BIC     #400,@VGCSR             ;DISABLE MAP
015774  010077  163460                  MOV     R0,@VGPC                ;LOAD MAP P.C.
016002  042777  000400  163450  DCONT:  BIC     #400,@VGCSR             ;DISABLE MAP
016010  105777  163444          1$:     TSTB    @VGCSR                  ;TEST IF MAP IS READY
016014  100375                          BPL     1$
016016  010177  163442                  MOV     R1,@VGBUF               ;LOAD A PIXEL
016022  052777  000400  163430          BIS     #400,@VGCSR             ;ENABLE MAP
016030  000207                          RTS     PC                      ;EXIT
```

```
                                        .SBTTL  DISPLAY BUFFER ON VTV01 SCREEN SUB-ROUTINE

016032  012700  024330          XPRNT:  MOV     #BUFFER,R0              ;LOAD BUFFER POINTER
016036  000240                  XPRNTA: NOP
016040  000240                          NOP
016042  000240                          NOP
016044  005777  163376          1$:     TST     @VTCSR                  ;TEST IF READY IS SET
016050  100375                          BPL     1$                      ;WAIT TILL DONE
016052  112077  163370                  MOVB    (R0)+,@VTCSR            ;LOAD THE CHARACTER
016056  122710  000377                  CMPB    #377,(R0)               ;TEST FOR TERM
016062  001370                          BNE     1$                      ;BR IF NOT TERM.
016064  000240                          NOP
016066  000240                          NOP
016070  000240                          NOP
016072  000240                          NOP
016074  000207                          RTS     PC                      ;EXIT

016076  013701  001500          FILLWC: MOV     KE,R1                   ;LOAD BASIC CHARACTER
016102  004737  016132                  JSR     PC,FILBUF               ;LOAD BUFFER WITH THE CHARACTER
016106  013737  001524  016130          MOV     VHO,10$                 ;LOAD VERTICAL COUNTER
016114  004737  016032          1$:     JSR     PC,XPRNT                ;DISPLAY A LINE
016120  005337  016130                  DEC     10$                     ;FINISHED ALL VERT LINES ?
016124  100373                          BPL     1$                      ;BR IF NOT
016126  000207                          RTS     PC                      ;EXIT
016130  000000          10$:    0

016132  013702  001522          FILBUF: MOV     WIDTH,R2        ;LOAD WIDTH VALUE
016136  012700  024330                  MOV     #BUFFER,R0      ;SET-UP BUFFER POINTER
016142  110120          1$:     MOVB    R1,(R0)+        ;SAVE THE CHARACTER IN THE BUFFER
016144  005302                  DEC     R2              ;FINISHED?
016146  001375                  BNE     1$              ;BRANCH IF NOT COMPLETED
016150  112720  000015          MOVB    #15,(R0)+       ;LOAD "CR"
016154  112720  000012          MOVB    #12,(R0)+               ;LOAD "LF"
016160  112710  000377          MOVB    #377,(R0)       ;LOAD TERM.
016164  000207                  RTS     PC              ;EXIT

                        ;LOAD A INCREMENTING CHARACTER ACROSS THE SCREEN WIDTH
                        ;ONLY 40 THRU 177 ARE LEGAL CHARACTERS

016166  012700  024330  LIC:     MOV     #BUFFER,R0      ;SET-UP BUFFER POINTER
016172  013502                  MOV     @(5)+,R2        ;SET-UP WIDTH
016174  110120          1$:     MOVB    R1,(0)+         ;SAVE A CHARACTER IN THE BUFFER
016176  005201                  INC     R1              ;UPDATE THE CHARACTER
016200  023701  001532          CMP     LASTCH,R1       ;TEST FOR
016204  001002                  BNE     2$              ;BRANCH IF NOT
016206  012701  000040          MOV     #40,R1          ;MAKE A LEGAL CHARACTER
016212  005302          2$:     DEC     R2              ;DECREMENT COUNT
016214  001367                  BNE     1$              ;BRANCH IF NOT COMPLETED
016216  112720  000015          MOVB    #15,(R0)+       ;LOAD "CR"
016222  112720  000012          MOVB    #12,(R0)+       ;LOAD "LF"
016226  112710  000377          MOVB    #377,(R0)       ;LOAD TERM
016232  000205                  RTS     R5              ;EXIT
```

```
2584                                        ;SUBROUTINE TO CLEAR THE BIT MAP
2585                                        ;
2586  016234  012737  000100  016316  CLRMAP: MOV     #BIT6,10$               ;LOAD BASE DELAY
2587  016242  005037  016320                  CLR     11$                     ;CLEAR COUNTER
2588  016246  004537  016666                  JSR     R5,DELAY
2589  016252  100001                          BIT15!1
2590  016254  000240                          NOP
2591  016256  000240                          NOP
2592  016260  000240                          NOP
2593  016262  012777  001000  163170          MOV     #BIT9,@VGCSR            ;SET 'CLEAR MAP' BIT
2594  016270  105777  163164          1$:     TSTB    @VGCSR                  ;TEST READY FLAG
2595  016274  100407                          BMI     2$                      ;BR IF READY SET
2596  016276  005237  016320                  INC     11$                     ;UPDATE COUNTER
2597  016302  001372                          BNE     1$                      ;BR IF NO OVERFLOW
2598  016304  005337  016316                  DEC     10$                     ;DECREMENT BASE DELAY
2599  016310  001367                          BNE     1$                      ;BR IF NO OVERFLOW
2600  016312  104015                          ERROR   15                      ;MAP READY FAILED TO SET AFTER A GROSS
2601                                                                          ;TIME DELAY
2602  016314  000207                  2$:     RTS     PC                      ;EXIT
2603
2604  016316  000030                  10$:    0                               ;BASE DELAY
2605  016320  000000                  11$:    0                               ;COUNT LOCATION
2606
2607                                        .SBTTL  CROSS HAIRS WITH 2 BIT MAPS ENABLED
2608
2609  016322  012706  001100          FRANKD: MOV     #STACK,SP
2610  016326  004737  003576                  JSR     PC,LODADR               ;LOAD DEVICE ADDRESS
2611  016332  004737  016234                  JSR     PC,CLRMAP
2612  016336  012777  000014  163122          MOV     #14,@VGCOLR             ;LOAD LUT TO GREEN
2613  016344  012777  000400  163108          MOV     #BIT8,@VGCSR            ;ENABLE THE MAP
2614  016352  004737  003730                  JSR     PC,MOVADR               ;UPDATE TO NEXT MAP ADDRESS
2615  016356  004737  016234                  JSR     PC,CLRMAP
2616  016362  012777  000014  163076          MOV     #14,@VGCOLR             ;LOAD LUT TO GREEN
2617  016370  012777  004404  163062          MOV     #BIT11!BIT8!4,@VGCSR    ;ENABME MAP - EXPAND - ORIGIN 4
2618  016376  012777  014000  163042          MOV     #BIT12!BIT11,@VGCSR     ;ENABLE CROSS HAIRS
2619  016404  104407                  1$:     CKSWR                           ;TEST FOR CTRL G
2620  016406  017777  162526  163034          MOV     @SWR,@VCHRLO            ;LOAD CROSS HAIR ORIGIN
2621  016414  000773                          BR      1$
2622
2623                                        .SBTTL  BUS TIME-OUT LOOP
2624
2625  016416  012706  001100          BUSOUT: MOV     #STACK,SP               ;LOAD SP
2626  016422  012737  016440  000004          MOV     #2$,ERRVEC              ;LOAD BUS TRAP RETURN
2627  016430  005777  162612          1$:     TST     @$BASE                  ;REFERENCE THE ADDRESS
2628  016434  000775                          BR      1$
2629  016436  000240                          NOP
2630  016440  022626                  2$:     CMP     (SP)+,(SP)+             ;CLEAN THE STACK
2631  016442  000772                          BR      1$
2632  016444  000240                          NOP
```

```
 2633  016446  000  001  002  TABLE1: .BYTE   0,1,2,3,4,5,6,7,10,11,12,13,14,15,16,17
 2634  016451  003  004  005
 2635  016454  006  007  010
 2636  016457  011  012  013
 2637  016462  014  015  016
 2638  016465  017
 2639  016466  000  017  000  TABLE2: .BYTE   0,17,0,0,0,0,0,0,0,0,0,0,0,0,0,0        ;L.U.T.1 = 17 OTHERS =0
 2640  016471  000  000  000
 2641  016474  000  000  000
 2642  016477  000  000  000
 2643  016502  000  000  000
 2644  016505  000
 2645  016506  014  063  063  TABLE3: .BYTE   14,63,63,63,63,63,63,63,63,63,63,63,63,63,63,63
 2646  016511  063  063  063
 2647  016514  063  063  063
 2648  016517  063  063  063
 2649  016522  063  063  063
 2650  016525  063
 2651  016526  004  017  004  TABLE4: .BYTE   4,17,4,4,4,4,4,4,4,4,4,4,4,4,4,4
 2652  016531  004  004  004
 2653  016534  004  004  004
 2654  016537  004  004  004
 2655  016542  004  004  004
 2656  016545  004
 2657  016546  001  002  003  TABLE5: .BYTE   1,2,3,4,10,14,20,40,60,63,74,17,33,25,52,77
 2658  016551  004  010  014
 2659  016554  020  040  060
 2660  016557  063  074  017
 2661  016562  033  025  052
 2662  016565  077
 2663  016566  001  002  003          .BYTE   1,2,3,4,10,14,20,40,60,63,74,17,33,25,52,77
 2664  016571  004  010  014
 2665  016574  020  040  060
 2666  016577  063  074  017
 2667  016602  033  025  052
 2668  016605  077
 2669  016606  000  000  017  TABLE6: .BYTE   0,0,17,0,0,0,0,0,0,0,0,0,0,0,0,0        ;L.U.T. 2 = 17  OTHERS = 0
 2670  016611  000  000  000
 2671  016614  000  000  000
 2672  016617  000  000  000
 2673  016622  000  000  000
 2674  016625  000
 2675  016626  000  000  000  TABLE7: .BYTE   0,0,0,0,17,0,0,0,0,0,0,0,0,0,0,0        ;L.U.T. 4 = 17  OTHERS = 0
 2676  016631  000  017  000
 2677  016634  000  000  000
 2678  016637  000  000  000
 2679  016642  000  000  000
 2680  016645  000
 2681  016646  000  000  000  TABLE8: .BYTE   0,0,0,0,0,0,0,17,0,0,0,0,0,0,0,0        ;L.U.T. 10 = 17  OTHERS = 0
 2682  016651  000  000  000
 2683  016654  000  000  017
 2684  016657  000  000  000
 2685  016662  000  000  000
 2686  016665  000
 2687                          .EVEN
```

```
                                        .SBTTL  DELAY SUBROUTINE -- WAIT FOR "VERTICAL SYNC INTERRUPT"

016666  012777  016772  162574  DELAY:  MOV     #5$,@VCINT              ;LOAD INTR. RETURN ADDRESS
016674  012777  000200  162570          MOV     #APRIOR,@VCINT1         ;LOAD RETURN PRIORITY
016702  012537  016770                  MOV     (R5)+,10$               ;GET THE ARG.
016706  100003                          BPL     2$                      ;BR IF POSITIVE

016710  005737  016770          1$:     TST     10$                     ;TEST IF DELAY CANNOT BE INHIBITED
016714  100404                          BMI     3$                      ;BR IF IT CANT
016716  032777  010000  162214  2$:     BIT     #BIT12,@SWR             ;TEST IF INHIBIT DELAY SWITCH IS SET
016724  001013                          BNE     4$                      ;BR IF INHIBIT IS SET
016726  052777  020000  162512  3$:     BIS     #BIT13,@VCCSR           ;ENABLE SYNC INTR.
016734  000001                          WAIT                            ;WAIT FOR INTERRUPT TO HAPPEN
016736  042777  020000  162502          BIC     #BIT13,@VCCSR           ;DISABLE INTR.
016744  104407                          CKSWR                   ;TEST FOR CTRL G
016746  105337  016770                  DECB    10$                     ;FINISHED DELAY ?
016752  001356                          BNE     1$                      ;BR IF NOT
016754  013777  001472  162506  4$:     MOV     VCINT1,@VCINT           ;RESET INTR. VECTOR
016762  005077  162504                  CLR     @VCINT1                 ;
016766  000205                          RTS     R5                      ;EXIT

016770  000000          10$:    0
016772  000002          5$:     RTI                                     ;CONTINUE


                                        ;LOAD MAP WITH TWO WORDS OF THE SAME VALUE
                                        ;       INTO THE ENTIRE BITMAP STARTING AT LOC. 0

016774  010046          LODSEQ: MOV     R0,-(SP)                        ;SAVE R0
016776  010146                  MOV     R1,-(SP)                        ;SAVE R1
017000  005077  162456          CLR     @VGPC                   ;ENSURE CLEAR PC
017004  013701  001542          MOV     NUMPIX,R1               ;LOAD COUNTER
017010  006201                  ASR     R1                      ;ADJUST COUNT

017012  005000          1$:     CLR     R0                      ;START WITH PIXEL VALUE OF 0
017014  105777  162440  2$:     TSTB    @VGCSR                  ;ENSURE MAP IS READY
017020  100375                  BPL     2$
017022  010077  162436          MOV     R0,@VGBUF               ;LOAD A PIXEL WORD
017026  105777  162426  3$:     TSTB    @VGCSR                  ;ENSURE MAP IS READY
017032  100375                  BPL     3$
017034  010077  162424          MOV     R0,@VGBUF               ;LOAD 2ND PIXEL WORD
017040  005301                  DEC     R1                      ;FINISHED ALL PIXELS ?
017042  100404                  BMI     4$                      ;BR IF FINISHED
017044  062700  010421          ADD     #10421,R0               ;UPDATE PIXEL DATA TO NEXT L.U.T. ADDRESS
017050  103361                  BCC     2$                      ;BR IF MORE ADDRESSES TO LOAD
017052  000757                  BR      1$                      ;BR TO RESET PIXEL DATA

017054  012601          4$:     MOV     (SP)+,R1                ;RESTORE R1
017056  012600                  MOV     (SP)+,R0                ;RESTORE R0
017060  000207                  RTS     PC                      ;EXIT
```

```
                                .SBTTL  SUBROUTINE TO LOAD THE L.U.T. WITH THE VALUE OF A TABLE
                                ;
017062  010046                LODSTA: MOV     R0,-(SP)                ;SAVE R0
017064  012537  017136                MOV     (R5)+,10$               ;GET STARTING LUT ADDRESS
017070  012500                        MOV     (R5)+,R0                ;GET TABLE ADDRESS
017072  012737  000020  017140        MOV     #16.,11$                ;LOAD COUNTER
                                ;
017100  113737  017136  017143        MOVB    10$,12$+1               ;LOAD HIGH BYTE WITH LUT ADDRESS
017106  112037  017142        1$:     MOVB    (R0)+,12$               ;LOAD VALUE INTO LOW BYTE
017112  013777  017142  162346        MOV     12$,@VGCOLR             ;LOAD THE TABLE
017120  105237  017143                INCB    12$+1                   ;UPDATE THE LUT POINTER
017124  005337  017140                DEC     11$                     ;FINISHED ALL L.U.T. ADDRESSES
017130  001366                        BNE     1$                      ;BR IF NOT
017132  012600                        MOV     (SP)+,R0                ;RESTORE R0
017134  000205                        RTS     R5                      ;EXIT
                                ;
017136  000000                10$:    0
017140  000020                11$:    16.                             ;COUNTER OF L.U.T.
017142  000000                12$:    0                               ;TEMP LOCATION
                                ;
017144  000000                YADDS:  0
017146  000000                XADDS:  0
017150  000000                SET:    0
017152  000000                OVRAL:  0
017154  051526  030126  026461  MSGTXT: .ASCII  \VSV01-DYNAMIC-CURSOR-ADDRESS-TEST-\
017162  054504  040516  044515
017170  026503  052503  051522
017176  051117  040455  042104
017204  042522  051523  052055
017212  051505  026524
017216  042055  043511  052111          .ASCII  \-DIGITAL-EQUIPMENT-CORPORATION\
017224  046101  042455  052521
017232  050111  042515  052116
017240  041455  051117  047520
017246  040522  044524  047117
017254  100000                MSGTND: BIT15
```

```
2776                                        .SBTTL  HOME THE DISPLAY SUBROUTINE
2777  017256  004537  016666          HOME:   JSR     R5,DELAY                ;WAIT FOR VERT SYNC
2778  017262  100001                          BIT15!BIT0
2779  017264  012737  177435  024330        MOV     #177435,BUFFER          ;LOAD "HOME" AND TERMINATOR
2780  017272  004737  016032                 JSR     PC,XPRNT                ;EXECUTE
2781  017276  005777  162144          1$:     TST     @VCCSR                  ;WAIT FOR CHAR READY
2782  017302  100375                          BPL     1$
2783  017304  000207                          RTS     PC                      ;RETURN
2784
2785                                        .SBTTL  DISPLAY BIT MAP # AND TEST #
2786  017306  032777  010000  161624  TESTID: BIT     #BIT12,@SWR             ;TEST IF INHIBIT DELAY IS SET
2787  017314  001071                          BNE     3$                      ;BR AND DONT DISPLAY THIS STUFF
2788  017316  112737  000060  017525        MOVB    #'0,MAPIND-1            ;LOAD 1 DIGIT VALUE
2789  017324  013737  001546  017502        MOV     COAXSW,10$              ;SAVE COAX SWITCH VALUE
2790  017332  006237  017502          4$:     ASR     10$                     ;CONVERT TO A ASCII DIGIT
2791  017336  103403                          BCS     5$
2792  017340  105237  017525                 INCB    MAPIND-1
2793  017344  000772                          BR      4$
2794  017346  012737  000060  017502  5$:     MOV     #'0,10$                 ;LOAD MAP # INDICATOR
2795  017354  063737  001204  017502        ADD     $UNIT,10$               ;ADD CURRENT MAP I.D.
2796  017362  113737  017502  017526        MOVB    10$,MAPIND              ;SAVE IN A MESSAGE
2797  017370  013700  001102          1$:     MOV     $TSTNM,R0               ;LOAD TEST #
2798  017374  112737  000060  017550        MOVB    #'0,TSTIND              ;LOAD 0
2799  017402  032700  000100                 BIT     #BIT6,R0                ;TEST IF TEST # > 100
2800  017406  001402                          BEQ     2$
2801  017410  105237  017550                 INCB    TSTIND                  ;MAKE TEST # 100
2802  017414  010001                  2$:     MOV     R0,R1
2803  017416  042701  177770                 BIC     #177770,R1              ;MASK LOWER BITS
2804  017422  062701  000060                 ADD     #60,R1                  ;MAKE IT ASCII
2805  017426  110137  017552                 MOVB    R1,TSTIND+2             ;LOAD MESSAGE
2806  017432  006200                          ASR     R0
2807  017434  006200                          ASR     R0
2808  017436  006200                          ASR     R0
2809  017440  042700  177770                 BIC     #177770,R0              ;MASK LOWER BITS
2810  017444  062700  000060                 ADD     #60,R0                  ;UPDATE TO ASCII
2811  017450  110037  017551                 MOVB    R0,TSTIND+1             ;UPDATE MESSAGE
2812  017454  112777  000014  161772        MOVB    #14,@VCPOS              ;LOAD X POSITION
2813  017462  112777  000030  161766        MOVB    #24.,@VCPOSH            ;LOAD Y POSITION
2814  017470  012700  017504                 MOV     #BRIAN,R0               ;LOAD MESSAGE POINTER
2815  017474  004737  016036                 JSR     PC,XPRNTA               ;DISPLAY MESSAGE
2816  017500  000207                  3$:     RTS     PC                      ;EXIT
2817  017502  000000                  10$:    0
2818  017504  042524  052123  047111  BRIAN:  .ASCII  /TESTING BIT MAP #/
2819  017512  020107  044502  020124
2820  017520  040515  020120     043
2821  017525     060                  .BYTE   60                      ;1 DIGIT OF MAP #
2822  017526     060                  MAPIND: .BYTE   60              ;2 DIGIT OF MAP #
2823  017527     040  020040  052522          .ASCII  /   RUNNING TEST #/
2824  017534  047116  047111  020107
2825  017542  042524  052123  021440
2826  017550     060     060     060  TSTIND: .BYTE   60,60,60,377
2827  017553     377
2828                                        .EVEN
```

```
2829
2830  017554  012777  000202  161666  LDLIN:  MOV     #202,@VCHRLO            ;LOAD X CROSS HAIR LINE
2831  017562  052777  010000  161656          BIS     #BIT12,@VCCSR           ;ENABLE X CROSS HAIR
2832  017570  000207                          RTS     PC                      ;EXIT
2833
2834  017572  050000                  BYTST:  BIT14!BIT12
2835  017574  120000                          BIT15!BIT13
2836  017576  050000                          BIT14!BIT12
2837  017600  120000                          BIT15!BIT13
2838  017602  000000                          0
2839
2840                                  .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
2841
2842                                  ;;*****************************************************************
2843                                  ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
2844                                  ;*WITH A RANGE OF 0 TO 2(+33)-1.
2845                                  ;*CALL:
2846                                  ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
2847                                  ;*      RETURN                  ;;RETURN HERE THE RANDOM
2848                                  ;*                              ;;NUMBER WILL BE IN
2849                                  ;*                              ;;$HINUM,$LONUM
2850
2851  017604                  $RAND:
2852  017604  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2853  017606  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2854  017610  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2855  017612  013700  017704                  MOV     $LONUM,R0       ;;SET R0 WITH LOW
2856  017616  013701  017702                  MOV     $HINUM,R1       ;;SET R1 WITH HIGH
2857  017622  012702  177771                  MOV     #-7,R2          ;;SET SHIFT COUNT
2858  017626  006300                  1$:     ASL     R0              ;;SHIFT R0 LEFT AND
2859  017630  006101                          ROL     R1              ;;ROTATE CARRY INTO R1 AND
2860  017632  005202                          INC     R2              ;;CHECK FOR DONE
2861  017634  001374                          BNE     1$              ;;CONTINUE SHIFT LOOP
2862  017636  063700  017704                  ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
2863  017642  005501                          ADC     R1              ;;PROPOGATE CARRY
2864  017644  063701  017702                  ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
2865  017650  062700  001057                  ADD     #1057,R0        ;;ADD LOW CONSTANT
2866  017654  005501                          ADC     R1              ;;PROPOGATE CARRY
2867  017656  062701  047401                  ADD     #47401,R1       ;;ADD HIGH CONSTANT
2868  017662  010037  017704                  MOV     R0,$LONUM       ;;SAVE R0
2869  017666  010137  017702                  MOV     R1,$HINUM       ;;SAVE R1
2870  017672  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2871  017674  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2872  017676  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2873  017700  000207                          RTS     PC              ;;RETURN
2874  017702  176543          $HINUM: .WORD   176543
2875  017704  123456          $LONUM: .WORD   123456
```

```
                                        .SBTTL  SCOPE HANDLER ROUTINE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW14=1         LOOP ON TEST
                                        ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT

017706                                  $SCOPE:
017706  104407                                 CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
017710  104407                                 CKSWR
017712  032777  040000  161220  1$:      BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
017720  001040                                 BNE     $OVER           ;;YES IF SW14=1
                                        ;*****START OF CODE FOR THE XOR TESTER*****
017722  000416                          $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
017724  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
017730  012737  017750  000004                 MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
017736  005737  177060                        TST     @#177060        ;;TIME OUT ON XOR?
017742  012637  000004                        MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
017746  000414                                 BR      $SVLAD          ;;GO TO THE NEXT TEST
017750  022626                          5$.     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
017752  012637  000004                        MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
017756  000421                                 BR      $OVER           ;;LOOP ON THE PRESENT TEST
017760                                  6$:;*****END OF CODE FOR THE XOR TESTER*****
017760  032777  000400  161152                 BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
017766  001404                                 BEQ     $SVLAD          ;;BR IF NO
017770  127737  161144  001102                 CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
017776  001411                                 BEQ     $OVER           ;;BR IF YES
020000  105237  001102                  $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
020004  113737  001102  001176                 MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
020012  011637  001106                        MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
020016  105037  001103                        CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
020022  013777  001102  161112  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
020030  013716  001106                        MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
020034  000002                                 RTI                     ;;FIXES PS
```

```
                                .SBTTL  ERROR HANDLER ROUTINE
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO $ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

020036                          $ERROR:
020036  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
020040  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
020044  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
020046  013777  001102  161066          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
020054  005237  001112                  INC     $ERTTL          ;;INC THE ERROR COUNT
020060  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
020064  162737  000002  001116          SUB     #2,$ERRPC
020072  117737  161020  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
020100  032777  020000  161032          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
020106  001004                          BNE     20$             ;;SKIP TYPEOUTS
020110  004737  020174                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
020114  104401  001167                  TYPE    ,$CRLF
020120                          20$:
020120  122737  000001  001212          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
020126  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
020130  113737  001114  020142          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
020136  004737  021522                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
020142     000          21$:    .BYTE   0
020143     000                  .BYTE   0
020144  000777          22$:    BR      22$             ;;APT ERROR LOOP
020146  005777  160786          2$:     TST     @SWR            ;;HALT ON ERROR
020152  100002                          BPL     3$              ;;SKIP IF CONTINUE
020154  000000                          HALT                    ;;HALT ON ERROR!
020156  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
020160                          3$:
020160  022737  003304  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
020166  001001                          BNE     6$              ;;BRANCH IF NO
020170  000000                          HALT                    ;;YES
020172                          6$:
020172  000002                          RTI                     ;;RETURN
```

B06


```
                                        .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

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

        020174                          $ERRTYP:
        020174  104401  001167                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
        020200  010046                          MOV     R0,-(SP)        ;;SAVE R0
        020202  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
        020204  153700  001114                  BISB    @#$ITEMB,R0
        020210  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                        ;;TYPE THE PC OF THE ERROR
        020212  013746  001116                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;ERROR ADDRESS
        020216  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        020220  000426                          BR      6$              ;;GET OUT
        020222  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
        020224  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
        020226  006300                          ASL     R0
        020230  006300                          ASL     R0
        020232  062700  001256                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
        020236  012037  020246                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
        020242  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
        020244  104401                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
        020246  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
        020250  104401  001167                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
        020254  012037  020264          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
        020260  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
        020262  104401                          TYPE                    ;;TYPE THE "DATA HEADER"
        020264  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
        020266  104401  001167                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
        020272  011000                  5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
        020274  001004                          BNE     7$              ;;GO TYPE THE DATA
        020276  012600                  6$:     MOV     (SP)+,R0        ;;RESTORE R0
        020300  104401  001167                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
        020304  000207                          RTS     PC              ;;RETURN
        020306                          7$:
        020306  013046                          MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
        020310  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
        020312  005710                          TST     (R0)            ;;IS THERE ANOTHER NUMBER?
        020314  001770                          BEQ     6$              ;;BR IF NO
        020316  104401  020324                  TYPE    ,8$             ;;TYPE TWO(2) SPACES
        020322  000771                          BR      7$              ;;LOOP
        020324  020040     000          8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
                020330                          .EVEN
```

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

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

020330  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
020334  116637  000001  020553          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
020342  112637  020555          MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
020346  062716  000002          ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
020352  000406                  BR      $TYPON
020354  112737  000001  020553  $TYPOC: MOVB   #1,$OFILL       ;;SET THE ZERO FILL SWITCH
020362  112737  000006  020555          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
020370  112737  000005  020552  $TYPON: MOVB   #5,$OCNT         ;;SET THE ITERATION COUNT
020376  010346                  MOV     R3,-(SP)        ;;SAVE R3
020400  010446                  MOV     R4,-(SP)        ;;SAVE R4
020402  010546                  MOV     R5,-(SP)        ;;SAVE R5
020404  113704  020555          MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
020410  005404                  NEG     R4
020412  062704  000006          ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
020416  110437  020554          MOVB    R4,$OMODE       ;;SAVE IT FOR USE
020422  113704  020553          MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
020426  016605  000012          MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
020432  005003                  CLR     R3              ;;CLEAR THE OUTPUT WORD
020434  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
020436  000404                  BR      3$              ;;GO DO MSB
020440  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
020442  006105                  ROL     R5
020444  006105                  ROL     R5
020446  010503                  MOV     R5,R3
020450  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
020452  105337  020554          DECB    $OMODE          ;;TYPE THIS DIGIT?
020456  100016                  BPL     7$              ;;BR IF NO
020460  042703  177770          BIC     #177770,R3      ;;GET RID OF JUNK
020464  001002                  BNE     4$              ;;TEST FOR 0
020466  005704                  TST     R4              ;;SUPPRESS THIS 0?
```

```
020470  001403                          BEQ     5$              ;;BR IF YES
020472  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
020474  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
020500  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
020504  110337  020550                  MOVB    R3,8$           ;;SAVE FOR TYPING
020510  104401  020550                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
020514  105337  020552          7$:     DECB    $OCNT           ;;COUNT BY 1
020520  003347                          BGT     2$              ;;BR IF MORE TO DO
020522  002402                          BLT     6$              ;;BR IF DONE
020524  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
020526  000744                          BR      2$              ;;GO DO THE LAST DIGIT
020530  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
020532  012604                          MOV     (SP)+,R4        ;;RESTORE R4
020534  012603                          MOV     (SP)+,R3        ;;RESTORE R3
020536  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
020544  012616                          MOV     (SP)+,(SP)
020546  000002                          RTI                     ;;RETURN
020550     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
020551     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
020552     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
020553     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
020554  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                ;;*******************************************************************
                                ;;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                ;;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                ;;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                ;;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                ;;*REPLACED WITH SPACES.
                                ;;*CALL:
                                ;;*       MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                ;;*       TYPDS                   ;;GO TO THE ROUTINE

020556                          $TYPDS:
020556  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
020560  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
020562  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
020564  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
020566  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
020570  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
020574  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
020600  100004                          BPL     1$              ;;BR IF INPUT IS POS.
020602  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
020604  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
020612  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
020614  012703  020772                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
020620  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
020624  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
020626  016001  020762                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
020632  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
020634  002402                          BLT     4$              ;;BR IF DONE
020636  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
020640  000774                          BR      3$
020642  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
020644  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
```

```
 3121  020646  001002                        BNE     5$              ;;FALL THROUGH IF 0
 3122  020650  105716                        TSTB    (SP)            ;;STILL DOING LEADING 0'S?
 3123  020652  100407                        BMI     7$              ;;BR IF YES
 3124  020654  106316                5$:     ASLB    (SP)            ;;MSD?
 3125  020656  103003                        BCC     6$              ;;BR IF NO
 3126  020660  116663  000001  177777        MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
 3127  020666  052702  000060        6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
 3128  020672  052702  000040        7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 3129  020676  110223                        MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 3130  020700  005720                        TST     (R0)+           ;;JUST INCREMENTING
 3131  020702  020027  000010                CMP     R0,#10          ;;CHECK THE TABLE INDEX
 3132  020706  002746                        BLT     2$              ;;GO DO THE NEXT DIGIT
 3133  020710  003002                        BGT     8$              ;;GO TO EXIT
 3134  020712  010502                        MOV     R5,R2           ;;GET THE LSD
 3135  020714  000764                        BR      6$              ;;GO CHANGE TO ASCII
 3136  020716  105726                8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 3137  020720  100003                        BPL     9$              ;;BR IF NO
 3138  020722  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 3139  020730  105013                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 3140  020732  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
 3141  020734  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
 3142  020736  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 3143  020740  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 3144  020742  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 3145  020744  104401  020772                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 3146  020750  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 3147  020756  012616                        MOV     (SP)+,(SP)
 3148  020760  000002                        RTI                     ;;RETURN TO USER
 3149  020762  023420                $DTBL:  10000.
 3150  020764  001750                        1000.
 3151  020766  000144                        100.
 3152  020770  000012                        10.
 3153  020772  000000                $DBLK:  .BLKW   4
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;**********************************************************************
                                        ;POWER DOWN ROUTINE
021002  012737  021146  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
021010  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
021016  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
021020  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
021022  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
021024  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
021026  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
021030  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
021032  017746  160102                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
021036  010637  021152                  MOV     SP,$SAVR6       ;;SAVE SP
021042  012737  021054  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
021050  000000                          HALT
021052  000776                          BR      .-2             ;;HANG UP

                                        ;;**********************************************************************
                                        ;POWER UP ROUTINE
021054  012737  021146  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
021062  013706  021152                  MOV     $SAVR6,SP       ;;GET SP
021066  005037  021152                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
021072  005237  021152          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
021076  001375                          BNE     1$              ;;OF  WORD
021100  012677  160034                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
021104  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
021106  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
021110  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
021112  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
021114  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
021116  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
021120  012737  021002  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
021126  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
021134  104401                          TYPE                    ;;REPORT THE POWER FAILURE
021136  021154                  $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
021140  012716                          MOV     (PC)+,(SP)      ;;RESTART AT BEGIN
021142  001572                  $PWRAD: .WORD   BEGIN           ;;RESTART ADDRESS
021144  000002                          RTI
021146  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
021150  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
021152  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
021154  005015  042522  052123  PWRMSG: .ASCIZ  <15><12>/RESTARTING AFTER A POWER FAILURE/<15><12>
021162  051101  044524  043516
021170  040440  052106  051105
021176  040440  050040  053517
021204  051105  043040  044501
021212  052514  042522  005015
021220     000
            021222                      .EVEN
```

```
                                        .SBTTL  TYPE ROUTINE

                                        ;;*****************************************************************
                                        ;;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                        ;;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;;*
                                        ;;*CALL:
                                        ;;*1) USING A TRAP INSTRUCTION
                                        ;;*       TYPE    ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                        ;;*OR
                                        ;;*       TYPE
                                        ;;*       MESADR
                                        ;;*
 021222  105737  001157                 $TYPE:  TSTB    $TPFLG           ;;IS THERE A TERMINAL?
 021226  100002                                 BPL     1$               ;;BR IF YES
 021230  000000                                 HALT                     ;;HALT HERE IF NO TERMINAL
 021232  000430                                 BR      3$               ;;LEAVE
 021234  010046                         1$:     MOV     R0,-(SP)         ;;SAVE R0
 021236  017600  000002                         MOV     @2(SP),R0        ;;GET ADDRESS OF ASCIZ STRING
 021242  122737  000001  001212                 CMPB    #APTENV,$ENV     ;;RUNNING IN APT MODE
 021250  001011                                 BNE     62$              ;;NO,GO CHECK FOR APT CONSOLE
 021252  132737  000100  001213                 BITB    #APTSPOOL,$ENVM  ;;SPOOL MESSAGE TO APT
 021260  001405                                 BEQ     62$              ;;NO,GO CHECK FOR CONSOLE
 021262  010037  021272                         MOV     R0,61$           ;;SETUP MESSAGE ADDRESS FOR APT
 021266  004737  021512                         JSR     PC,$ATY3         ;;SPOOL MESSAGE TO APT
 021272  000000                         61$:    .WORD   0                ;;MESSAGE ADDRESS
 021274  132737  000040  001213         62$:    BITB    #APTCSUP,$ENVM   ;;APT CONSOLE SUPPRESSED
 021302  001003                                 BNE     60$              ;;YES,SKIP TYPE OUT
 021304  112046                         2$:     MOVB    (R0)+,-(SP)      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 021306  001005                                 BNE     4$               ;;BR IF IT ISN'T THE TERMINATOR
 021310  005726                                 TST     (SP)+            ;;IF TERMINATOR POP IT OFF THE STACK
 021312  012600                         60$:    MOV     (SP)+,R0         ;;RESTORE R0
 021314  062716  000002                 3$:     ADD     #2,(SP)          ;;ADJUST RETURN PC
 021320  000002                                 RTI                      ;;RETURN
 021322  122716  000011                 4$:     CMPB    #HT,(SP)         ;;BRANCH IF <HT>
 021326  001430                                 BEQ     8$
 021330  122716  000200                         CMPB    #CRLF,(SP)       ;;BRANCH IF NOT <CRLF>
 021334  001006                                 BNE     5$
 021336  005726                                 TST     (SP)+            ;;POP  <CR><LF> EQUIV
 021340  104401                                 TYPE                     ;;TYPE A CR AND LF
 021342  001167                                 $CRLF
 021344  105037  021500                         CLRB    $CHARCNT         ;;CLEAR CHARACTER COUNT
 021350  000755                                 BR      2$               ;;GET NEXT CHARACTER
 021352  004737  021434                 5$:     JSR     PC,$TYPEC        ;;GO TYPE THIS CHARACTER
 021356  123726  001156                 6$:     CMPB    $FILLC,(SP)+     ;;IS IT TIME FOR FILLER CHARS.?
 021362  001350                                 BNE     2$               ;;IF NO GO GET NEXT CHAR.
 021364  013746  001154                         MOV     $NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
                                                                         ;;AND THE NULL CHAR.
 021370  105366  000001                 7$:     DECB    1(SP)            ;;DOES A NULL NEED TO BE TYPED?
 021374  002770                                 BLT     6$               ;;BR IF NO--GO POP THE NULL OFF OF STACK
 021376  004737  021434                         JSR     PC,$TYPEC        ;;GO TYPE A NULL
 021402  105337  021500                         DECB    $CHARCNT         ;;DO NOT COUNT AS A COUNT
```

```
3261  021406  000772                        BR      7$              ;;LOOP
3262
3263                                  ;HORIZONTAL TAB PROCESSOR
3264
3265  021410  112716  000040          6$:   MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
3266  021414  004737  021434          9$:   JSR     PC,$TYPEC       ;;TYPE A SPACE
3267  021420  132737  000007  021500        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
3268  021426  001372                        BNE     9$              ;;TAB STOP
3269  021430  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
3270  021432  000724                        BR      2$              ;;GET NEXT CHARACTER
3271  021434  105777  157510          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3272  021440  100375                        BPL     $TYPEC
3273  021442  116677  000002  157502        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3274  021450  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
3275  021456  001003                        BNE     1$              ;;BRANCH IF NO
3276  021460  105037  021500                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
3277  021464  000406                        BR      $TYPEX          ;;EXIT
3278  021466  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
3279  021474  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
3280  021476  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
3281  021500  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
3282  021502  000207          $TYPEX: RTS     PC
3283
3284                                  .SBTTL  APT COMMUNICATIONS ROUTINE
3285
3286                                  ;;**************************************************************
3287  021504  112737  000001  021750  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
3288  021512  112737  000001  021746  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
3289  021520  000403                        BR      $ATYC
3290  021522  112737  000001  021750  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
3291  021530                          $ATYC:
3292  021530  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3293  021532  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3294  021534  105737  021746                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
3295  021540  001450                        BEQ     5$              ;;IF NOT:  BR
3296  021542  122737  000001  001212        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
3297  021550  001031                        BNE     3$              ;;IF NOT:  BR
3298  021552  132737  000100  001213        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
3299  021560  001425                        BEQ     3$              ;;IF NOT:  BR
3300  021562  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
3301  021566  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
3302  021574  005737  001172          1$:   TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
3303  021600  001375                        BNE     1$              ;;IF NOT:  WAIT
3304  021602  010037  001206                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
3305  021606  105720                  2$:   TSTB    (R0)+           ;;FIND END OF MESSAGE
3306  021610  001376                        BNE     2$
3307  021612  163700  001206                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
3308  021616  006200                        ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
3309  021620  010037  001210                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
3310  021624  012737  000004  001172        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
3311  021632  000413                        BR      5$
3312  021634  017637  000004  021660  3$:   MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
3313  021642  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
3314  021650  013746  177776                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
3315  021654  004737  021222                JSR     PC,$TYPE        ;;CALL TYPE MACRO
3316  021660  000000                  4$:   .WORD   0
```

```
 3317  021662                          5$:
 3318  021662  105737  021750          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
 3319  021666  001416                          BEQ     12$             ;;IF NOT:  BR
 3320  021670  005737  001212                  TST     $ENV            ;;RUNNING UNDER APT?
 3321  021674  001413                          BEQ     12$             ;;IF NOT:  BR
 3322  021676  005737  001172          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
 3323  021702  001375                          BNE     11$             ;;IF NOT:  WAIT
 3324  021704  017637  000004  001174          MOV     @4(SP),$FATAL   ;;GET ERROR #
 3325  021712  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 3326  021720  005237  001172                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
 3327  021724  105037  021750          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
 3328  021730  105037  021747                  CLRB    $LFLG           ;;CLEAR LOG FLAG
 3329  021734  105037  021746                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
 3330  021740  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 3331  021742  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 3332  021744  000207                          RTS     PC              ;;RETURN
 3333  021746  000             $MFLG:  .BYTE   0               ;;MESSG. FLAG
 3334  021747  000             $LFLG:  .BYTE   0               ;;LOG FLAG
 3335  021750  000             $FFLG:  .BYTE   0               ;;FATAL FLAG
 3336  021752                          .EVEN
 3337  000200                  APTSIZE=200
 3338  000001                  APTENV=001
 3339  000100                  APTSPOOL=100
 3340  000040                  APTOSUF=040
```

```
                                .SBTTL  TTY INPUT ROUTINE
                                ;;*****************************************************************
                                .ENABL  LSB
                                ;;*****************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                ;*WHEN OPERATING IN TTY FLAG MODE.
021752  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
021760  001074                         BNE     15$             ;;BRANCH IF NO
021762  105777  157156                 TSTB    @$TKS           ;;CHAR THERE?
021766  100071                         BPL     15$             ;;IF NO, DON'T WAIT AROUND
021770  117746  157152                 MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
021774  042716  177600                 BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
022000  022726  000007                 CMP     #7,(SP)+        ;;IS IT A CONTROL G?
022004  001062                         BNE     15$             ;;NO, RETURN TO USER
022006  123727  001134  000001         CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
022014  001456                         BEQ     15$             ;;BRANCH IF YES

022016  104401  022477                 TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
022022  104401  022504         $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
022026  013746  000176                 MOV     $SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
022032  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
022034  104401  022515                 TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
022040  005046                 19$:    CLR     -(SP)           ;;CLEAR COUNTER
022042  005046                         CLR     -(SP)           ;;THE NEW SWR
022044  105777  157074         7$:     TSTB    @$TKS           ;;CHAR THERE?
022050  100375                         BPL     7$              ;;IF NOT TRY AGAIN

022052  117746  157070                 MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
022056  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII



022062  021627  000025         9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
022066  001005                         BNE     10$             ;;BRANCH IF NOT
022070  104401  022472                 TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
022074  062706  000006         20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
022100  000757                         BR      19$             ;;LET'S TRY IT AGAIN

022102  021627  000015         10$:    CMP     (SP),#15        ;;IS IT A <CR>?
022106  001022                         BNE     16$             ;;BRANCH IF NO
022110  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
022114  001403                         BEQ     11$             ;;BRANCH IF YES
022116  016677  000002  157014         MOV     2(SP),@SWR      ;;SAVE NEW SWR
022124  062706  000006         11$:    ADD     #6,SP           ;;CLEAR UP STACK
022130  104401  001167         14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
022134  123727  001135  000001         CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
022142  001003                         BNE     15$             ;;BRANCH IF NOT
022144  012777  000100  156772         MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
022152  000002                 15$:    RTI                     ;;RETURN
022154  004737  021434         16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
022160  021627  000060                 CMP     (SP),#60        ;;CHAR < 0?
```

```
 3397  022164  002420                          BLT     18$             ;;BRANCH IF YES
 3398  022166  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
 3399  022172  003015                          BGT     18$             ;;BRANCH IF YES
 3400  022174  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
 3401  022200  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
 3402  022204  001403                          BEQ     17$             ;;BRANCH IF YES
 3403  022206  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
 3404  022210  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
 3405  022212  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
 3406  022214  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
 3407  022220  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
 3408  022224  000707                          BR      7$              ;;GET THE NEXT ONE
 3409  022226  104401  001166          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
 3410  022232  000720                          BR      20$             ;;SIMULATE CONTROL-U
 3411                                  .DSABL  LSB
 3412
 3413
 3414                                  ;;**************************************************************
 3415                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 3416                                  ;*CALL:
 3417                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
 3418                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 3419                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
 3420                                  ;
 3421
 3422  022234  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
 3423  022236  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
 3424  022244  105777  156674          1$:     TSTB    @$TKS           ;;WAIT FOR
 3425  022250  100375                          BPL     1$              ;;A CHARACTER
 3426  022252  117766  156670  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
 3427  022260  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
 3428  022266  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
 3429  022274  001013                          BNE     3$              ;;BRANCH IF NO
 3430  022276  105777  156642          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
 3431  022302  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
 3432  022304  117746  156636                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
 3433  022310  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 3434  022314  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 3435  022320  001366                          BNE     2$              ;;IF NOT DISCARD IT
 3436  022322  000750                          BR      1$              ;;YES, RESUME
 3437  022324  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
 3438  022332  002407                          BLT     4$              ;;BRANCH IF YES
 3439  022334  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
 3440  022342  003003                          BGT     4$              ;;BRANCH IF YES
 3441  022344  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
 3442  022352  000002                  4$:     RTI                     ;;GO BACK TO USER
 3443                                  ;;**************************************************************
 3444                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 3445                                  ;*CALL:
 3446                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 3447                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 3448                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 3449
 3450  022354  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 3451  022356  012703  022462          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
 3452  022362  022703  022472          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
```

```
 3453  022366  101405                          BLOS    4$              ;;BR IF YES
 3454  022370  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 3455  022372  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
 3456  022374  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
 3457  022400  001003                          BNE     3$              ;;SKIP IF NOT
 3458  022402  104401  001166          4$:     TYPE    ,$QUES          ;;TYPE A '?'
 3459  022406  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
 3460  022410  111337  022460          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
 3461  022414  104401  022460                  TYPE    ,9$
 3462  022420  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
 3463  022424  001356                          BNE     2$              ;;LOOP IF NOT RETURN
 3464  022426  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
 3465  022432  104401  001170                  TYPE    ,$LF            ;;TYPE A LINE FEED
 3466  022436  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 3467  022440  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 3468  022442  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 3469  022450  012766  022462  000004          MOV     #$TTYIN,4(SP)
 3470  022456  000002                          RTI                     ;;RETURN
 3471  022460     000          9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
 3472  022461     000                  .BYTE   0                       ;;TERMINATOR
 3473  022462  000010          $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
 3474  022472  052536  005015  000   $CNTLU: .ASCIZ  /^U/<15><12>     ;;CONTROL "U"
 3475  022477     136  006507  000012$CNTLG: .ASCIZ  /^G/<15><12>     ;;CONTROL "G"
 3476  022504  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
 3477  022512  020075     000
 3478  022515     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
 3479  022522  036440  000040
 3480                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
 3481
 3482                                  ;;**************************************************************
 3483                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
 3484                                  ;*CHANGE IT TO BINARY.
 3485                                  ;*CALL:
 3486                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
 3487                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
 3488                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
 3489
 3490  022526  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
 3491  022530  016666  000004  000002  MOV     4(SP),2(SP)     ;;INPUT NUMBER
 3492  022536  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 3493  022540  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 3494  022542  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 3495  022544  104411          1$:     RDLIN                   ;;READ AN ASCIZ LINE
 3496  022546  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
 3497  022550  005001                  CLR     R1              ;;CLEAR DATA WORD
 3498  022552  005002                  CLR     R2
 3499  022554  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
 3500  022556  001412                  BEQ     3$              ;;IF ZERO GET OUT
 3501  022560  006301                  ASL     R1              ;;*2
 3502  022562  006102                  ROL     R2
 3503  022564  006301                  ASL     R1              ;;*4
 3504  022566  006102                  ROL     R2
 3505  022570  006301                  ASL     R1              ;;*8
 3506  022572  006102                  ROL     R2
 3507  022574  042716  177770          BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
 3508  022600  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
```

```
3509  022602  000764                      BR      2$              ;;LOOP
3510  022604  005726                3$:   TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
3511  022606  010166  000012              MOV     R1,12(SP)       ;;SAVE THE RESULT
3512  022612  010237  022626              MOV     R2,$HIOCT
3513  022616  012602                      MOV     (SP)+,R2        ;;POP STACK INTO R2
3514  022620  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
3515  022622  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
3516  022624  000002                      RTI                     ;;RETURN
3517  022626  000000              $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
```

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

022630  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
022632  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
022636  005740                          TST     -(R0)           ;;BACKUP BY 2
022640  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
022642  006300                          ASL     R0              ;;POSITION FOR INDEXING
022644  016000  022664                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
022650  000200                          RTS     R0              ;;GO TO ROUTINE

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

022652  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
022654  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
022662  000002                          RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                        ;       -------
022664  022652                  $TRPAD: .WORD   $TRAP2
022666  021222                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
022670  020354                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
022672  020330                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
022674  020370                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
022676  020556                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

022700  022022                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

022702  021752                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
022704  022234                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
022706  022354                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
022710  022526                          $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
```

```
                                                .SBTTL  ASCII MESSAGES
        022712  025015  046412  026504  TITLE:  .ASCIZ  <15><12><12>/MD-11-DZVSE-A0     AUG. 1976<15><12>
        022720  030461  042055  053132
        022726  042523  040455  034460
        022734  052501  027107  030440
        022742  033471  006466  000012
        022750     015     012          VTHDR1: .BYTE   15,12
        022752  052126  030126  020061          .ASCIZ  /VTV01 AT ADDRESS /
        022760  052101  040440  042104
        022766  042522  051523  000040
        022774  044040  051501  000040  VTHDR2: .ASCIZ  / HAS /
        023002  041040  052111  046440  VTHDR3: .ASCIZ  / BIT MAP(S) INSTALLED/<15><12>
        023010  050101  051450  020051
        023016  047111  052123  046101
        023024  042514  006504  000012
        023032  047516  041040  051525  EM1:    .ASCIZ  /NO BUS RESPONSE FROM A CHARACTER BUS REGISTER  ** FATAL ERROR **/
        023040  051040  051505  047520
        023046  051516  020105  051106
        023054  046517  040440  041440
        023062  040510  040522  052103
        023070  051105  041040  051525
        023076  051040  043505  051511
        023104  042524  020122  025040
        023112  020052  040506  040524
        023120  020114  051105  047522
        023126  020122  025052     000
        023133     116  020117  052502  EM2:    .ASCIZ  /NO BUS RESPONSE FROM A SELECTED BIT MAP REGISTER/
        023140  020123  042522  050123
        023146  047117  042523  043040
        023154  047522  020115  020101
        023162  042523  042514  052103
        023170  042105  041040  052111
        023176  046440  050101  051040
        023204  043505  051511  042524
        023212  000122
        023214  044103  051101  041501  EM3:    .ASCIZ  /CHARACTER STATUS REGISTER ERROR/
        023222  042524  020122  052123
        023230  052101  051525  051040
        023236  043505  051511  042524
        023244  020122  051105  047522
        023252  000122
        023254  042526  052122  041511  EM4:    .ASCIZ  /VERTICAL SYNC ERROR/
        023262  046101  051440  047131
        023270  020103  051105  047522
        023276  000122
        023300  042526  052122  041511  EM5:    .ASCIZ  /VERTICAL SYNC INTERRUPT ERROR/
        023306  046101  051440  047131
        023314  020103  047111  042524
        023322  051122  050125  020124
        023330  051105  047522  000122
        023336  044502  020124  040515  EM6:    .ASCIZ  /BIT MAP STATUS REGISTER ERROR/
        023344  020120  052123  052101
        023352  051525  051040  043505
        023360  051511  042524  020122
        023366  051105  047522  000122
        023374  044502  020124  040515  EM7:    .ASCIZ  /BIT MAP P.C. REGISTER ERROR/
```

```
3618  023402  020120  027120  027103
3619  023410  051040  043505  051511
3620  023416  042524  020122  051105
3621  023424  047522  000122
3622  023430  044502  020124  040515  EM10:   .ASCIZ  /BIT MAP P.C. INCREMENT ERROR/
3623  023436  020120  027120  027103
3624  023444  044440  041516  042522
3625  023452  042515  052116  042440
3626  023460  051122  051117     003
3627  023465     127  051117  020104  EM11:   .ASCIZ  /WORD TEST OF PIXEL BIT 0 ERROR/
3628  023472  042524  052123  047440
3629  023500  020106  044520  042530
3630  023506  020114  044502  020124
3631  023514  020060  051105  047522
3632  023522  000122
3633  023524  054502  042524  052040  EM12:   .ASCIZ  /BYTE TEST OF PIXEL BIT 0 ERROR/
3634  023532  051505  020124  043117
3635  023540  050040  054111  046105
3636  023546  041040  052111  030040
3637  023554  042440  051122  051117
3638  023562     000
3639  023563     103  040510  040522  EM13:   .ASCIZ  /CHARACTER READY ERROR/
3640  023570  052103  051105  051040
3641  023576  040505  054504  042440
3642  023604  051122  051117     000
3643  023611     102  052111  046440  EM14:   .ASCIZ  /BIT MAP READY ERROR/
3644  023616  050101  051040  040505
3645  023624  054504  042440  051122
3646  023632  051117     000
3647  023635     120  042522  044526  EM16:   .ASCIZ  /PREVIOUSLY EXISTING BIT MAP DOES NOT EXIST NOW
3648  023642  052517  046123  020131
3649  023650  054105  051511  044524
3650  023656  043516  041040  052111
3651  023664  046440  050101  042040
3652  023672  042517  020123  047516
3653  023700  020124  054105  051511
3654  023706  020124  047516  000127
3655  023714  051120  053105  047511  EM17:   .ASCIZ  /PREVIOUSLY EXISTING VTVQ1 CONTROLLER DOES NOT EXIST NOW/
3656  023722  051525  054514  042440
3657  023730  044530  052123  047111
3658  023736  020107  052126  030126
3659  023744  020061  047503  052116
3660  023752  047522  046114  051105
3661  023760  042040  042517  020123
3662  023766  047516  020124  054105
3663  023774  051511  020124  047516
3664  024002  000127
3665  024004  051105  050122  004503  DH1:    .ASCIZ  /ERRPC  DEVICE ADDRESS THAT CAUSED A BUS TRAP/
3666  024012  042504  044526  042503
3667  024020  040440  042104  042522
3668  024026  051523  052040  040510
3669  024034  020124  040503  051525
3670  024042  042105  040440  041040
3671  024050  051525  052040  040522
3672  024056  000120
3673  024060  051105  050122  004503  DH3:    .ASCIZ  /ERRPC  VCCSR   GOOD    BAD/
```

```
 3674  024266  041526  051503  004522
 3675  024274  047507  042117  041011
 3676  024102  042101     000
 3677  024105     105  051122  041520  DH4:    .ASCIZ  /ERRPC  VCCSR   TIME1   TIME2
 3678  024112  053011  041503  051123
 3679  024120  052011  046511  030505
 3680  024126  052011  046511  031105
 3681  024134     000
 3682  024135     105  051122  041520  DH5:    .ASCIZ  /ERRPC  VCCSR/
 3683  024142  053011  041503  051123
 3684  024150     000
 3685  024151     105  051122  041520  DH6:    .ASCIZ  /ERRPC  VGCSR   GOOD    BAD/
 3686  024156  053011  041507  051123
 3687  024164  043411  047517  004504
 3688  024172  040502  000104
 3689  024176  051105  050122  004503  DH15:   .ASCIZ  /ERRPC  VGCSR/
 3690  024204  043526  051503  000122
 3691  024212  051105  050122  004503  DH16:   .ASCIZ  /ERRPC  VTV01/
 3692  024220  052126  030126  000061
 3693                                   .EVEN
 3694  024226  001116  001126  000000  DT1:    $ERRPC,$BDDAT,0
 3695  024234  001116  001446  001124  DT3:    $ERRPC,VCCSR,$GDDAT,$BDDAT,0
 3696  024242  001126  000000
 3697  024246  001116  001446  001510  DT4:    $ERRPC,VCCSR,SAVE0,SAVE2,0
 3698  024254  001514  000000
 3699  024260  001116  001446  000000  DT5:    $ERRPC,VCCSR,0
 3700  024266  001116  001460  001124  DT6:    $ERRPC,VGCSR,$GDDAT,$BDDAT,0
 3701  024274  001126  000000
 3702  024300  001116  001460  000000  DT15:   $ERRPC,VGCSR,0
 3703  024306  001116  001120  000000  DT16:   $ERRPC,$GDADR,0,0,0,0
 3704  024314  000000  000000  000000
 3705  024322     000     000     000  DF0:    .BYTE   0,0,0,0,0,0
 3706  024325     000     000     000
 3707  024330  000000                   BUFFER: 0
 3708          000001                   .END
```

```
ABASE = 172600          121#    234     275     393     394     395     396     397     398     399     400     401
ADDW1 = 000000          234     244
ADDW2 = 000000          234     245
ADPUOP= 000000          234     249
ADDW0 = 000000          234
ADDW1 = 000000          234
ADDW10= 000000          234
ADDW11= 000000          234
ADDW12= 000000          234
ADDW13= 000000          234
ADDW14= 000000          234
ADDW15= 000000          234
ADDW2 = 000000          234
ADDW3 = 000000          234
ADDW4 = 000000          234
ADDW5 = 000000          234
ADDW6 = 000000          234
ADDW7 = 000000          234
ADDW8 = 000000          234
ADDW9 = 000000          234
ADEVCT= 000000          234     240
ADEVM = 000003          124#    234     276     835
ADENV = 000000          234     245
ADENVM= 000000          234     246
ADATAL= 000000          234     247
AMADR1= 000000          234     262
AMADR2= 000000          234     266
AMADR3= 000000          234     269
AMADR4= 000000          234     272
AMAMS1= 000000          234     256
AMAMS2= 000000          234     264
AMAMS3= 000000          234     267
AMAMS4= 000000          234     270
AMMSGAD= 000000         234     242
AMMSGLG= 000000         234     243
AMMSGTY= 000000         234     238
AMTYP1= 000000          234     257
AMTYP2= 000000          234     265
AMTYP3= 000000          234     268
AMTYP4= 000000          234     271
APASS = 000000          234     239
APPRIOR= 000200         123#    234     2692
APTCSU= 000040          3235    3340#
APTENV= 000001          2944    3228    3296    3338#
APTSIZ= 000200          478     3337#
APTSPO= 000100          3230    3298    3339#
ASWREG= 000000          234     247
ATESTN= 000000          234     238
AUNIT = 000000          234     241
AUSWR = 000000          234     248
AVECT1= 100360          122#    234     273     403     404
AVECT2= 000000          234     274
BEGIN   001572          147     441#    2448    3192
BIT0  = 000001          106#    795     1302    1401    1468    1547    2778
BIT00 = 000001          96#     106
BIT01 = 000002          95#     105
```

```
BIT02 = 000004          94#   104
BIT03 = 000010          93#   103
BIT04 = 000020          92#   102
BIT05 = 000040          91#   101
BIT06 = 000100          90#   100
BIT07 = 000200          89#    99
BIT08 = 000400          88#    98  2906
BIT09 = 001000          87#    97
BIT1  = 000002         105#  2446
BIT10 = 002000          86#   841   894  1220  1289  1376  1415  1416  1428  1443  1592  1609  1629
                       1651  1673  1695  1728  1755  1782  1809  1835  1862  1889  1916  1942  1969
                       1996  2023  2049  2076  2103  2130  2155  2182  2208  2235  2251  2293  2311
                       2332
BIT11 = 004000          85#   853   894  1074  1364  1415  1416  1428  1443  2254  2297  2332  2351
                       2372  2393  2420  2443  2446  2485  2617  2618
BIT12 = 010000          84#   865   894  1057  1546  2239  2485  2618  2698  2786  2831  2834  2836
BIT13 = 020000          83#   880   887   894   976  1009  2700  2702  2835  2837  2939
BIT14 = 040000          82#   908  2496  2834  2836  2892
BIT15 = 100000          81#   503   508   513   542   547   552   571   576   581   600   605   610
                       629   634   639   908   947   960   961  1458  1468  1484  2589  2775  2778
                      2835  2837
BIT2  = 000004         104#
BIT3  = 000010         103#
BIT4  = 000020         102#  1537
BIT5  = 000040         101#
BIT6  = 000100         100#  1525  2586  2799
BIT7  = 000200          99#  1485  1538
BIT8  = 000400          98#  1326  1388  1415  1416  1428  1443  1592  1609  1636  1658  1680  1702
                       1728  1755  1782  1809  1835  1862  1889  1916  1942  1969  1996  2023  2049
                       2076  2103  2130  2155  2182  2208  2235  2251  2293  2311  2332  2351  2372
                       2393  2420  2443  2446  2613  2617
BIT9  = 001000          97#  1459  1466  1494  2593
BPTVEC= 000014         113#
BRIAN   017504        2814  2818#
BUFFER  024330        1172  1179*  1186  1202*  1224  1262  2533  2558  2570  2779*  3707#
BURN    001544         431#   439*   441*   531  1047  1573
BURNIN  001564         151    439#
BUSOUT  016416         152   2625#
BYTST   017572        2834#
CKSWR = 104407        1205  2486  2619  2703  2890  2891  2931  2954  3557#
CLRMAP  016234        1322  1586  1607  1628  1650  1672  1694  1715  1742  1769  1796  1822  1849
                      1876  1903  1929  1956  1983  2010  2036  2063  2090  2117  2143  2169  2195
                      2222  2249  2309  2329  2350  2371  2392  2413  2452  2586#  2611  2615
CNTSAV  015766        2495*  2508*  2517  2519*  2521#
COAX    001550         433#  1049*  1575*
COAXSW  001546         432#   517*   556*   585*   614*   643*  1049  1575  2789
CONT    001576         440    442#
COUNT   015676         920    922  2496#  2518
COUNTA  015672         918    977  1010  2495#
CR    = 000015          21#  3274  3284
CRLF  = 000200          22#  3245  3284
CROSS   015642         149  2483#
DCATST  006270        1168#
DCONT   016002        2272  2279  2525#
DCISP = 177570          28#   214   466
DELAY   016666        1050  1061  1078  1095  1117  1125  1149  1157  1208  1257  1286  1331  1457
```

```
                          1467    1483    1576    1594    1613    1637    1659    1681    1703    1730    1757    1784    1811
                          1837    1864    1891    1918    1944    1971    1998    2025    2051    2078    2105    2132    2157
                          2184    2210    2237    2252    2255    2294    2298    2314    2336    2353    2356    2359    2374
                          2377    2390    2395    2399    2401    2421    2598    2691#   2777
DETECT    003340          497     536     565     594     623     692#
DETECT    003222          533     650#
DETER0    002062          495#
DETER1    002312          532     534#
DETER2    002474          563#
DETER3    002656          592#
DETER4    003040          621#
DEVTMP    004360          791*    792*    796     835#    1298*   1299*   1303
DFC       024322          302     308     314     320     326     332     339     345     351     357     363     369     375
                          381     387     3705#
DH1       024004          300     306     3665#
DH15      024176          373     3689#
DH16      024212          379     385     3691#
DH3       024060          312     3673#
DH4       024105          318     3677#
DH5       024135          324     3682#
DH6       024151          330     337     343     349     355     361     367     3685#
DISPLA    001142          214#    466*    474*    2914*   2934*
DISPRE    000174          144#    474
DISPY     015770          2268    2275    2283    2286    2523#
DSWR    = 177570          27#     213     465
DT1       024226          301     307     3694#
DT15      024300          374     3702#
DT16      024306          380     386     3703#
DT3       024234          313     3695#
DT4       024246          319     3697#
DT5       024260          325     3699#
DT6       024266          331     338     344     350     356     362     368     3700#
DUALPL    007346          1317    1337#
EMTVEC= 000030            116#    453*    454*
EM1       023032          299     3577#
EM10      023430          342     3622#
EM11      023465          348     3627#
EM12      023524          354     3633#
EM13      023563          360     3639#
EM14      023611          366     372     3643#
EM16      023635          378     3647#
EM17      023714          384     3655#
EM2       023133          305     3588#
EM3       023214          311     3597#
EM4       023254          317     3603#
EM5       023300          323     3607#
EM6       023336          329     3612#
EM7       023374          336     3617#
ERRVEC= 000004            109#    463     464*    475*    693*    697*    706*    707*    770*    781*    782*    2626*   2897
                          2898*   2900*   2903*
FILBUF    016132          1111    2549    2557#
FILLWC    016076          1094    2548#
FRANKD    016322          150     2609#
GENER     006354          1180#   1184    1199    1207
GENRX     006426          1188    1192#
GNS     = ****** U        143     3549    3550    3551    3552    3553    3555    3557    3558    3559    3560
```

```
$TSWR = 104406          3555#
MATCH   015442           148    2440#
MATCHC  015510          2442    2445    2452#
HOME    017256          1093    1106    1119    1135    1139    1151    1167    1210    1219    1288    2777#
HT    = 000011            19#   3243    3284
INTVEC= 000020           114#    451#    452#
JOHNK = ******           969    1568
LE      001500           413#   2548
LASTCH  001532           426#   1122    1154    2574
LBMT    006566          1224#
LBMT1   006732          1241    1256#
LDM     006744          1261#
LDLIN   017554          1729    1756    1783    1810    1836    1863    1890    1917    1943    1970    1997    2024    2050
                        2077    2104    2131    2156    2183    2209    2236    2830#
LF    = 000012            20#   3278    3284
LIC     016166          1142    2570#
LOCADR  003576           499     538     567     596     625     736#   1314    2441    2484    2610
LOCSEQ  016774          2330    2414    2717#
LOCSTA  017062          1588    1631    1653    1675    1697    1716    1743    1770    1797    1823    1850    1877    1924
                        1930    1957    1984    2011    2037    2064    2091    2118    2144    2170    2196    2223    2333
                        2417    2742#
MAPIND  017526          2788#   2792#   2796#   2822#
MAPRET  007174          1304    1306#   2433
MAPRTA  007362          1300    1313    1333    1344#
MAPRTB  007160          1303#   1310
MOVADR  003730           755#   1329    2444    2614
MOVDN1  006676          1227    1244#   1247
MOVDWN  006672          1234    1243#
MOVRIG  006714          1228    1230    1231    1235    1237    1238    1250#
MSAVE0  001552           434#    482#    503#    508#    512#    513#    514     520     524
MSAVE1  001554           435#    483#    542#    547#    551#    552#    553     559
MSAVE2  001556           436#    484#    571#    576#    580#    581#    582     588
MSAVE3  001560           437#    485#    600#    605#    609#    610#    611     617
MSAVE4  001562           438#    486#    629#    634#    638#    639#    640     646
MSGINC  017254          1175    2775#
MSGTXT  017154          1173    2764#
NOCHAR  007112          1295#
NUMBR   007270          1305    1351#
NUMPIX  001542           430#   1722    1749    1776    1803    1829    1856    1893    1910    1936    1963    1990    2017
                        2043    2070    2097    2124    2149    2176    2202    2229    2720
OVRAL   017152          1170#   1206#   2763#
PIRG  = 177772            26#
PIRQVE= 000240           120#
PR0   = 000000            43#
PR1   = 000040            44#
PR2   = 000100            45#
PR3   = 000140            46#
PR4   = 000200            47#
PR5   = 000240            48#
PR6   = 000300            49#
PR7   = 000340            50#
PS    = 177776            23#     24
PSW   = 177776            24#
PWRMSG  021154          3190    3197#
PWRVEC= 000024           115#    467#    458#   3159#   3160#   3169#   3175#   3187#   3188#
RDCHR = 104410          3454    3558#
```

```
RDLIN = 104411          3495   3559#
RDOCT = 104412          3560#
RESTAT  002060           488    491#   683
RESVEC= 000010           110#
RETURN  007366           496*   535*   564*   593*   622*  1344   1346#
SAVE0   001510           417#   921*   925    932    934   3697
SAVE1   001512           418#   512    514    551    553    580    582    609    611    638    640    720*
SAVE2   001514           419#   923*   924    928    938   3697
SAVE3   001516           420#  1014*  1015*  1016
SET     017150          1171*  1177*  1186*  1189*  1196*  1197   2762#
STACK = 001100            14#   449   2440   2483   2609   2625
STCHAR  001530           425#  1107*  1110   1121*  1122   1137*  1141   1153*  1154
STKLMT= 177774            25#
SWR     001140           213#   447    465*   467    473*   480*  1055   1072   1091   1104   1134   1165   1217
                        1581   1603   1624   1646   1668   1690   1712   1739   1766   1793   1819   1846   1873
                        1900   1926   1953   1980   2007   2033   2060   2087   2114   2140   2166   2192   2219
                        2246   2263   2306   2326   2347   2368   2389   2410   2487   2620   2698   2786   2892
                        2906   2908   2939   2951   3167   3180*  3351   3398*
SWREG   000176           145#   473   3351   3364
SW0   = 000001            78#
SW00  = 000001            68#    78
SW01  = 000002            67#    77
SW02  = 000004            66#    76
SW03  = 000010            65#    75
SW04  = 000020            64#    74
SW05  = 000040            63#    73
SW06  = 000100            62#    72
SW07  = 000200            61#    71
SW08  = 000400            60#    70
SW09  = 001000            59#    69   1581   1603   1624   1646   1668   1690   1712   1739   1766   1793   1819
                        1846   1873   1900   1926   1953   1980   2007   2033   2060   2087   2114   2140   2166
                        2192   2219   2246   2263   2306   2326   2347   2368   2389   2410
SW1   = 000002            77#
SW10  = 002000            58#  1055   1072   1091   1104   1134   1165   1217
SW11  = 004000            57#
SW12  = 010000            56#
SW13  = 020000            55#
SW14  = 040000            54#
SW15  = 100000            53#
SW2   = 000004            76#
SW3   = 000010            75#
SW4   = 000020            74#
SW5   = 000040            73#
SW6   = 000100            72#
SW7   = 000200            71#
SW8   = 000400            70#
SW9   - 001000            69#
TABLE1  016446          2335   2633#
TABLE2  016466          1718   1745   1772   1799   2146   2172   2198   2225   2639#
TABLE3  016506          1590   2645#
TABLE4  016526          1633   1655   1677   1699   2651#
TABLE5  016546          2419   2657#
TABLE6  016606          1825   1852   1879   1906   2669#
TABLE7  016626          1932   1959   1986   2013   2675#
TABLE8  016646          2039   2066   2093   2120   2681#
TBITVE= 000014           111#
```

```
TEMP0   001534   427* 1109* 1115* 1120* 1140* 1147* 1152*
TEMP1   001502   414* 789* 790* 799* 811 1296* 1297* 1306* 1351 1352 1353 1354
TEMP2   001504   415* 695* 701* 702 716 720 723*
TEMP3   001506   416* 696* 722* 725* 726* 728 795* 796 801* 802 1302* 1303 1308*
                 1309
TESTID  017306   1584 1605 1626 1648 1670 1692 1714 1741 1768 1795 1821 1848 1875
                 1902 1928 1955 1992 2009 2035 2062 2089 2116 2142 2168 2194 2221
                 2248 2265 2308 2328 2349 2370 2391 2412 2786*
TITLE   022712   490 3563*
TKVEC = 000060   118*
TOTALC  001520   421* 1170
TPVEC = 000064   119*
TRAPVE= 000034   117* 455* 456*
TRTVEC= 000014   112*
TSTIND  017550   2798* 2801* 2805* 2811* 2826*
TST1    003762   519 558 587 616 645 764*
TST10   004666   898 903*
TST100  015220   2367 2369 2386*
TST101  015322   2388 2390 2407*
TST102  015434   2409 2411 2432*
TST11   004736   910 916*
TST12   005100   945*
TST13   005202   963 972*
TST14   005336   988 999 1005*
TST15   005526   1046*
TST16   005636   1054 1056 1069*
TST17   005730   1071 1073 1088*
TST2    004074   787*
TST20   005766   1090 1092 1101*
TST21   006112   1103 1105 1131*
TST22   006244   1133 1135 1162*
TST23   006534   1164 1166 1214*
TST24   007110   1216 1218 1294*
TST25   007450   1363*
TST26   007514   1369 1375*
TST27   007560   1381 1387*
TST3    004362   805 839*
TST30   007624   1393 1399*
TST31   007712   1414*
TST32   007762   1421 1426*
TST33   010036   1435 1441*
TST34   010074   1447 1453*
TST35   010256   1495 1501*
TST36   010354   1520*
TST37   010470   1544*
TST4    004434   846 852*
TST40   010626   1572*
TST41   010724   1580 1582 1600*
TST42   011012   1602 1604 1621*
TST43   011106   1623 1625 1643*
TST44   011202   1645 1647 1665*
TST45   011276   1667 1669 1687*
TST46   011372   1689 1691 1709*
TST47   011504   1711 1713 1736*
TST5    004500   858 864*
TST50   011616   1738 1740 1763*
```

```
TST51   011730          1765    1767    1790*
TST52   012042          1792    1794    1816*
TST53   012154          1818    1820    1843*
TST54   012266          1845    1847    1870*
TST55   012400          1872    1874    1897*
TST56   012512          1899    1901    1923*
TST57   012624          1925    1927    1950*
TST6    004544           970     876*
TST60   012736          1952    1954    1977*
TST61   013050          1979    1981    2004*
TST62   013162          2006    2008    2030*
TST63   013274          2032    2034    2057*
TST64   013406          2059    2061    2084*
TST65   013520          2086    2088    2111*
TST66   013632          2113    2115    2137*
TST67   013744          2139    2141    2163*
TST7    004630           892*
TST7C   014060          2165    2167    2189*
TST71   014174          2191    2193    2216*
TST72   014320          2218    2220    2243*
TST73   014406          2245    2247    2260*
TST74   014612          2262    2264    2303*
TST75   014712          2305    2307    2323*
TST76   015014          2325    2327    2344*
TST77   015116          2346    2348    2365*
TYPDS = 104405           672    3553*
TYPE  = 104401           489     670     673     712     715     719    2942    2970    2937    2989    2992    2994    2998
                        3005    3070    3145    3189    3248    3362    3363    3366    3379    3390    3409    3458    3461
                        3465    3549*
TYPOC = 104402           714    2978    3002    3365    3550*
TYPON = 104404          3552*
TYPOS = 104403           717    3551*
VCCSR   001446           393*    736     842*    843     854*    855     866*    867     881*    882     887*    894*    896
                         905*    906*    907     917*    946*    948     955*    956     962     976*    986*    998*   1009*
                        1021*   1025*   1035*   1057*   1074*   1083*   1220*   1289*   2239*   2485*   2498    2537    2539*
                        2618*   2700*   2702*   2781    2831*   3695    3697    3699

VCHRHI  001452           395*    742*   1077*
VCHRLO  001450           394*    741*   1059*   1060*   1076*   2487*   2620*   2830*
VCINT   001470           403*    739     749*    750*    751     978*   1011*   1028*   2691*   2706*
VCINT1  001472           404*    751*    752*    979*    991*   1012*   2692*   2706    2707*
VCPOS   001454           396*    743*   1201*   2812*
VCPOSH  001456           397*    744*   1200*   2913*
VGBUF   001464           400*    747*    757*   1353*   1356*   1524*   1549*   1553*   1635*   1657*   1679*   1701*   1721
                        1748    1775    1802    1828    1855    1882    1909    1935    1962    1989    2016    2042    2069
                        2096    2123    2148    2174    2200    2227    2458*   2464*   2467*   2528*   2726*   2729*
VGCOLR  001466           401*    748*    758*   1324*   1354*   1357*   1612*   2250*   2290*   2291*   2310*   2352*   2355*
                        2358*   2373*   2376*   2379*   2394*   2397*   2400*   2475*   2476*   2612*   2616*   2749*
VGCSR   001460           398*    745*    755*   1325*   1326*   1351*   1365*   1366    1377*   1378    1389*   1390    1402*
                        1403    1416*   1417*   1418    1428*   1429    1432    1443*   1445    1459*   1460    1469    1486
                        1506    1526    1531    1551    1554    1556    1592*   1609*   1629*   1636*   1651*   1658*   1673*
                        1680*   1695*   1702*   1723    1728*   1750    1755*   1777    1782*   1804    1809*   1830    1835*
                        1857    1862*   1884    1889*   1911    1916*   1937    1942*   1964    1969*   1991    1996*   2018
                        2023*   2044    2049*   2071    2076*   2098    2103*   2125    2130*   2150    2155*   2177    2182*
                        2203    2208*   2230    2235*   2251*   2254*   2293*   2297*   2311*   2313*   2332*   2351*   2372*
                        2393*   2416*   2420*   2426*   2443*   2446*   2456    2462    2465    2523*   2525*   2526    2529*
                        2593*   2594    2613*   2617*   2724    2727    3700    3702
```

```
VGPC    001462          399*   746*   756*  1352*  1355*  1505*  1523*  1548*  1550*  1634*  1656*  1678*  1700*
                       1720*  1747*  1774*  1801*  1827*  1854*  1881*  1908*  1934*  1961*  1988*  2015*  2041*
                       2068*  2095*  2122*  2147*  2173*  2195*  2226*  2454*  2524*  2719*
VHO     001524          423*  1109   1120   1140   1152   1171   1183   2550
VH1     001526          424*  1225   1243   1261   1273
VTHDR1  022750          712   3568*
VTHDR2  022774          715   3572*
VTHDR3  023002          719   3573*
WIDTH   001522          422*  1143   1189   2557
XADDS   017146         1195*  1201   2761*
XHAIR   001536          428*  1064
XPRNT   016032         1113   1145   1204   1256   1285   2533*  2551   2780
XPRNTA  016036         2534*  2815
YADDS   017144         1185*  1200   2760*
YHAIR   001540          429*  1081
$APTHD  001000          173    179*
$ASTAT= ****** U       3318   3333
$ATYC   021530         3289   3291*
$ATY1   021504         3287*
$ATY3   021512         3233   3288*
$ATY4   021522         2947   3290*
$AUTOB  001134          210*  3359   3480
$BASE   001246          275*   495    528   2627
$BDADR  001122          205*
$BDDAT  001126          207*   528*   698*   699*   700    769*   771    772    774*   811*   812    813*   814
                        815*   816    817*   818    819*   820    821*   822    823*   824    825*   826    827*
                        843*   844*   845    855*   856*   857    867*   868*   869    882*   883*   884    896*
                        897*   907*   908*   909    924*   925*   928*   929*   930*   931*   932    934*   935*
                        936*   937*   938    948*   956*   962*  1366*  1367*  1368   1378*  1379*  1380   1390*
                       1391*  1392   1403*  1404*  1405   1418*  1419*  1420   1432*  1433*  1434   1445*  1446*
                       1456*  1460*  1469*  1470   1477   1486*  1487   1494   1506*  1507*  1508   1531*  1532*
                       1533   1556*  1557*  1558   3694   3695   3700
$CDW1   001252          277*   502*   507*   541*   546*   570*   575*   599*   604*   628*   633*   728*   791
                       1298   1315
$CDW2   001254          278*   500    502    539    541    568    570    597    599    626    628
$CHARC  021500         3250*  3260*  3267   3276*  3281*
$CKSWR  021752         3351*  3557
$CMTAG  001100          193*   444    445    453    459    460
$CM1  = 000002          225*   226*   227*
$CM2  = 000004          225*   226*   227*
$CM3  = 000002          223*   225
$CNTLG  022477         3362   3475*
$CNTLU  022472         3379   3474*
$CPUOP  001220          249*
$CRLF   001167          228*  2942   2961   2970   2989   2994   2998   3249   3284   3390   3474
$DBLK   020772         3111   3145   3153*
$DEVCT  001202          240*
$DEVM   001250          276*   507    546    575    604    633
$DOAGN  003314          666    675    681*
$DTBL   020762         3114   3149*
$ENDAD  003304          159    677*  2956
$ENDCT  003252          459    668*
$ENDMG  003323          670    685*
$ENULL  003320          673    684*
$ENV    001212          245*  2944   3228   3296   3320
$ENVM   001213          246*   478    505    544    573    602    631   3230   3235   3298
```

```
$EOP    003222  659#
$EOPCT  003244  459*  665#  669
$ERFLG  001103  196#  2882  2913*  2917  2932*  2961
$ERMAX  001115  202#  2917
$ERROR  020036  453  2930#
$ERRPC  001116  203#  2936*  2937*  2938  2961  2976  3694  3695  3697  3699  3700  3702  3703
$ERRTB  001256  295#  2984
$ERRTY  020174  2941  2969#
$ERTTL  001112  200#  2935*  2961
$ETABL  001212  244#
$ETEND  001256  185  279#
$FATAL  001174  237#  3324*
$FFLG   021750  3287*  3290*  3318  3327*  3335#
$FILLC  001156  221#  3253  3284
$FILLS  001155  220#  3284
$GDADR  001120  204#  495*  534*  563*  592*  621*  692  737  767  769  789  1296  3703
$GDDAT  001124  206#  692*  694  698  713  767*  768*  772  841*  842  845  853*  854
                857  865*  866  869  880*  881  884  893*  904*  905  909  947*  954*
                960*  1058*  1060  1063*  1064  1075*  1077  1080*  1081  1364*  1365  1368  1376*
                1377  1380  1388*  1389  1392  1401*  1402  1405  1408*  1409  1415*  1420  1427*
                1434  1442*  1455*  1466*  1470  1476*  1485*  1487  1493*  1503*  1508  1512*  1522*
                1533  1537*  1538  1546*  1558  1565*  3695  3700
$GET42  003274  674#
$GTSWR  022022  3363*  3555
$HD   = 000000  11
$HIBTS  001000  180#
$HINUM  017702  1169*  1181  1193  2856  2864  2869*  2874#
$HIOCT  022626  3512*  3517#
$ICNT   001104  197#
$ILLUP  021146  3159  3175  3194#
$INTAG  001135  211#  3391  3480
$ITEMB  001114  201#  2938*  2946  2961  2973
$LF     001170  229#  2961  3284  3465  3474
$LFLG   021747  3328*  3334#
$LONUM  017704  1168*  2855  2862  2868*  2875#
$LPADR  001106  198#  460*  527*  2912*  2915  2917
$LPERR  001110  199#  526*  766*  788*  1400*  1502*  1521*  1545*
$MADR1  001224  262#
$MADR2  001230  266#
$MADR3  001234  269#
$MADR4  001240  272#
$MAIL   001172  181  185  235#  477  2911  2944  3228
$MAMS1  001222  256#
$MAMS2  001226  264#
$MAMS3  001232  267#
$MAMS4  001236  270#
$MBADR  001002  181#
$MFLG   021746  3288*  3294  3329*  3333#
$MNEW   022515  3366  3478#
$MSGAD  001206  242#  3304*  3307
$MSGLG  001210  243#  3309*
$MSGTY  001172  236#  3302  3310*  3322  3326*
$MSWR   022504  3363  3476#
$MTYP1  001223  257#
$MTYP2  001227  265#
$MTYP3  001233  268#
```

```
$MTYP4  001237          271#
$NULL   001154          219#  3255  3284
$NWTST= 000001          761#   784#   836#   849#   861#   873#   889#   900#   913#   942#   969#  1002#  1043#
                       1066#  1085#  1098#  1128#  1159#  1211#  1291#  1360#  1372#  1384#  1396#  1411#  1423#
                       1438#  1450#  1498#  1517#  1541#  1569#  1597#  1618#  1640#  1662#  1684#  1706#  1733#
                       1760#  1787#  1813#  1840#  1867#  1894#  1920#  1947#  1974#  2001#  2027#  2054#  2081#
                       2108#  2134#  2160#  2186#  2213#  2240#  2257#  2300#  2320#  2341#  2362#  2383#  2404#
                       2429#
$OCNT   020552         3042*  3071*  3084#
$OMCDE  020554         3037*  3041*  3046   3049*  3060*  3086#
$OVER   020022         2893   2904   2909   2914#
$PASS   001200          239#   477*   510    549    578    607    636    662*   663*   671    684    710   1053
                       1070   1089   1102   1132   1163   1215   1312   1579   1601   1622   1644   1666   1688
                       1710   1737   1764   1791   1817   1844   1871   1898   1924   1951   1978   2005   2031
                       2058   2085   2112   2138   2164   2190   2217   2244   2261   2304   2324   2345   2366
                       2387   2408
$PASTM  001006          183#
$PWRAD  021142         3192#
$PWRDN  021002          457   3159#  3187
$PWRMG  021136         3190#
$PWRUP  021054         3169   3175#
$QUES   001166          227#  2961   3284   3409   3458   3474
$RAND   017604         1180   1192   2851#
$RDCHR  022234         3422#  3558
$RDDEC= ****** U       3561
$RDLIN  022354         3450#  3559
$RDOCT  022526         3490#  3560
$RDSZ = 000010         3443#
$REGAD  001160          223#
$REGO   001162          225#
$REG1   001164          226#
$RTNAD  003316          683#
$R2A  = ****** U       3561
$SAVRE= ****** U       3561
$SAVR6  021152         3168*  3176   3177*  3178*  3196#
$SCOPE  017706          451   2889#
$SETUP= 000137          441#   450    451    453    455    457    459    460    661   2890   2931   2954   2956
                       3346   3480
$STUP = 177777          441#
$SVLAD  020000         2901   2907   2910#
$SVPC = 000230          157#   162
$SWR  = 160400            1#    11    130    131    132    133    134    135    227    460    461    656    662
                        676    682    684    765    788    840    853    865    877    893    904    917    946
                        973   1006   1047   1070   1089   1102   1132   1163   1215   1295   1364   1376   1388
                       1400   1415   1427   1442   1454   1502   1521   1545   1573   1601   1622   1644   1666
                       1688   1710   1737   1764   1791   1817   1844   1871   1898   1924   1951   1978   2005
                       2031   2058   2085   2112   2138   2164   2190   2217   2244   2261   2304   2324   2345
                       2366   2387   2408   2433   2883   2884   2885   2892   2904   2906   2907   2910   2913
                       2917   2924   2925   2926   2927   2935   2939   2951   2955   2961   3193
$SWREG  001214          247#   480
$SWRMK= 000000          135    136   2885   2886   2908
$TEMP   001474          411#  1504*  1505   1513*  1514   1547*  1553   1564*  1562*  1563*  1564*  1610*  1612
                       1615*  2312*  2313   2316*  2317
$TEMPO  001476          412#
$TESTN  001176          238#  2911*
$TKB    001146          216#  3344   3355   3372   3426   3432
```

```
$TKS      001144          2154    3344    3353    3369    3393*   3424    3430
$TN     = 000103            1*      11     761     765*    784     788*    805     836     840*    846     849     853*    858
                           861     865*    870     873     877*    885     893*    898     900     904*    910     913     917*
                           942     946*    963     969     973*    988     999    1002    1006*   1043    1047*   1054    1056
                          1066    1070*   1071    1073    1085    1089*   1090    1092    1098    1102*   1103    1105    1128
                          1132*   1133    1135    1159    1163*   1164    1166    1211    1215*   1216    1218    1291    1295*
                          1358    1360    1364*   1369    1372    1376*   1381    1384    1388*   1393    1396    1400*   1411
                          1415*   1421    1423    1427*   1435    1438    1442*   1447    1450    1454*   1495    1498    1502*
                          1517    1521*   1541    1545*   1569    1573*   1580    1582    1597    1601*   1602    1604    1618
                          1622*   1623    1625    1640    1644*   1645    1647    1662    1666*   1667    1669    1684    1692*
                          1689    1691    1706    1710*   1711    1713    1733    1737*   1738    1740    1760    1764*   1765
                          1767    1787    1791*   1792    1794    1813    1817*   1818    1820    1840    1844*   1845    1847
                          1867    1871*   1872    1874    1894    1898*   1899    1901    1920    1924*   1925    1927    1947
                          1951*   1952    1954    1974    1978*   1979    1981    2001    2005*   2006    2008    2027    2031*
                          2032    2034    2054    2058*   2059    2061    2081    2085*   2086    2088    2108    2112*   2113
                          2115    2134    2138*   2139    2141    2160    2164*   2165    2167    2186    2190*   2191    2193
                          2213    2217*   2218    2220    2240    2244*   2245    2247    2257    2261*   2262    2264    2300
                          2304*   2305    2307    2320    2324*   2325    2327    2341    2345*   2346    2348    2362    2366*
                          2367    2369    2383    2387*   2388    2390    2404    2408*   2409    2411    2429    2433*
$TPB      001152           218*   3273*   3284
$TPFLG    001157           222*   3222    3284
$TPS      001150           217*   3271    3284
$TRAP     022630           455    3526*
$TRAP2    022652          3537*   3548
$TRP    = 000013          3541*   3550*   3551*   3552*   3553*   3554*   3555    3556*   3557    3558*   3559*   3560*   3561*
$TRPAD    022664          3531    3548*
$TSTM     001004           182*
$TSTNM    001102           195*    518*    557*    586*    615*    644*    661*   1358*   2797    2882    2908    2910*   2911
                          2914    2917    2934    2961
$TTYIN    022462          3451    3452    3469    3473*
$TYPBN= ****** U          3554
$TYPDS    020556          3099*   3553
$TYPE     021222          3222*   3315    3541    3549
$TYPEC    021434          3252    3259    3266    3271*   3272    3395
$TYPEX    021502          3277    3279    3282*
$TYPOC    020354          3040*   3550
$TYPON    020370          3039    3042*   3552
$TYPOS    020330          3035*   3551
$UNIT     001204           241*    765*    794*    800*    840*   1301*   1307*   2795
$UNITM    001010           184*
$USWR     001216           248*
$VECT1    001242           273*    749    1014
$VECT2    001244           274*
$XTSTR    017722          2895*
$$GET4= 000000             676*
$0FILL    020553          3036*   3040*   3050    3085*
$40CAT= ****** U          2892    2941
.       = 024332           139*    143*    157     158*    160*    162*    163*    169     170*    172*    174*    192*    230
                           448     460     684     688    2917    2961    3008*   3153*   3171    3195    3204*   3294    3336*
                          3344    3473*   3474    3480
.$ASTA= ****** U          3288    3291
.$X     = 001000           169*    174
```

```
COMMEN      1#    121#
DELAY     439#   1095   1117   1125   1149   1157   1208   1257   1286   1594   1637   1659   1681   1703   1730
         1757   1784   1811   1837   1864   1891   1918   1944   1971   1998   2025   2051   2078   2105   2132
         2157   2184   2210   2237   2252   2255   2314   2336   2353   2356   2359   2374   2377   2380   2395
         2398   2401   2421
DELAY1    439#   1613
ENDCOM      1#    121#
ERROR      15#    516    522    529    555    561    584    590    613    619    642    648    778    831    847
          859    871    886    899    911    927    940    950    958    966    987   1001   1026   1036   1370
         1382   1394   1407   1422   1436   1448   1462   1472   1479   1489   1496   1510   1530   1535   1560
         2504   2513   2600
ESCAPE      1#    121#
FIXADX    495#    496    535    564    593    622
GETPRI      1#    121#
GETSWR      1#    121#
HOME      439#   1093   1106   1119   1136   1139   1151   1167   1210   1219   1288
INH09     439#   1578   1601   1622   1644   1666   1688   1710   1737   1764   1791   1817   1844   1871   1898
         1924   1951   1978   2005   2031   2058   2085   2112   2138   2164   2190   2217   2244   2261   2304
         2324   2345   2366   2397   2408
INHIO     439#   1052   1070   1089   1102   1132   1163   1215
LABLIN    439#
LDXLIN    439#   1729   1756   1783   1810   1836   1863   1890   1917   1943   1970   1997   2024   2050   2077
         2104   2131   2156   2183   2209   2236
MULT        1#    121#
NEWTST      1#    121#    761    784    836    849    861    873    889    900    913    942    969   1002   1043
         1066   1085   1098   1128   1159   1211   1291   1360   1372   1384   1396   1411   1423   1438   1450
         1498   1517   1541   1569   1597   1618   1640   1662   1684   1706   1733   1760   1787   1813   1840
         1867   1894   1920   1947   1974   2001   2027   2054   2081   2108   2134   2160   2186   2213   2240
         2257   2300   2320   2341   2362   2383   2404   2429
POP         1#    121#   2870   3140   3180   3181   3330   3331   3513
PUSH        1#    121#   2851   3099   3161   3167   3291   3293   3314   3492
REPORT      1#    121#
SCOPE      16#    764    787    839    852    864    876    892    903    916    945    972   1005   1046   1069
         1088   1101   1131   1162   1214   1294   1363   1375   1387   1399   1414   1426   1441   1453   1501
         1520   1544   1572   1600   1621   1643   1665   1687   1709   1736   1763   1790   1816   1843   1870
         1897   1923   1950   1977   2004   2030   2057   2084   2111   2137   2163   2189   2216   2243   2260
         2303   2323   2344   2365   2386   2407   2432
SETPRI      1#    121#
SETTRA   3541#   3550   3551   3552   3553   3555   3557   3558   3559   3560
SETUP       1#    121#    442
SKIP        1#    121#    773    804    846    858    870    885    898    910    926    933    949    957    963
          988    999   1022   1037   1054   1056   1071   1073   1090   1092   1103   1105   1133   1135   1164
         1166   1216   1218   1241   1369   1381   1393   1406   1421   1435   1447   1461   1471   1478   1489
         1495   1509   1534   1539   1559   1580   1582   1602   1604   1616   1623   1625   1645   1647   1667
         1669   1689   1691   1711   1713   1738   1740   1765   1767   1792   1794   1818   1820   1845   1847
         1872   1874   1899   1901   1925   1927   1952   1954   1979   1981   2006   2008   2032   2034   2059
         2061   2086   2088   2113   2115   2139   2141   2165   2167   2191   2193   2218   2220   2245   2247
         2262   2264   2305   2307   2325   2327   2346   2348   2367   2369   2388   2390   2409   2411
SLASH       1#    121#
SPACE     121#
STARS       1#    121#    155    166    168    175    188    230    233    653    761    763    784    786    836
          838    849    851    861    863    873    875    889    891    900    902    913    915    942    944
          969    971   1002   1004   1043   1045   1066   1068   1085   1087   1098   1100   1128   1130   1159
         1161   1211   1213   1291   1293   1360   1362   1372   1374   1384   1386   1396   1399   1411   1413
         1423   1425   1438   1440   1450   1452   1498   1500   1517   1519   1541   1543   1569   1571   1597
         1599   1618   1620   1640   1642   1662   1664   1684   1686   1706   1708   1733   1735   1760   1762
```

```
          1787   1789   1813   1815   1840   1842   1867   1869   1894   1896   1920   1922   1947   1949   1974
          1976   2001   2003   2027   2029   2054   2056   2081   2083   2108   2110   2134   2136   2160   2162
          2186   2188   2213   2215   2240   2242   2257   2255   2300   2302   2320   2322   2341   2343   2362
          2364   2393   2385   2404   2406   2429   2431   2842   2879   2920   2964   3012   3089   3157   3173
          3207   3285   3343   3346   3414   3443   3482   3520
SWRSU        1#   121#   461#
TESTID     439#  1583   1605   1626   1648   1670   1692   1714   1741   1768   1795   1821   1848   1875   1902
          1928   1955   1982   2009   2035   2062   2089   2116   2142   2168   2194   2221   2248   2308   2328
          2349   2370   2391   2412
TRMTRP    3541#
TYPBIN       1#   121#
TYPDEC       1#   121#   671
TYPNAM       1#   121#
TYPNUM       1#   121#
TYPOCS       1#   121#
TYPOCT       1#   121#  2976   3000   3364
TYPTXT       1#   121#
$$CMRE     186#   225    226
$$CMTM     186#
$$ESCA       1#   121#
$$NEWT       1#   121#   761    784    836    849    861    873    889    900    913    942    969   1002   1043
          1066   1085   1098   1129   1159   1211   1291   1360   1372   1384   1396   1411   1423   1438   1450
          1498   1517   1541   1569   1597   1618   1640   1662   1684   1706   1733   1760   1787   1813   1840
          1867   1894   1920   1947   1974   2001   2027   2054   2081   2108   2134   2160   2186   2213   2240
          2257   2300   2320   2341   2362   2383   2404   2429
$$SET     3541#  3550   3551   3552   3553   3555   3557   3558   3559   3560
$$SETM     477#
$$SKIP       1#   121#   805    846    858    870    898    910    963    988    999   1054   1056   1071   1073
          1090   1092   1103   1105   1133   1135   1164   1166   1216   1218   1369   1381   1393   1421   1435
          1447   1495   1580   1582   1602   1604   1623   1625   1645   1647   1667   1669   1689   1691   1711
          1713   1738   1740   1765   1767   1792   1794   1818   1820   1845   1847   1872   1874   1999   1901
          1925   1927   1952   1954   1979   1981   2006   2008   2032   2034   2059   2061   2096   2098   2113
          2115   2139   2141   2165   2167   2191   2193   2218   2220   2245   2247   2262   2264   2305   2307
          2325   2327   2346   2348   2367   2369   2388   2390   2409   2411
.EQUAT       1#    11
.HEADE       1#
.KT11        1#
.SETUP       1#   441
.SWRHI       1#   126
.SWRLO     136#
.$ACTI       1#   153
.$APTB       1#   231#
.$APTH       1#   164
.$APTY       1#  3284
.$ASTA       1#
.$CATC       1#   137
.$CMTA       1#   186
.$DB2D       1#
.$DB2O       1#
.$DIV        1#
.$EOP        1#   650
.$ERRO       1#  2918
.$ERRT       1#  2962
.$MULT       1#
.$PARM       1#
.$POWE       1#  3155
```

```
.$RAND      1#   2940
.$RDDE      1#
.$RDOC      1#   3480
.$READ      1#   3341
.$R2AZ      1#
.$SAVE      1#
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#   2877
.$SIZE      1#
.$SPAC      1#
.$SUPR      1#
.$SWDO      1#
.$TRAP      1#   3518
.$TYPB      1#
.$TYPD      1#   3087
.$TYPE      1#   3205
.$TYPO      1#   3010
.$4DCA      1#
..117O      1#

. ABS.  024332     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 ,DZVSEA.SEQ/CRF/SOL/NL:TOC=DSKZ:SYSMAC.SML,DZVSEA.P11
 RUN-TIME: 68 65 5 SECONDS
 RUN-TIME RATIO: 286/138=2.0
 CORE USED:  34K  (67 PAGES)
```

Spooler runtime 16 Seconds, 70 KCS, 405 disk reads, 3 disk writes, 95 pages