NC11

GAMMA 11 EXERCISER
MD-11-DZNCB-C

EP-DZNCB-C-DL-A

FICHE 1 OF 1

DEC 1977
digital
MADE IN USA

.REM

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZNCB-C-D
PRODUCT NAME:	GAMMA 11 EXERCISER
DATE CREATED:	MAY 1976
MAINTAINER:	DIAGNOSTIC ENGINEERING
AUTHOR:	R. MOORE

0.0 TABLE OF CONTENTS

1.0 ABSTRACT

THIS IS A SELF CONTAINED PROGRAM DESIGNED TO CONVERT DATA FROM THE GAMMA CAMERA VIA THE NC11 INTERFACE AND DISPLAY IT ON A VT01 OR VSV01 DISPLAY. IT CAN BE USED FOR CAMERA/INTERFACE AND JOYSTICK SETUP OR FOR PRACTICE IN IMAGE MANIPULATION. IT IS MEANT AS A EXERCISER RATHER THAN A DIAGNOSTIC. TOTAL PROGRAM CONTROL IS ACCOMPLISHED THRU THE CONSOLE KEYBOARD.

2.0 REQUIREMENTS

2.1 EQUIPMENT

1. PDP-11 COMPUTER
2. I/O TERMINAL (I.E., ASR33 TTY)
3. NC11 INTERFACE
4. GAMMA CAMERA (OR SUBSTITUTE)
5. VT01 OR VSV01 DISPLAY

2.2 EQUIPMENT DOCUMENTATION - OPTIONAL

1. 4306 JOYSTICK
2. AR11 IF SELECTED FOR JOYSTICK MODE
3. NC11-A CAMERA FRONT-END INSTRUCTION MANUAL
4. DRAWING SET NC11-A-0

2.3 PRELIMINARY PROGRAMS

1. MAINDEC-11-DZNCA SHOULD HAVE PREVIOUSLY BEEN RUN

2.4 STORAGE

THIS PROGRAM USES ALL OF LOWER 12K OF MEMORY.

3.0 LOADING PROCEDURE

NORMAL PROCEDURE FOR LOADING A BINARY PROGRAM INTO MEMORY SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE

LOADING ADDRESS 200 AND STARTING WILL INITIALIZE THE SYSTEM AND AWAIT A KEYBOARD COMMAND.

5.0 OPERATOR OPTIONS

5.1 OPERATOR KEYBOARD OPTIONS

TYPE "N"- COLLECT NEW DATA FROM CAMERA - CLEARS CORE MATRIX FIRST
TYPE "C"- COLLECT DATA FROM CAMERA - STARTS NC11, PREVICUS CORE MATRIX DATA NOT CLEARED
TYPE "S"- STOP NC11 DATA COLLECTION - ALSO TERMINATES FREE RUN MODE
TYPE "Z"- ZOOM - SET NC11 TO GAIN 1
TYPE "R"- REGULAR - SET NC11 TO GAIN 2 (DEFAULT COND)
TYPE "V"- SELECT VT01 DISPLAY
TYPE "W"- SELECT VSV01 DISPLAY (DEFAULT COND USING 1ST BIT MAP)
TYPE "M"- SELECT VSV01 DISPLAY AND USE 2ND BIT MAP
TYPE "E"- ERASE THE SELECTED DISPLAY
TYPE "A"- DISPLAY ISOTOPE A (DEFAULT COND)
TYPE "B"- DISPLAY ISOTOPE B (B GAMMA)
TYPE "D"- DISPLAY DATA - DISPLAY SELECTED ISOTOPE
TYPE "L"- GET LOWER THRESHOLD FROM KEYBOARD (DEFAULT=0) - ALL MATRIX VALUES LESS THAN TYPED VALUE ARE NOT DISPLAYED
TYPE "U"- GET UPPER THRESHOLD FROM KEYBOARD (DEFAULT=177777) - ALL MATRIX VALUES GREATER THAN TYPED VALUE ARE NOT DISPLAYED
TYPE "F"- FREE RUN MODE - COLLECT AND DISPLAY NEW DATA CONTINUOUSLY - TYPE "S" TO TERMINATE THIS MODE
TYPE "G"- INITIALIZE EVERYTHING - START FRESH
TYPE "J"- JOYSTICK CALIBRATION USING AR11 - USE "CNTRL C" TO TERMINATE MODE
TYPE "K"- JOYSTICK CALIBRATION USING NC11 - USE "CNTRL C" TO TERMINATE MODE - USE "CNTRL G" TO TURN OFF/ON BELL (NO CONVERT INDICATION)
TYPE "T"- DISPLAY INTENSITY TEST PATTERN ON SELECTED DISPLAY - THIS FEATURE IS FOR DISPLAY VERIFCATION ONLY

TYPE "CNTRL & C" - ABORT WHATEVER - BACK TO KEYBOARD MONITOR
TYPE "CNTRL & G" - TURN OFF/ON PRINTER BELL UPON A NO CONVERT - NC11 JOYSTICK CAL ONLY

5.2 OPERATOR SWITCH REGISTER OPTION(S)

NONE

6.0 RECOMMENDED OPERATOR ACTION

6.1 CAMERA/ADC SETUP

1. PLACE RADIOACTIVE FLOOD SOURCE IN FRONT OF CAMERA DETECTOR, OR USE A WEAK RADIOACTIVE SAMPLE ABOUT 3 FEET FROM A DETECTOR WITH THE COLLIMATOR REMOVED.

2. COLLECT DATA (C) AND DISPLAY (D)

3. ADJUST THE X AND Y GAIN CONTROLS ON THE NADC1 UNTIL A ROUND CIRCLE* IS SEEN AND CENTERED WITHIN THE BOX ON SCREEN.

4. INCREASE THE CIRCLE* SIZE UNTIL THE EDGES JUST BEGIN TO TOUCH THE BOX.

5. COLLECT DATA FOR A 3 MINUTE PERIOD AND OBSERVE THE IMAGE. IF THERE IS AN ARTIFACT ON THE CENTER OF THE SCREEN THAT LOOKS LIKE A RIGHT ANGLE BRACKET, THE COINCIDENCE DELAY CIRCUIT SHOULD BE ADJUSTED.

6. IF THE MATRIX COUNT EQUALS 65,536, THE Z COUNT EQUALS 0 AND THE DISPLAY IS BLANK, THEN TURN OFF THE 'TEST CLOCK' SWITCH ON THE BACK OF THE INTERFACE AND THE SITUATION WILL BE REMEDIED.

7. IF THE MATRIX AND Z COUNTS EQUAL 0 THEN NO DATA WAS COLLECTED. THEREFORE, NO DISPLAY.

* THIS MAY BE A HEXAGON FOR SOME MAKES OF OF CAMERA.

6.2 JOYSTICK SETUP

6.2.1 H306/AR11

1. SELECT AR11 JOYSTICK CONFIGRATION BY TYPING "J".

2. ADJUST THE X & Y POTS ON THE H306 JOYSTICK SO THAT THE DOT (VTO1) OR CROSS HAIRS (VSV01) PROVIDE UNIFORM ACCESS WITHIN THE BOX DRAWN ON THE SCREEN. IDEALLY, A PHYSCIAL CENTER OF THE JOYSTICK WILL PROVIDE A POINT APPROXIMATELY CENTERED IN THE BOX.

3. CHECK THAT WHEN THE JOYSTICK INTERRUPT BAR IS DEPRESSED THE DISPLAYED POINT DOES NOT FOLLOW THE JOYSTICK.

4. A "CNTRL C" WILL RETURN USER BACK TO THE KEYBOARD MONITOR.

6.2.2 H306/NC11

1. SELECT NC11 JOYSTICK CONFIGURATION BY TYPING "K".
2. REPEAT STEP 2 IN 6.2.1. THE X & Y GAIN CONTROLS ON THE NAD01 ADC'S MAY REQUIRE ADJUSTMENT TO REALIZE THIS STEP*.
3. CHECK STEP 3 IN 6.2.1.
4. A "CNTRL C" WILL RETURN USER BACK TO THE KEYBOARD MONITOR.

* NOTE: IF THE INPUT TO THE ADC'S IS OUT OF RANGE THEN THE NC11 NO CONVERT FLAG WILL SET. THIS FLAG SETTING IS INDICATED BY RINGING THE BELL ON THE PRINTER (A 'CNTRL G' WILL TURN OFF/ON THE BELL).

7.0 MISCELLANEOUS

7.1 DEVICE BUS ADDRESS MODIFICATIONS

NC11	MODIFY LOCATION "NCADR" IF BASE BUS ADDRESS IS NOT 164000
VSV01	MODIFY LOCATION "VTVADR" IF BASE BUS ADDRESS (CHARACTER GENERATOR) IS NOT 172600 MODIFY LOCATION "VTMADR" IF BASE BUS ADDRESS (BIT MAP) IS NOT 172620
VTO1	MODIFY LOCATION "VTADR" IF BASE BUS ADDRESS IS NOT 176756
AR11	MODIFY LOCATION "ARADR" IF BASE BUS ADDRESS IS NOT 170400
NOTE:	A RESTART IS REQUIRED AFTER ANY OF THE ABOVE ADDRESS MODIFICATIONS.

7.2 FREE RUN MODE (TYPE "F")

MODIFICATION OF LOCATION "TIME" WILL ALTER THE DATA COLLECTION TIME IN THE FREE RUN MODE. THE DEFAULT VALUE OF 10(8) PROVIDES ABOUT 1 SECOND.

7.3 ERROR REPORTING

1. INCORRECT KEYBOARD COMMANDS ARE RESPONDED WITH '?'
2. ANY SELECTED DEVICE (SEE SECTION 7.1) WHICH DOES NOT RESPOND ON THE UNIBUS WILL BE INDICATED.

7.4 EXECUTION TIME

THE EXECUTION TIME IS COMPLETELY DEPENDENT UPON THE USER FOR WHATEVER OPTION SELECTED.

8.0 PROGRAM DESCRIPTION

8.1 DATA COLLECTION & DISPLAY

THE USER MAY COLLECT DATA FROM THE GAMMA CAMERA SYSTEM VIA THE NC11 INTERFACE AND DISPLAY THIS DATA ON A VT01 STORAGE SCOPE OR THE VSV01 (BITMAP) DISPLAY. ALL FUNCTIONS ARE ENTERED THRU THE KEYBOARD AS DEFINED IN SECTION 5.1. WHEN THE DATA COLLECTION MODE IS SELECTED THE NC11 IS RECEIVING X & Y DATA FROM THE GAMMA CAMERA (LOOKING A RADIOACTIVE SOURCE). THIS INFORMATION IS TRANSFORMED VIA THE ADDRESS MAKER LOGIC WHICH FORMS A UNIQUE ADDRESS WITHIN A DEFINED MATRIX RELATIVE TO THE SCAN POSITION OF THE CAMERA. THE NC11 PREFORMS AN NPR INCREMENT TO MEMORY TO THIS UNIQUE ADDRESS. THE PROGRAM SELECTS THE ADDRESS MATRIX 64X64X16 (RESOLUTION 2) OFFSET TO ADDRESS 20000(8) FOR THE 'A' ISOTOPE, AND FOR THE 'B' ISOTOPE, (CAMERAS EQUIPPED WITH THE DUAL ISOTOPE ATTACHMENT) INFORMATION VIA B GAMMA LOGIC IS STORED STARTING AT ADDRESS 40000 (8). THE INTENSITY OF EACH CELL (MEMORY LOCATION) OF THE IMAGE IN MEMORY IS ADJUSTED WITH THE UPPER AND LOWER THRESHOLDS AND SCALED TO AN INTENSITY LEVEL (32 FOR VT01 AND 16 FOR VSV01). THE CELL WITH THE HIGHEST COUNT REPRESENTS THE HIGHEST INTENSITY LEVEL, THEREFORE APPEARS BRIGHTEST ON THE DISPLAY. AT THE COMPLETION OF THE IMAGE DISPLAY, THE FOLLOWING PARAMETERS ARE DISPLAYED:

LOWER THRESHOLD-	ALL VALUES GREATER THAN THIS VALUE ARE DISPLAYED
UPPER THRESHOLD-	ALL VALUES ABOVE THIS VALUE ARE OMITTED
Z COUNT-	32 BIT COUNTER OF A/D START PULSES
MATRIX COUNT-	TOTAL COUNT OF THE CONTENTS OF EACH CELL IN THE 64X64 MATRIX
ZOOM-	DISPLAYED IF GAIN WAS SELECTED
B-GAMMA-	DISPLAYED IF 'B' ISOTOPE SELECTED

8.2 JOYSTICK MODES

8.2.1 H306/AR11

SELECTING THIS MODE WILL PROVIDE THE USER WITH A CHECK OF THE ABILITY OF THE H306 WHEN CONNECTED TO THE AR11 TO DISPLAY A X AND Y INDICATION ON THE SELECTED DISPLAY. THIS INDICATION SHOULD AGREE WITH THE PROCEDURE DEFINED IN SECTION 6.2.1. THE AR11 A/D CHANNELS HAVE THE FOLLOWING RELATIONSHIP: CHAN 0 = Y DATA, CHAN 1 = X DATA AND CHAN 2 IS THE INTERRUPT BAR INDICATOR (0=DEPRESSED COND.).

8.2.2 H306/NC11

SELECTING THE NC11 CONFIGURATION TAKES ADVANTAGE OF THE JOYSTICK MODE PROVIDED BY THE NC11. THIS MODE SELECTS THE EXTERNAL INPUTS AND THE NC11 PERFORMS LIKE A A/D WITH THE X & Y CONVERTED VALUES AVAILABLE IN X-Y HOLDING REGISTER. THE X AND Y DATA IS APPLIED TO THE SELECTED DISPLAY AND SHOULD CONFORM TO THE REQUIREMENTS DEFINED IN SECTION 6.2.2.

NOTE: WHEN USING THE VT01 DISPLAY THE WRITE - THRU MODE IS USED WHICH DISPLAYS A SMALL CIRCLE OR DOT. THE VSV01 USES THE X & Y CROSS HAIRS AS THE DISPLAY INDICATOR.

9.0 LISTING

```
                        ;
                .TITLE  MAINDEC-11-DZNCB-C  GAMMA 11 EXERCISER
                ;*COPYRIGHT (C) 1976
                ;*DIGITAL EQUIPMENT CORP.
                ;*MAYNARD, MASS. 01754
                ;*
                ;*PROGRAM BY R.MOORE
                ;*
                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                ;*PACKAGE (MAINDEC-11-DZQAC-CO),MAR 21, 1976.
                ;*
000001          $TN=1
160000          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
                .SBTTL  BASIC DEFINITIONS

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

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

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

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

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
000020                          SW04=   20
000010                          SW03=   10
000004                          SW02=   4
000002                          SW01=   2
000001                          SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
```

```
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

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

                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                          TBITVEC=14              ;;"T" BIT
000014                          TRTVEC= 14              ;;TRACE TRAP
000014                          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                          PWRVEC= 24              ;;POWER FAIL
000030                          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                          TRAPVEC=34              ;;"TRAP" TRAP
000060                          TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                          TPVEC=  64              ;;TTY PRINTER VECTOR
000240                          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR


                                ;SOME COMMON PROGRAM VALUES AND EQUATES

000020                          CLZ=20                  ;CLEARS Z REG AT REG 14
000100                          CLALL=100               ;CLEARS NC11 AT REG 14
020000                          MATRIX=20000            ;STARTING ADRS OF MATRIX DATA

                                .SBTTL  TRAP CATCHER
```

```
                                                .=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
                                                .=174
000174  000000                      DISPREG:  .WORD  0                ;;SOFTWARE DISPLAY REGISTER
000176  000000                      SWREG:    .WORD  0                ;;SOFTWARE SWITCH REGISTER
                                    .SBTTL  STARTING ADDRESS(ES)
000200  000137  001356                        JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
```

```
                                .SBTTL  COMMON TAGS

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

                        001100                  .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102  000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002             $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000             $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
001161  015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001162  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


001164                          $ERRTB:
                                ;THIS PROGRAM DOES NOT USE THE ABOVE ERROR TABLE









                                        ;NC11 BUS ADRS ASSIGNMENTS - MODS ARE TO BE MADE HERE

001164  164000                  NCADR:  164000

                                        ;VSV01 BUS ADRS ASSIGNMENTS (CHAR GEN) - MODS ARE TO BE MADE HERE

001166  172600                  VTVADR: 172600

                                        ;VSV01 BUS ADRS ASSIGNMENTS (BIT MAP) - MODS ARE TO BE MADE HERE

001170  172620                  VTMADR: 172620

                                        ;VT01 BUS ADRS ASSIGNMENTS - MODS ARE TO BE MADE HERE

001172  176756                  VTADR:  176756

                                        ;AR11 BUS ADRS ASSIGNMENTS - MODS ARE TO BE MADE HERE

001174  170400                  ARADR:  170400
```

```
                                        ;NC11 REGISTER ADDRESS POINTERS

001176  164000          NCCSR:  164000          ;COMMAND/STATUS
001200  164002          NCOFF:  164002          ;OFFSET
001202  164004          NCAREG: 164004          ;ADDRESS
001204  164006          NCZHI:  164006          ;HI Z COUNT
001206  164010          NCZLO:  164010          ;LO Z COUNT
001210  164012          NCIREG: 164012          ;I
001212  164014          NCSFUN: 164014          ;SPECIAL FUNCTIONS

                                        ;VSV01 REGISTER ADDRESS POINTERS (CHARACTER GENERATOR)

001214  172600          VTVCRG: 172600          ;CHAR/STATUS
001216  172602          VTVCHP: 172602          ;CROSS HAIR POS
001220  172604          VTVPOS: 172604          ;CHAR POS

                                        ;VSV01 REGISTER ADDRESS POINTERS (BIT MAP)

001222  172620          VTVCSR: 172620          ;COMMAND/STATUS
001224  172622          VTVMAP: 172622          ;MAP ADRS
001226  172624          VTVPX:  172624          ;PIXEL WD
001230  172626          VTVPX1: 172626          ;PIXEL BYTE
001232  172630          VTVINT: 172630          ;INTENSITY LOOK UP

                                        ;VT01 REGISTER ADDRESS POINTERS

001234  176756          VTCSR:  176756          ;COMMAND/STATUS
001236  176760          VTXDAC: 176760          ;X DAC
001240  176762          VTYDAC: 176762          ;Y DAC

                                        ;AR11 REGISTER ADDRESS POINTERS

001242  172400          ARCSR:  172400          ;COMMAND STATUS & CHANNEL
001244  172402          ARBUF:  172402          ;A D BUFFER
```

CO2


```
                                  ;COMMON PROGRAM TAGS AND STORAGE LOCATIONS

001246  000000      TEMP0:   0          ;COMMON UTILITY LCC
001250  000000      TEMP1:   0          ;COMMON UTILITY LOC
001252  000000      TEMP2:   0          ;COMMON UTILITY LOC
001254  000000      DTYPE:   0          ;0=VSV01 DISPLAY,-1=VT01 DISPLAY
001256  000020      INTENS:  20         ;TOTAL # OF INTENSITY LEVELS, 16 OR 32
001260  000000      INTLUT:  0          ;INITIAL INTENSITY SHADE
001262  006400      VTVSAV:  6400       ;VSV01 SET UP - FULL SCREEN, MONO(BLK/WHT), ENA DISPLAY
001264  000000      MRXADR:  0          ;CURRENT CORE ADRS OF CELL BEING DISPLAYED
001266  000000      MAPADR:  0          ;CURRENT ADRS OF BIT MAP LD
001270  000003      PIXCNT:  3          ;CURRENT PIXEL BYTE COUNT
001272  000000      PIXASM:  0          ;4 PIXELS ASSEMBLED HERE BEFORE BIT MAP LD
001274  000000      HORIZ:   0          ;COUNTS 64 DISPLAYED POSITIONS EACH ROW (VT01)
001276  000000      VERT:    0          ;COUNTS 64 ROWS FOR VT01 DISPLAY
001300  000000      XREF:    0          ;X POS FOR VT01 DISPLAY
001302  000000      YREF:    0          ;Y POS FOR VT01 DISPLAY
001304  000000      KBDBUF:  0          ;CONTAINS ASCII CHAR THAT NEEDS CONVERSION FOR VT01 DISP
001306  000000      KBUFF:   0          ;CONTAINS KEYBOARD CHAR TYPED
001310  000000      TTYOUT:  0          ;OUTPUT CHAR TO PRINTER
001312  000000      THLO:    0          ;LOW THRESHOLD VALUE APPLIED TO MATRIX CELLS
001314  177777      THHI:    177777     ;HIGH THRESHOLD VALUE APPLIED TO MATRIX CELLS
001316  000000      COMSAV:  0          ;SAVED NC11 CSR CONTENTS WHEN DISPLAYING
001320  000000      GAIN:    0          ;GAIN 2=0, GAIN 1=40
001322  010000      TOTSIZ:  4096.      ;TOTAL # OF CELLS IN MATRIX (ISTOPE A OR B)
001324  000000      CHARCT:  0          ;COUNTS TOTAL # OF CELLS IN MATRIX (ISOTOPE A OR B)
001326  000000      BASX:    0          ;CONTAINS CURRENT X DAC POSITION (VT01)
001330  000000      BASY:    0          ;CONTAINS CURRENT Y DAC POSITION (VT01)
001332  000006      INCXY:   6          ;CONTAINS CHAR SIZE OFFSET (VT01)
001334  000000      OPERL:   0          ;CONTAINS LARGEST CELL OF MATRIX WITHIN THRESHOLDS
001336  000000      ROWCNT:  0          ;COUNTS 64 CELLE EACH ROW OF CORE MATRIX
001340  020000      TABLEX:  MATRIX     ;CONTAINS STARTING ADRS OF MATRIX OF SELECTED ISOTOPE
001342  000000      FREERN:  0          ;NON-ZERO SAYS COLLECT NEW DATA & DISPLAY IN FREE RUN MO
001344  000010      TIME:    10         ;HIGH ORDER COUNT DELAY FOR COLLECTING DATA-FREE RUN MOD
001346  000000      JTYPE:   0          ;0=NC11 JOYSTICK SOURCE,-1=AR11 JOYSTICK SOURCE
001350  020000      ARCHAN:  20000      ;CONTAINS CURRENT AR11 CHAN & UNIPOLAR BIT
001352  000000      MSELCT:  0          ;0 SAYS 1ST BIT MAP, -1 SAYS 2ND BIT MAP(VS.01)
001354  000000      BELLEN:  0          ;0 SAYS RING BELL ON NO CONVERT, -1 SAYS CON'T
```

```
                                                .SBTTL  MAIN PROGRAM

001356                                  START:
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001356  012706  001100                          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001362  005026                                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001364  022706  001140                          CMP     #SWR,R6 ;;DONE?
001370  001374                                  BNE     .-6             ;;LOOP BACK IF NO
001372  012706  001100                          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
001376  012737  011150  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001404  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
001412  012737  011214  000024                  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001420  012737  000340  000026                  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001426  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001432  012737  001466  000004                  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001440  012737  177570  001140                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001446  012737  177570  001142                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001454  022777  177777  177456                  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001462  001012                                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
001464  000403                                  BR      65$             ;;BRANCH IF NO TIMEOUT
001466  012716  001474                  64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001472  000002                                  RTI
001474  012737  000176  001140          65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001502  012737  000174  001142                  MOV     #DISPREG,DISPLAY
001510  012637  000004                  66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001514  104400  013263                          TYPE    ,MSG1           ;GO IDENTIFY PROGRAM
001520  004737  005112                          JSR     PC,SELCTA       ;DEFAULT TO VSV01 DISPLAY IF THERE
001524  103010                                  BCC     3$              ;BR IF DISPLAY VSV01 IS THERE
001526  004737  005262                          JSR     PC,SELCTB       ;IF NOT GO LOOK FOR VTC1 DISPLAY
001532  103005                                  BCC     3$              ;BR IF AT LEAST THERE IS A VTC1 DISPLAY
001534  104400  013412                          TYPE    ,MSG3           ;GO TYPE 'BUS TIMEOUT ER - VSV01 DISPLAY'
001540  104400  013453                          TYPE    ,MSG4           ;GO TYPE 'BUS TIMEOUT ER - VTC1 DISPLAY'
001544  000000                                  HALT                    ;NO DISPLAY SEEN AT ASSIGNED BUS ADRS'S
001546  013700  001164          3$:     MOV     NCADR,R0        ;GET NC11 BASE ADRS
001552  012701  001176                  MOV     #NCCSR,R1       ;GET POINTER ADRS
001556  010021                  NCSET:  MOV     R0,(R1)+        ;SET UP NC11 REG ADRS PTRS
001560  062700  000002                  ADD     #2,R0           ;BUMP REG ADRS
001564  022701  001214                  CMP     #VTVCRG,R1      ;ALL SET UP?
001570  001372                          BNE     NCSET           ;BR IF NOT
001572  012737  001606  000004          MOV     #1$,ERRVEC      ;SET UP TIMEOUT ADRS
001600  005777  177372                  TST     @NCCSR          ;WILL TRAP TO LOC 4 IF NOT THERE
001604  000404                          BR      START1          ;BR IF THERE
001606  022626                  1$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
001610  104400  013513                  TYPE    ,MSG5           ;GO TYPE 'BUS TIMEOUT ER - NC11'
001614  000000                          HALT                    ;NC11 NOT SEEN AT ASSIGNED BUS ADRS
001616  012737  000006  000004  START1: MOV     #ERRVEC+2,@#ERRVEC      ;RESTORE ERR TRAP LOC TO PT TO 6
001624  000005                          RESET                   ;CLR WORLD
001626  004737  005506                  JSR     PC,NCSTP1       ;GO STOP NC11 & CLR CORE MATRIX
001632  012777  020000  177340          MOV #MATRIX,@NCOFF      ;SET OFFSET ADRS 20000
001640  005037  001320                  CLR GAIN                ;REGULAR GAIN
```

```
                        ;INITIALIZE THE COMMON TAGS

   748  001644  012737  020000  001340          MOV #MATRIX,TABLEX      ;ISOTOPE A
   749  001652  005037  001312                  CLR THLO                ;CLEAR LOWER THRESHOLD
   750  001656  012737  177777  001314          MOV #177777,THHI        ;CEILING THRESHOLD
   751  001664  012737  006122  000060          MOV     #KBINT,TKVEC    ;SET KEYBOARD INTR RETURN ADRS
   752  001672  012737  000340  000062          MOV     #340,TKVEC+2    ;SET UP NEW PRIORTY ON INTR
   753  001700  012777  000100  177235          MOV     #100,@STKS      ;SET INTR ENABLE
   754  001706  104400  013543                  TYPE    ,MSG5           ;GO ASK FOR KEYBOARD COMMAND'S.
```

```
                                        ;THIS IS A LIST OF KEYBOARD COMMANDS FOR THE NC11/DISPLAY/JOYSTICK.

                                                ;D      DISPLAY DATA
                                                ;E      ERASE THE SCOPE
                                                ;L      GET LOWER THRESHOLD FROM KEYBOARD & DISPLAY DATA
                                                ;U      GET UPPER THRESHOLD FROM KEYBOARD & DISPLAY DATA
                                                ;N      COLLECT NEW DATA FROM CAMERA (CLEAR CORE MATRIX)
                                                ;C      COLLECT DATA FROM CAMERA
                                                ;S      STOP COLLECTION
                                                ;G      INITIALIZE EVERYTHING
                                                ;Z      ZOOM - SET GAIN TO 1
                                                ;R      REGULAR - SET GAIN TO 2
                                                ;A      DISPLAY ISOTOPE A
                                                ;B      DISPLAY ISOTOPE B
                                                ;V      SELECT VTO1 DISPLAY
                                                ;W      SELECT VSV01 DISPLAY (DEFAULT USING 1ST BIT MAP)
                                                ;M      SELECT VSV01 DISPLAY AND USE 2ND BIT MAP
                                                ;F      FREE RUN MODE - COLLECT & DISPLAY NEW DATA CONTINUALLY
                                                ;J      JOYSTICK CALIBRATION USING AR11
                                                ;K      JOYSTICK CALIBRATION USING NC11
                                                ;T      DISPLAY INTENSITY TEST IMAGE ON SELECTED DISPLAY
                                                ;CNTRL C  ABORT WHATEVER - BACK TO KEYBOARD MONITOR
                                                ;CNTRL G  TURN ON/OFF BELL (NO CONVERT INDICATION) - IN NC11
                                                        ;JOYSTICK CALIBRATION ONLY

                                        ;THIS CODE WILL DISPATCH PROGRAM TO THE PROPER
                                        ;ROUTINE VIA THE DISPATCH TABLE 'RTABLE'
001712  005037  001342  LISEN:  CLR     FREERN          ;KNOCK DOWN FREE RUN MODE IF SET
001716  012706  001100          MOV     #STACK,SP       ;RESET STACK PTR
001722  005037  001306          CLR     KBUFF           ;INSURE NO KEYBOARD GARBAGE
001726  005037  177776          CLR     PSW             ;ALLOW KEYBOARD INTR
001732  013700  001306  _ISN:   MOV     KBUFF,R0        ;LOOK FOR CHAR
001736  001775                  BEQ     LISN            ;WAIT FOR ONE
001740  005037  001306          CLR     KBUFF           ;
001744  020027  000101          CMP     R0,#101         ;WAS IT A CHAR?
001750  103445                  BCS     B00B00          ;NOT A GOOD CHAR
001752  042700  177740          BIC     #177740,R0      ;ELIMINATE LOWER CASE
001756  020027  000033          CMP     R0,#33          ;IS IT AN ALPHA CHAR?
001762  103040                  BCC     B00B00          ;BR IF NOT
001764  006300                  ASL     R0              ;MAKE UP WORD OFFSET
001766  005760  001776          TST     RTABLE-2(R0)    ;IS IT A LEGAL COMMAND?
001772  001434                  BEQ     B00B00          ;BR IF NOT
001774  000170  001776          JMP     @RTABLE-2(R0)   ;GO DO IT

                                        ;KEYBOARD DISPATCH TABLE

002000  002104  RTABLE: ROUTA                   ;'A' DISPLAY ISOTOPE A
002002  002116          ROUTB                   ;'B' DISPLAY ISOTOPE B
002004  002130          ROUTC                   ;'C' COLLECT DATA
002006  002140          ROUTD                   ;'D' DISPLAY DATA
002010  002144          ROUTE                   ;'E' ERASE SCOPE
002012  002154          ROUTF                   ;'F' FREE RUN MODE(COLLECT NEW DATA & DISPLAY CONTINOUSL
002014  002250          ROUTG                   ;'G' INITIALIZE EVERYTHING
002016  000000          0                       ;'H' B00B00
002020  000000          0                       ;'I' B00B00
002022  002260          ROUTJ                   ;'J' GO TO AR11 JOYSTICK CALIBRATION
```

```
002024  002302                          ROUTK                   ;'K' GO TO NC11 JOYSTICK CALIBRATION
002026  002312                          ROUTL                   ;'L' GET LOWER THRESHOLD FROM KEYBOARD
002030  002342                          ROUTM                   ;'M' SELECT VSV01 DISPLAY AND USE 2ND BIT MAP
002032  002402                          ROUTN                   ;'N' GET NEW DATA FROM CAMERA
002034  000000                          0                       ;'O' BOOBOO
002036  000000                          0                       ;'P' BOOBOO
002040  000000                          0                       ;'Q' BOOBOO
002042  002420                          ROUTR                   ;'R' REGULAR GAIN
002044  002436                          ROUTS                   ;'S' STOP COLLECTION
002046  002454                          ROUTT                   ;'T' DISPLAY INTENSITY TEST IMAGE ON SELECTED DISPLAY
002050  002470                          ROUTU                   ;'U' GET UPPER THRESHOLD FROM KEYBOARD
002052  002520                          ROUTV                   ;'V' SELECT VTO1 DISPLAY
002054  002540                          ROUTW                   ;'W' SELECT VSV01 DISPLAY,DEFAULT COND USING 1ST BIT MAP
002056  000000                          0                       ;'X' BOOBOO
002060  000000                          0                       ;'Y' BOOBOO
002062  002576                          ROUTZ                   ;'Z' ZOOM - SET GAIN TO 1

                                ;REPORTS ILLEGAL KEYBOARD CHARACTERS

002064  012737  000077  001310  BOOBOO: MOV     #77,TTYOUT      ;SET UP '?'
002072  004737  006174                  JSR     PC,TYPO         ;TYPE IT
002076  004737  006154                  JSR     PC,TYPCR        ;DO A 'CR'
002102  000713                          BR      LISN            ;GO LOOK FOR GOOD CHAR

002104  012737  020000  001340  ROUTA:  MOV     #MATRIX,TABLEX  ;WILL DISPLAY ISOTOPE A
002112  000137  002616                  JMP     CHANGE          ;GO DO IT
002116  012737  040000  001340  ROUTB:  MOV     #MATRIX+20000,TABLEX    ;WILL DISPLAY ISOTOPE B
002124  000137  002616                  JMP     CHANGE          ;GO DO IT
002130  004737  005440          ROUTC:  JSR     PC,NCSTRT       ;GO START NC11
002134  000137  001732                  JMP     LISN            ;COLLECT DATA UNTIL KEYBRD COMMAND
002140  000137  002616          ROUTD:  JMP     CHANGE          ;GO DISPLAY DATA
002144  004737  005536          ROUTE:  JSR     PC,ERASE        ;GO ERASE SCOPE
002150  000137  001732                  JMP     LISN            ;GO WAIT ON NEXT COMMAND
002154  012737  177777  001348  ROUTF:  MOV     #-1,FREERN      ;SELECT FREE RUN MODE
002162  004737  005506                  JSR     PC,NCSTP1       ;GO STOP NC11 & CLR CORE MATRIX AREA
002166  004737  005440                  JSR     PC,NCSTRT       ;GO START NC11
002172  005000                          CLR     R0              ;SET UP TIMER
002174  013701  001344                  MOV     TIME,R1         ;SET UP GROSS TIMER VALUE
002200  005300                  1$:     DEC     R0              ;COUNT
002202  001376                          BNE     1$              ;WAIT FOR ZERO
002204  042737  000040  001306          BIC     #BIT5,KBUFF     ;LOOK FOR POSSIBLE STOP KEY - RID LOWER CASE
002212  022737  000123  001306          CMP     #123,KBUFF      ;STOP BEEN STRUCK?
002220  001007                          BNE     2$              ;BR IF NOT
002222  042777  000003  176746          BIC     #3,@NCCSR       ;STOP NC11
002230  005077  176742                  CLR     @NCCSR          ;ZERO NC CSR
002234  000137  001712                  JMP     LISEN           ;GO AWAIT NEXT COMMAND
002240  005301                  2$:     DEC     R1              ;DO LOOP AGAIN?
002242  001356                          BNE     1$              ;BR IF SO
002244  000137  002616                  JMP     CHANGE          ;NOW GO DISPLAY DATA JUST COLLECTED
002250  004737  005536          ROUTG:  JSR     PC,ERASE        ;GO ERASE SELECTED DISPLAY
002254  000137  001616                  JMP     START1          ;INITIALIZE AND START FRESH
002260  004737  005356          ROUTJ:  JSR     PC,SELCTC       ;GO SELECT AR11 FOR JOYSTICK CAL
002264  103402                          BCS     1$              ;BR IF BUS ER FROM AR11
002266  000137  004146                  JMP     TJOY            ;GO TO IT
002272  104400  013577          1$:     TYPE    ,MSG7           ;GO TYPE 'BUS TIMEOUT ER - AR11'
002276  000137  001732                  JMP     LISN            ;GO BACK TO KEYBOARD LISEN
```

```
 957  002302  005037  001346          ROUTK:  CLR     JTYPE           ;JTYPE=0 SAYS NC11 FOR JOYSTICK CAL
 958  002306  000137  004146                  JMP     TJOY            ;GO TO IT
 959  002312  012737  000340  177776  ROUTL:  MOV     #PR7,PSW        ;DON'T WANT ANY INTR'S NOW
 960  002320  104400  013331                  TYPE    .MSG2           ;ASK FOR OCTAL DATA
 961  002324  104410                          RDOCT                   ;GO GET IT
 962  002326  012637  001312                  MOV     (SP)+,THLO      ;SAVE IT
 963  002332  005037  177776                  CLR     PSW             ;ENABLE INTR'S
 964  002336  000137  002616                  JMP     CHANGE          ;GO DISPLAY
 965  002342  012737  177777  001352  ROUTM:  MOV     #-1,MSELCT      ;SELECT 2ND BIT MAP
 966  002350  004737  005112                  JSR     PC,SELCTA       ;GO SELECT VSV01
 967  002354  103407                          BCS     2$              ;BR IF BUS TIMEOUT ER FROM VSV01
 968  002356  013700  001222                  MOV     VTVCSR,R0       ;GET 2ND BIT MAP ADRS
 969  002362  042760  000400  177760          BIC     #BIT8,-20(R0)   ;TURN OFF 1ST BIT MAP
 970  002370  000137  001732          1$:     JMP     LISN            ;GO AWAIT NEXT COMMAND
 971  002374  104400  013412          2$:     TYPE    ,MSG3           ;GO TYPE 'BUS TIMEOUT ER - VSV01 DISPLAY'
 972  002400  000773                          BR      1$              ;GO BACK TO KEYBOARD LISEN
 973  002402  004737  005506  ROUTN:  JSR     PC,NCSTP1       ;GO STOP NC11 & CLR CORE MATRIX
 974  002406  013777  001316  176562          MOV     COMSAV,@NCCSR   ;RESTART WITH PREVIOUS CSR CONTENTS
 975  002414  000137  001732                  JMP     LISN            ;COLLECT DATA & AWAIT NEXT KEYBRD COMMAND
 976  002420  005037  001320  ROUTR:  CLR GAIN                        ;WANT REGULAR GAIN
 977  002424  042777  000040  176544          BIC     #40,@NCCSR      ;INSURE REGULAR GAIN
 978  002432  000137  001732                  JMP     LISN            ;LOOK FOR NEXT COMMAND
 979  002436  042777  000003  176532  ROUTS:  BIC     #3,@NCCSR       ;STOP NC
 980  002444  005077  176526                  CLR     @NCCSR          ;ZERO NC CSR
 981  002450  000137  001732                  JMP     LISN            ;LOOK FOR NEXT COMMAND
 982  002454  004737  005464  ROUTT:  JSR     PC,NCSTP        ;GO STOP THE NC11 IF RUNNING
 983  002460  004737  005756                  JSR     PC,LDIMGE       ;GO SET UP TEST CORE IMAGE
 984  002464  000137  002616                  JMP     CHANGE          ;GO DISPLAY IT
 985  002470  012737  000340  177776  ROUTU:  MOV     #PR7,PSW        ;DON'T WANT ANY INTR'S
 986  002476  104400  013331                  TYPE    .MSG2           ;ASK FOR OCTAL DATA
 987  002502  104410                          RDOCT                   ;GO GET IT
 988  002504  012637  001314                  MOV     (SP)+,THHI      ;SAVE IT
 989  002510  005037  177776                  CLR     PSW             ;ENABLE INTR'S
 990  002514  000137  002616                  JMP     CHANGE          ;GO DISPLAY
 991  002520  004737  005262  ROUTV:  JSR     PC,SELCTB       ;GO SELECT VTO1
 992  002524  103402                          BCS     2$              ;BR IF TIMEOUT ER FROM VTO1
 993  002526  000137  001732          1$:     JMP     LISN            ;GO AWAIT NEXT COMMAND
 994  002532  104400  013453          2$:     TYPE    ,MSG4           ;GO TYPE 'BUS TIMEOUT ER - VTO1 DISPLAY'
 995  002536  000773                          BR      1$              ;GO BACK TO KEYBOARD LISEN
 996  002540  005037  001352  ROUTW:  CLR     MSELCT          ;SELECT 1ST BIT MAP
 997  002544  004737  005112                  JSR     PC,SELCTA       ;GO SELECT VSV01(DEFAULT COND)
 998  002550  103407                          BCS     2$              ;BR IF TIMOUT ER FROM VSV01
 999  002552  013700  001222                  MOV     VTVCSR,R0       ;GET 1ST BIT MAP ADRS
1000  002556  042760  000400  000020          BIC     #BIT8,2C(R0)    ;TURN OFF 2ND BIT MAP
1001  002564  000137  001732          1$:     JMP     LISN            ;GO AWAIT NEXT COMMAND
1002  002570  104400  013412          2$:     TYPE    ,MSG3           ;GO TYPE 'BUS TIMEOUT ER - VSV01 DISPLAY'
1003  002574  000773                          BR      1$              ;GO BACK TO KEYBOARD LISEN
1004  002576  012737  000040  001320  ROUTZ:  MOV     #40,GAIN        ;ENABLE GAIN 1
1005  002604  053777  001320  176364          BIS     GAIN,@NCCSR     ;SET IT AT NC CSR
1006  002612  000137  001732                  JMP     LISN            ;GO AWAIT NEXT COMMAND
```

```
                                        ;GO CLEAR SCREEN OF SELECTED DISPLAY

002616  004737  005536         CHANGE: JSR     PC,ERASE        ;GO ERASE SCOPE

                                        ;STOP NC11 AND GET SET TO DISPLAY

002622  004737  005464         DISDAT: JSR     PC,NCSTP        ;GO STOP NC11

                                        ;NOW DRAW A BOX AROUND POTENTIAL MATRIX DISPLAY
                                        ;AREA ON THE SELECTED DISPLAY
002626  005737  001254         BOX:    TST     DTYPE           ;WHAT DISPLAY?
002632  001067                         BNE     BOX1            ;BR IF VT01
002634  012700  001750                 MOV     #1750,R0        ;SET UP BIT MAP ADRS - TOP LINE
002640  012701  073567                 MOV     #73567,R1       ;SET UP PIXEL DATA IN R1 - INT 7
002644  004737  005634                 JSR     PC,DISPY        ;GO LOAD BIT MAP
002650  005200                 1$:     INC     R0              ;ADVANCE ADRS
002652  022700  001770                 CMP     #1770,R0        ;TOP LINE DONE?
002656  001403                         BEQ     2$              ;BR IF SO
002660  004737  005646                 JSR     PC,DCONT        ;LOAD NEXT PIXEL WD
002664  000771                         BR      1$              ;NEXT MAP LOAD
002666  012700  006010         2$:     MOV     #6010,R0        ;SET UP BIT MAP ADRS - BOT LINE
002672  004737  005634                 JSR     PC,DISPY        ;GO LOAD BIT MAP
002676  005200                 3$:     INC     R0              ;ADVANCE ADRS
002700  022700  006030                 CMP     #6030,R0        ;BOT LINE DONE?
002704  001403                         BEQ     4$              ;BR IF SO
002706  004737  005646                 JSR     PC,DCONT        ;LOAD NEXT PIXEL WD
002712  000771                         BR      3$              ;NEXT MAP LOAD
002714  012700  001730         4$:     MOV     #1730,R0        ;SET UP BIT MAP ADRS SIDE LINES
002720  062700  000017         5$:     ADD     #17,R0          ;OFFSET NEXT ROW
002724  012701  070000                 MOV     #70000,R1       ;SET UP PIXEL 3 DATA IN R1 - INT 7
002730  022700  006047                 CMP     #6047,R0        ;AT BOTTOM OF SCREEN?
002734  001411                         BEQ     6$              ;GO START VSV01 DISPLAY
002736  004737  005634                 JSR     PC,DISPY        ;GO LOAD BIT MAP
002742  012701  000007                 MOV     #7,R1           ;SET UP PIXEL 0 DATA IN R1 - INT 7
002746  062700  000021                 ADD     #21,R0          ;OFFSET TO RIGHT LINE
002752  004737  005634                 JSR     PC,DISPY        ;GO LOAD BIT MAP
002756  000760                         BR      5$              ;DO NEXT ROW
002760  013777  001262  176234 6$:     MOV     VTVSAV,@VTVCSR  ;START DISPLAY
002766  012737  002010  001266         MOV     #2010,MAPADR    ;OFFSET BIT MAP ADRS
002774  012737  000003  001270         MOV     #3,PIXCNT       ;SET UP PIXEL BYTE COUNT
003002  005037  001272                 CLR     PIXASM          ;CLR PIXEL ASSEMBLY WORD
003006  000137  003132                 JMP     LARGES          ;GO DISPLAY CELL DATA

003012  012700  003777         BOX1:   MOV     #3777,R0        ;SET UP X
003016  012701  003636                 MOV     #3636,R1        ;SET UP Y
003022  004737  005676         1$:     JSR     PC,DISPY1       ;DISPLAY POINT
003026  005300                         DEC     R0              ;MOVE X BY 1
003030  100374                         BPL     1$              ;AGAIN TILL DONE
003032  005200                         INC     R0              ;X=0,Y=3636
003034  004737  005676         2$:     JSR     PC,DISPY1       ;DISPLAY POINT
003040  005301                         DEC     R1              ;MOVE Y BY 1
003042  022701  000635                 CMP     #635,R1         ;AT BOTTOM LEFT?
003046  001372                         BNE     2$              ;BR IF NOT
003050  005201                         INC     R1              ;Y=636,X=0
003052  004737  005676         3$:     JSR     PC,DISPY1       ;DISPLAY POINT
003056  005200                         INC     R0              ;MOV X BY 1
```

```
 973  003060  022700  004000                    CMP     #4000,R0        ;AT BOTTOM RIGHT?
 974  003064  001372                            BNE     3$              ;BR IF NOT
 975  003066  005300                            DEC     R0              ;X=3777,Y=636
 976  003070  004737  005676            4$:     JSR     PC,DISPY1       ;DISPLAY POINT
 977  003074  005201                            INC     R1              ;MOV Y BY 1
 978  003076  022701  003636                    CMP     #3636,R1        ;AT TOP RIGHT?
 979  003102  001372                            BNE     4$              ;BR IF NOT
 980  003104  012737  000100  001274            MOV     #100.,HORIZ     ;# OF HORIZONTAL POINTS PER ROW
 981  003112  012737  000100  001276            MOV     #100.,VERT      ;# OF ROWS
 982  003120  005037  001300                    CLR     XREF            ;START AT LEFT MARGIN
 983  003124  012737  003606  001302            MOV     #1926.,YREF     ;AND TOP OF SCREEN
 984
 985                                    ;NOW LETS FIND THE LARGEST CELL
 986  003132  013737  001322  001324    LARGES: MOV TOTSIZ,CHARCT       ;NUMBER OF ELEMENTS TO CONSIDER
 987  003140  017700  176174                    MOV @TABLEX,R0          ;THIS IS FIRST GUESS
 988  003144  013701  001340                    MOV TABLEX,R1           ;R1 POINTS TO TABLE
 989  003150  020021                    LARGEL: CMP R0,(R1)+            ;COMPARE GUESS AGAINST NEW
 990  003152  103005                            BHIS GSG                ;GUESS STILL GOOD
 991  003154  024137  001314                    CMP -(R1),THHI          ;GUESS SMALLER BUT CHECK NEW
 992  003160  101001                            BHI OVTHHI              ;HI AGAINST UPPER THESHOLD
 993  003162  011100                            MOV (R1),R0             ;WITHIN BOUNDS. MAKE THIS NEW HI
 994  003164  005721                    OVTHHI: TST (R1)+               ;INCREASE REGISTER BY 2
 995  003166  005337  001324            GSG:    DEC CHARCT              ;COUNT THIS LAST COMPARISON
 996  003172  001366                            BNE LARGEL
 997
 998                                    ;THE LARGEST CELL WITHIN THE UPPER THRESHOLD IS NOW IN R0
 999                                    ;NOW ACCOUNT FOR LOWER THRESHOLD - BOW OUT IF TOO SMALL
1000  003174  163700  001312                    SUB THLO,R0             ;SUBTRACT LOWER THRESHOLD
1001  003200  101002                            BHI     1$              ;BR IF CELL VALUE GREATER THAN LO THRESHOLD
1002  003202  000137  003500                    JMP     DISDON          ;DON'T DISPLAY-ALL VALUES BELOW LO THRESHOLD
1003  003206  010037  001334            1$:     MOV     R0,CPERL        ;SAVE LARGEST CELL OF MATRIX
1004
1005                                    ;NOW PICK OUT EACH VALUE IN CORE MATRIX AND SCALE TO THE
1006                                    ;PROPER INTENSITY LEVEL.  THEN DISPLAY IT ON THE SELECTED DISPLAY
1007
1008  003212  013737  001340  001264    DMATRX: MOV     TABLEX,MRXADR   ;BEGIN AT TOP LEFT ROW
1009  003220  062737  017600  001264            ADD     #8064.,MRXADR   ;OFFSET TO BOTTOM OF CORE MATRIX
1010  003226  012737  000100  001336            MOV     #64.,ROWCNT     ;THERE ARE 64 CELLS PER ROW
1011  003234  017702  176024            DISLOP: MOV     @MRXADR,R2      ;GET A CELL VALUE FROM MATRIX
1012  003240  062737  000002  001264            ADD     #2,MRXADR       ;BUMP MATRIX ADRS
1013  003246  005337  001335                    DEC     ROWCNT          ;COUNT CELL THIS ROW
1014  003252  001006                            BNE     CKVALU          ;BR IF ROW NOT FINISHED
1015  003254  012737  000100  001336            MOV     #64.,ROWCNT     ;RESET NEXT ROW COUNT
1016  003262  162737  000400  001264            SUB     #256.,MRXADR    ;SET UP FOR NEXT ROW IN MATRIX
1017  003270  020237  001314            CKVALU: CMP     R2,THHI         ;CELL WITHIN HI THRESHOLD?
1018  003274  101003                            BHI     OUTLIM          ;BR IF NOT
1019  003276  163702  001312                    SUB     THLO,R2         ;SUB LOW THRESHOLD
1020  003302  101006                            BHI     SCLCEL          ;BR IF CELL ABOVE LOW THRESHOLD
1021  003304  005737  001254            OUTLIM: TST     DTYPE           ;WHAT DISPLAY
1022  003310  001052                            BNE     CHDN            ;BR IF VT01
1023  003312  005004                            CLR     R4              ;SET CELL TO LOWEST INTENSITY
1024  003314  000137  003504                    JMP     MAPLD           ;LOAD BIT MAP
1025  003320  013701  001334            SCLCEL: MOV     CPERL,R1        ;NOW ESTABLISH WHAT INTENSITY LEVEL
1026  003324  012703  000006                    MOV     #6,R3           ;SCALE TO 32 LEVELS
1027  003330  005737  001254                    TST     DTYPE           ;IS IT A VSV01 DISPLAY?
1028  003334  001001                            BNE     1$              ;BR IF NOT
```

```
1029  003336  005303                          DEC     R3              ;SCALE TO 16 LEVELS ON VSV01 DISPLAY
1030  003340  005004                  1$:     CLR     R4              ;LEVEL BUILDS IN R4
1031  003342  006304                  DVLOOP: ASL     R4              ;MUL BY 2
1032  003344  020201                          CMP     R2,R1           ;COMPARE THIS CELL TO LARGEST 'DIVIDED BY 2)
1033  003346  103404                          BLO     NODEF           ;BR IF SMALLER
1034  003350  005701                          TST     R1              ;CHECK CASE WHERE 0 DIVISOR
1035  003352  001401                          BEQ     1$              ;BR IF SO
1036  003354  005204                          INC     R4              ;ACCOUNT FOR GOOD SUBTRACTION
1037  003356  160102                  1$:     SUB     R1,R2           ;NOW DO THE SUBTRACTION
1038  003360  000241                  NODEF:  CLC                     ;
1039  003362  006001                          ROR     R1              ;MAKE DIVISOR SMALLER
1040  003364  005303                          DEC     R3              ;COUNT POSITION
1041  003366  001365                          BNE     DVLOOP          ;AGAIN IF NOT SCALED TO 16 OR 32 LEVELS YET
1042  003370  160102                          SUB     R1,R2           ;ACCOUNT FOR REMAINDER
1043  003372  101401                          BLOS    GOON            ;BR IF TOO SMALL
1044  003374  005204                          INC     R4              ;ROUND UP
1045  003376  005737  001254          GOON:   TST     DTYPE           ;WHAT DISPLAY?
1046  003402  001440                          BEQ     MAPLD           ;IF VSV01 GO LOAD MAP
1047  003404  006304                          ASL     R4              ;MAKE LEVEL A WORD POINTER
1048  003406  016405  011364                  MOV     LEVTAB-2(R4),R5 ;POINTER TO CORRECT LEVEL
1049  003412  006204                          ASR     R4              ;RESTORE ACTUAL LEVEL
1050  003414  001410                          BEQ     CHDN            ;IF 0 - DON'T DISPLAY
1051
1052                                  ;THIS CODE DISPLAYS EACH SCALED CELL ON THE VT01 (ONE OF 32 LEVELS)
1053
1054  003416  013700  001300          MLOOP:  MOV XREF,R0             ;GET READY TO DISPLAY
1055  003422  013701  001302                  MOV YREF,R1             ;THESE ARE THE CORNER COORD.
1056  003426  004737  006212                  JSR PC,GET              ;GET A POINT AND DISPLAY IT
1057  003432  005304                          DEC R4                  ;DO AS MANY POINTS AS THE LEVEL
1058  003434  001370                          BNE MLOOP
1059  003436  062737  000040  001300  CHDN:   ADD #32.,XREF           ;SHIFT TO NEXT POSSIBLE DATUM
1060  003444  005337  001274                  DEC     HORIZ           ;DONE WITH LINE?
1061  003450  001271                          BNE DISLOP              ;NO
1062  003452  012737  000100  001274          MOV #100,HORIZ          ;RESET LINE COUNTER
1063  003460  005037  001300                  CLR     XREF            ;DO CR
1064  003464  162737  000030  001302          SUB #24.,YREF           ;DO LF
1065  003472  005337  001276                  DEC VERT                ;DONE PAGE?
1066  003476  001256                          BNE DISLOP              ;NO. GO BACK FOR MORE
1067  003500  000137  003652          DISDON: JMP PATWRT              ;YES. GO DISPLAY PRAMETERS
1068
1069                                  ;THIS CODE DISPLAYS EACH SCALED CELL ON THE VSV01 (ONE OF 16 LEVELS)
1070
1071  003504  013700  001266          MAPLD:  MOV     MAPADR,R0       ;SET UP BIT MAP ADRS
1072  003510  005704                          TST     R4              ;LOOK FOR NO INTENSITY
1073  003512  001401                          BEQ     1$              ;BR IF NO INTENSITY
1074  003514  005304                          DEC     R4              ;OFFSET TO 0-17
1075  003516  000304                  1$:     SWAB    R4              ;PREPARE FOR PIXEL LOC
1076  003520  006304                          ASL     R4              ;MOVE TO TOP 4 BITS
1077  003522  006304                          ASL     R4              ;
1078  003524  006304                          ASL     R4              ;
1079  003526  006304                          ASL     R4              ;
1080  003530  000241                          CLC                     ;NOW ASSEMBLE THIS PIXEL INTO PIXEL WORD
1081  003532  006037  001272                  ROR     PIXASM          ;NOW MAKE ROOM IN PIXEL WORD
1082  003536  006037  001272                  ROR     PIXASM          ;
1083  003542  006037  001272                  ROR     PIXASM          ;
1084  003546  006037  001272                  ROR     PIXASM          ;
```

```
1095  003552  060437  001272                ADD     R4,PIXASM       ;ADD THIS PIXEL TO OTHERS
1096  003556  005737  001270                TST     PIXCNT          ;ALL 4 PIXELS DONE FOR THIS WORD?
1097  003562  001004                        BNE     2$              ;BR IF NOT
1098  003564  013701  001272                MOV     PIXASM,R1       ;LD PIXEL WORD INTO R1
1099  003570  004737  005634                JSR     PC,DISPY        ;LOAD BIT MAP
1100  003574  005337  001270        2$:     DEC     PIXCNT          ;COUNT PIXEL
1101  003600  100215                        BPL     DISLOP          ;BR IF 4 PIXELS NOT ASSEMBLED YET
1102  003602  005037  001272                CLR     PIXASM          ;CLR PIXEL ASSEMBLY WORD
1103  003606  012737  000003  001270        MOV     #3,PIXCNT       ;RESET PIXEL COUNT
1104  003614  005237  001266                INC     MAPADR          ;ADVANCE MAP ADRS
1105  003620  013700  001266                MOV     MAPADR,R0       ;LOAD INTO R0
1106  003624  022700  005770                CMP     #5770,R0        ;HAVE ALL 64 ROWS BEEN DONE?
1107  003630  001002                        BNE     3$              ;BR IF NOT
1108  003632  000137  003662                JMP     PATWRT          ;NOW GO DISPLAY PARAMETERS
1109  003636  042700  177740        3$:     BIC     #177740,R0      ;SAVE ROW POSITION BITS
1110  003642  022700  000030                CMP     #30,R0          ;NOW LOOK FOR END OF ROW
1111  003646  001003                        BNE     4$              ;BR IF NOT AT END
1112  003650  062737  000020  001266        ADD     #20,MAPADR      ;ADVANCE MAP ADRS TO NEXT ROW
1113  003656  000137  003234        4$:     JMP     DISLOP          ;GO GET NEXT MATRIX DATUM
1114
1115                                        ;CODE TO WRITE PARAMETER DATA AT BOTTOM OF SCREEN
1116
1117  003662  005737  001254        PATWRT: TST     DTYPE           ;WHAT DISPLAY?
1118  003666  100404                        BMI     1$              ;BR IF VT01
1119  003670  012777  012000  175322        MOV     #12000,@VTVPOS  ;VSV01 - SET CHAR POS,LINE 20, LEFT MARGIN
1120  003676  000414                        BR      2$              ;START 1ST MSG
1121  003700  012737  000006  001332  1$:     MOV #6,INCXY            ;SET CHARACTER SIZE
1122  003706  012777  000014  175320        MOV #14,@VTCSR          ;SET MODE
1123  003714  012737  000000  001326        MOV     #0,BASX         ;SET TO LEFT HAND MARGIN
1124  003722  012737  000600  001330        MOV #384.,BASY          ;AND Y LEVEL
1125  003730  004737  006246          2$:     JSR PC,VTWRIT           ;"CR.LOWER THRESHOLD"
1126  003734  012506                        MESS1
1127  003736  013737  001312  006724        MOV THLO,DUMMY1
1128  003744  004737  006700                JSR PC,AWRIT
1129  003750  004737  006246                JSR PC,VTWRIT           ;"CR. UPPER THRESHOLD"
1130  003754  012512                        MESS2
1131  003756  013737  001314  006724        MOV THHI,DUMMY1
1132  003764  004737  006700                JSR PC,AWRIT
1133  003770  004737  006246                JSR PC,VTWRIT           ;"Z COUNT ="
1134  003774  012516                        MESS3
1135  003776  017737  175204  006724        MOV @NCZLO,DUMMY1
1136  004004  017737  175174  006726        MOV @NCZHI,DUMMY2
1137  004012  004737  006704                JSR PC,OTTY
1138  004016  004737  006246                JSR PC,VTWRIT           ;"MATRIX COUNT="
1139  004022  012522                        MESS4
1140  004024  013737  001322  001246        MOV TOTSIZ,TEMPO
1141  004032  005002                        CLR R2
1142  004034  005003                        CLR R3
1143  004036  013701  001340                MOV TABLEX,R1
1144  004042  062103          MXSML:  ADD     (R1)+,R3        ;NOW ADD UP ALL VALUES IN MATRIX
1145  004044  005502                        ADC R2
1146  004046  005337  001246                DEC TEMPO
1147  004052  001373                        BNE MXSML
1148  004054  010337  006724                MOV R3,DUMMY1
1149  004060  010237  006726                MOV R2,DUMMY2
1150  004064  004737  006704                JSR PC,OTTY
```

```
1141  004070  005737  001320                    TST GAIN
1142  004074  001403                            BEQ 1$
1143  004076  004737  006246)                   JSR PC,VTWRIT
1144  004102  012526                            MESS5                   ;"ZOOM"
1145  004104  023727  001340  040000  1$:       CMP TABLEX,#MATRIX+20000 ;ISOTOPE B?
1146  004112  001003                            BNE 2$                  ;NO
1147  004114  004737  006246                    JSR PC,VTWRIT           ;YES
1148  004120  012532                            MESS6                   ;"B-GAMMA"
1149  004122  005737  001342          2$:       TST     FREERN          ;IN FREE RUN MODE?
1150  004126  001402                            BEQ     3$              ;BR IF NOT
1151  004130  000137  002154                    JMP     ROUTF           ;YES, GO GET NEW DATA
1152  004134  013777  001316  175034  3$:       MOV COMSAV,@NCCSR       ;RESUME THE NC11
1153  004142  000137  001732                    JMP     LISN            ;DONE WITH MESSAGES
1154
1155
1156                                    ;THIS CODE DRAWS A BUG(VT01) OR CROSS HAIRS(VSV01) WITH THE X & Y DATA
1157                                    ;FROM THE NC11(TYPE 'K' OR DEFAULT) OR THE AR11(TYPE 'J') - THE BUG,
1158                                    ;ETC. WILL FOLLOW THE JOYSTICK WHEN THE INTERRUPT BAR IS NOT DEPRESSED -
1159                                    ;ALL PARTS WITHIN THE DISPLAYED BOX ON THE SELECTED DISPLAY SHOULD
1160                                    ;BE ACCESSIBLE IN A UNIFORM MANNER
1161                                    ;A CNTRL 'C' WILL GET USER BACK TO KEYBOARD MONITOR
1162  004146  004737  005536            TJOY:   JSR     PC,ERASE        ;START FRESH
1163  004152  005737  001254                    TST     DTYPE           ;WHAT DISPLAY?
1164  004156  001053                            BNE     JBOX1           ;BR IF VT01
1165
1166                                    ;DRAW A BOX UP ON VSV01 SCREEN FOR NC11 OR AR11 JOYSTICK CALIBRATION
1167  004160  005000                    JBOX:   CLR     R0              ;SET UP BIT MAP ADRS - TOP LINE
1168  004162  012701  073567                    MOV     #73567,R1       ;SET UP PIXEL DATA IN R1 - INT 7
1169  004166  004737  005634                    JSR     PC,DISPY        ;GO LOAD BIT MAP
1170  004172  005200                    1$:     INC     R0              ;ADVANCE BIT MAP ADRS
1171  004174  022700  000040                    CMP     #40,R0          ;TOP LINE DONE?
1172  004200  001403                            BEQ     2$              ;BR IF SO
1173  004202  004737  005646                    JSR     PC,DCONT        ;LOAD NEXT PIXEL
1174  004206  000771                            BR      1$              ;NEXT MAP LOAD
1175  004210  012700  007740            2$:     MOV     #7740,R0        ;SET UP BIT MAP ADRS - BOT LINE
1176  004214  004737  005634                    JSR     PC,DISPY        ;GO LOAD BIT MAP
1177  004220  005200                    3$:     INC     R0              ;ADVANCE ADRS
1178  004222  022700  010000                    CMP     #10000,R0       ;BOT LINE DONE?
1179  004226  001403                            BEQ     4$              ;BR IF SO
1180  004230  004737  005646                    JSR     PC,DCONT        ;LOAD NEXT PIXEL WORD
1181  004234  000771                            BR      3$              ;NEXT MAP LOAD
1182  004236  012700  000040            4$:     MOV     #40,R0          ;SET UP BIT MAP ADRS SIDE LINES
1183  004242  012701  000007            5$:     MOV     #7,R1           ;SET UP PIXEL 0 DATA IN R1 - INT 7
1184  004246  004737  005634                    JSR     PC,DISPY        ;GO LOAD BIT MAP
1185  004252  012701  070000                    MOV     #70000,R1       ;SET UP PIXEL 3 DATA IN R1 - INT 7
1186  004256  062700  000037                    ADD     #37,R0          ;OFFSET TO RIGHT SIDE LINE
1187  004262  004737  005634                    JSR     PC,DISPY        ;GO LOAD BIT MAP
1188  004266  005200                            INC     R0              ;GO TO NEXT ROW ON LEFT
1189  004270  022700  007740                    CMP     #7740,R0        ;TO BOTTOM YET?
1190  004274  001362                            BNE     5$              ;BR IF NOT
1191  004276  013777  001262  174716            MOV     VTVSAV,@VTVCSR  ;ENABLE BIT MAP
1192  004304  000466                            BR      DISBG           ;CONTINUE TO ANALOG DATA
1193
1194                                    ;DRAW BOX UP ON VT01 SCREEN - DIFF SIZE FOR AR11 JOYSTICK CALIBRATION
1195  004306  005737  001346            JBOX1:  TST     JTYPE           ;AR11 ANALOG SOURCE?
1196  004312  100432                            BMI     JBOX2           ;BR IF SO
```

```
1197  004314  005000                          CLR     R0              ;X=0
1198  004316  005001                          CLR     R1              ;Y=0
1199  004320  004737  005676          1$:     JSR     PC,DISPY1       ;DISPLAY POINT
1200  004324  005200                          INC     R0              ;ADVANCE X
1201  004326  022700  003761                  CMP     #3761,R0        ;OUT TO RIGHT LIMIT?
1202  004332  001372                          BNE     1$              ;BR IF NOT
1203  004334  005300                          DEC     R0              ;X=3760, Y=0
1204  004336  004737  005676          2$:     JSR     PC,DISPY1       ;DISPLAY POINT
1205  004342  005201                          INC     R1              ;ADVANCE Y
1206  004344  022701  003761                  CMP     #3761,R1        ;UP TO TOP YET?
1207  004350  001372                          BNE     2$              ;BR IF NOT
1208  004352  005301                          DEC     R1              ;X=3760, Y=3760
1209  004354  004737  005676          3$:     JSR     PC,DISPY1       ;DISPLAY POINT
1210  004360  005300                          DEC     R0              ;ADVANCE X
1211  004362  100374                          BPL     3$              ;BR IF NOT TO TOP LEFT
1212  004364  005200                          INC     R0              ;X=0, Y=3760
1213  004366  004737  005676          4$:     JSR     PC,DISPY1       ;DISPLAY POINT
1214  004372  005301                          DEC     R1              ;ADVANCE Y
1215  004374  100374                          BPL     4$              ;BR IF NOT TO BOT LEFT
1216  004376  000431                          BR      DISBG           ;CONTINUE TO ANALOG DATA
1217
1218                                  ;AR11 BOX IS LARGER
1219  004400  005000          JBOX2:  CLR     R0              ;X=0
1220  004402  005001                  CLR     R1              ;Y=0
1221  004404  004737  005676          1$:     JSR     PC,DISPY1       ;DISPLAY POINT
1222  004410  005200                          INC     R0              ;ADVANCE X
1223  004412  022700  003771                  CMP     #3771,R0        ;OUT TO RIGHT LIMIT?
1224  004416  001372                          BNE     1$              ;BR IF NOT
1225  004420  005300                          DEC     R0              ;X=3770, Y=0
1226  004422  004737  005676          2$:     JSR     PC,DISPY1       ;DISPLAY POINT
1227  004426  005201                          INC     R1              ;ADVANCE Y
1228  004430  022701  003771                  CMP     #3771,R1        ;UP TO TOP LIMIT?
1229  004434  001372                          BNE     2$              ;BR IF NOT
1230  004436  005301                          DEC     R1              ;X=3770, Y=3770
1231  004440  004737  005676          3$:     JSR     PC,DISPY1       ;DISPLAY POINT
1232  004444  005300                          DEC     R0              ;ADVANCE X
1233  004446  100374                          BPL     3$              ;BR IF NOT TO TOP LEFT
1234  004450  005200                          INC     R0              ;X=0, Y=3770
1235  004452  004737  005676          4$:     JSR     PC,DISPY1       ;DISPLAY POINT
1236  004456  005301                          DEC     R1              ;ADVANCE Y
1237  004460  100374                          BPL     4$              ;BR IF NOT TO BOT LEFT
1238
1239                                  ;NOW COLLECT ANALOG DATA FROM SELECTED SOURCE (NC11 OR AR11)
1240
1241  004462  005737  001346          DISBG:  TST     JTYPE           ;WHAT SOURCE?
1242  004466  100520                          BMI     DISBG1          ;BR IF AR11
1243  004470  012777  000032  174500          MOV     #32,@NCCSR      ;EXT, JOYSTICK & ENA ADC
1244  004476  012777  000010  174506  1$:     MOV     #10,@NCSFUN     ;CLR TIME OUT IF SET
1245  004504  012777  000004  174500          MOV     #4,@NCSFUN      ;START CONVERTERS
1246  004512  105777  174460          2$:     TSTB    @NCCSR          ;DONE?
1247  004516  100025                          BPL     3$              ;BR IF SO
1248  004520  032777  040000  174450          BIT     #40000,@NCCSR   ;CONVERSION FAIL TO FINISH?
1249  004526  001771                          BEQ     2$              ;BR IF NOT
1250  004530  022737  000007  001306          CMP     #7,KBUFF        ;TURN ON/OFF BELL?
1251  004536  001004                          BNE     10$             ;BR IF NOT
1252  004540  005037  001306                  CLR     KBUFF           ;CLR OUT BELL CODE
```

```
004544  005137  001354                       COM     BELLEN          ;YES, DO IT
004550  005737  001354               10$:    TST     BELLEN          ;SOUND BELL ON NO CONVERTS
004554  001350                               BNE     1$              ;BR IF BELL NOT DESIRED
004556  012737  000207  001310               MOV     #207,TTYOUT     ;SET UP BELL CODE
004564  004737  006174                       JSR     PC,TYPO         ;GO ZING BELL
004570  000742                               BR      1$              ;GO TRY AGAIN
004572  017737  174404  001246       3$:     MOV     @NCAREG,TEMPO   ;STORE CONVERSION VALUES
004600  042737  100200  001246               BIC     #100200,TEMPO   ;MASK OFF BITS 7,15
004606  004737  007210                       JSR     PC,XYAVE        ;GO AVG LAST 32. X-Y JOYSTICK VALUES
004612  012777  000010  174372               MOV     #10,@NCSFUN     ;CLEAR JOYSTICK DEPRESS FLAG
004620  005777  174352                       TST     @NCCSR          ;LOOK FOR BUTTON DOWN
004624  100004                               BPL     4$              ;BR IF NOT
004626  005737  001254                       TST     DTYPE           ;DON'T MOVE BUG BUT REFRESH
004632  001025                               BNE     5$              ;REFRESH VT01 ONLY
004634  000720                               BR      1$              ;NO NEED ON VSV01
004636  006337  001246               4$:     ASL     TEMPO           ;MUL BY 2
004642  005737  001254                       TST     DTYPE           ;ALL DONE SCALING IF VSV01
004646  001425                               BEQ     6$              ;BR IF SO
004650  005137  001246                       COM     TEMPO           ;INVERT DATA FOR VT01
004654  042737  000401  001246               BIC     #401,TEMPO      ;RID GARBAGE
004662  113700  001246                       MOVB    TEMPO,R0        ;GET X
004666  113701  001247                       MOVB    TEMPO+1,R1      ;GET Y
004672  006300                               ASL     R0              ;VT01 REQUIRES X & Y TO BE INFLATED BY 16
004674  006300                               ASL     R0              ;
004676  006300                               ASL     R0              ;
004700  006301                               ASL     R1              ;
004702  006301                               ASL     R1              ;
004704  006301                               ASL     R1              ;
004706  012777  000020  174320       5$:     MOV     #20,@VTCSR      ;SELECT WRITE THRU MODE
004714  004737  005676                       JSR     PC,DISPY1       ;GO DISPLAY THE BUG - VT01
004720  000666                               BR      1$              ;DO AGAIN
004722  004737  005726               6$:     JSR     PC,DISPY2       ;GO DISPLAY CROSS HAIRS - VSV01
004726  000663                               BR      1$              ;DO ANOTHER CONVERSION

                                     ;THIS CODE LOOKS AT THE AR11 FOR ANALOG DATA
004730                               DISBG1:
004730  012737  020000  001350       1$:     MOV     #20000,ARCHAN   ;SELECT CHAN 0 & UNIPOLAR
004736  004737  006024                       JSR     PC,ARCONV       ;GO START A/D - CHAN 0
004742  010337  001246                       MOV     R3,TEMPO        ;SAVE Y
004746  105237  001351                       INCB    ARCHAN+1        ;SELECT CHAN 1
004752  004737  006024                       JSR     PC,ARCONV       ;GO START A/D - CHAN 1
004756  010337  001250                       MOV     R3,TEMP1        ;SAVE X
004762  105237  001351                       INCB    ARCHAN+1        ;SELECT CHAN 2
004766  004737  006024                       JSR     PC,ARCONV       ;GO START A/D - CHAN 2
004772  010337  001252                       MOV     R3,TEMP2        ;READ JOYSTICK BUTTON INDICATOR
004776  000337  001246                       SWAB    TEMPO           ;PUT Y IN HI BYTE
005002  113737  001250  001246               MOVB    TEMP1,TEMPO     ;AND X IN LO BYTE
005010  004737  007210                       JSR     PC,XYAVE        ;GO AVG LAST 32. X-Y JOYSTICK VALUES
005014  105737  001252                       TSTB    TEMP2           ;LOOK FOR BAR DOWN (ZERO COND)
005020  001004                               BNE     2$              ;BR IF NOT DEPRESSED
005022  005737  001254                       TST     DTYPE           ;DON'T MOVE POINT BUT REFRESH
005026  001023                               BNE     3$              ;REFRESH VT01
005030  000737                               BR      1$              ;NO NEED ON VSV01
005032  005737  001254               2$:     TST     DTYPE           ;EACH DISPLAY SCALES DIFFERENTLY
005036  001422                               BEQ     4$              ;BR IF VSV01
005040  005137  001246                       COM     TEMPO           ;INVERT FOR VT01
```

C03


```
005044  113700  001246                      MOVB    TEMPO,R0        ;GET X
005050  113701  001247                      MOVB    TEMPO+1,R1      ;GET Y
005054  006300                              ASL     R0              ;SCALE X BY 8
005056  006300                              ASL     R0              ;
005060  006300                              ASL     R0              ;
005062  006301                              ASL     R1              ;SCALE Y BY 8
005064  006301                              ASL     R1              ;
005066  006301                              ASL     R1              ;
005070  012737  000020  174136              MOV     #20,@VTCSR      ;SELECT WRITE THRU MODE
005076  004737  005676          3$:         JSR     PC,DISPY1       ;GO DISPLAY THE BUG - [illegible]
005102  000772                              BR      1$              ;DO AGAIN
005104  004737  005726          4$:         JSR     PC,DISPY2       ;GO DISPLAY CROSS HAIRS - [illegible]
005110  [illegible]                         BR      1$              ;DO AGAIN
```

```
                                        .SBTTL  PROGRAM SUBROUTINES
                                ;;**************************************************************
                                ;ROUTINE SELECTS VSV01 DISPLAY - SETS UP BUS ADRS AND INTENSITY LEVEL
                                ;AND INTENSITY LOOK-UP TABLE - THE CARRY BIT IS SET ON EXIT IF THE
                                ;VSV01 IS NOT SEEN AT THE ASSIGNED BUS ADDRESS
                                ;
                                ;;**************************************************************
005112  013700  001166          SELCTA: MOV     VTVADR,R0       ;GET VSV01 BASE ADRS
005116  012701  001214                  MOV     #VTVCRG,R1      ;GET PTR ADRS
005122  010021                  1$:     MOV     R0,(R1)+        ;SET UP REG ADRS PTRS
005124  062700  000002                  ADD     #2,R0           ;BUMP REG ADRS
005130  022701  001222                  CMP     #VTVCSR,R1      ;CHAR GEN REGS ALL SET UP?
005134  001372                          BNE     1$              ;BR IF NOT
005136  013700  001170                  MOV     VTMADR,R0       ;GET BASE ADRS OF BIT MAP
005142  005737  001352                  TST     MSELCT          ;USING SECOND MAP?
005146  001402                          BEQ     2$              ;BR IF NOT
005150  062700  000020                  ADD     #20,R0          ;POINT TO 2ND BIT MAP ADRS'S
005154  010021                  2$:     MOV     R0,(R1)+        ;CONTINUE TO BIT MAP ADRS'S
005156  062700  000002                  ADD     #2,R0           ;BUMP REG ADRS
005162  022701  001234                  CMP     #VTCSR,R1       ;ALL SET UP?
005166  001372                          BNE     2$              ;BR IF NOT
005170  012737  005254  000004          MOV     #5$,@#ERRVEC    ;SET UP BUS TIMEOUT RETURN ADRS IF NO VSV01
005176  005777  174012                  TST     @VTVCRG         ;IS CHARACTER GENERATOR THERE?
005202  005777  174014                  TST     @VTVCSR         ;IS BIT MAP THERE?
005206  013700  001260                  MOV     INTLUT,R0       ;SET UP ADRS & DATA OF INTENSITY LOOK-UP TABLE
005212  010077  174014          3$:     MOV     R0,@VTVINT      ;SET UP TABLE
005216  062700  000401                  ADD     #401,R0         ;ADVANCE ADRS & INTENSITY
005222  032700  010000                  BIT     #10000,R0       ;TABLE LOADED?
005226  001771                          BEQ     3$              ;BR IF NOT
005230  005037  001254                  CLR     DTYPE           ;DTYPE=0 SAYS VSV01
005234  012737  000020  001256          MOV     #16.,INTENS     ;MAX 16. INTENSITIES
005242  000241                          CLC                     ;ZERO CARRY SAYS VSV01 THERE
005244  012737  000004  000004  4$:     MOV     #ERRVEC,@#ERRVEC        ;RESTORE ERR VEC ADR LOC TO 0 TO 5
005252  000207                          RTS     PC              ;EXIT
005254  122626                  5$:     CMPB    (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
005256  000261                          SEC                     ;CARRY ON EXIT SAYS NO VSV01
005260  000771                          BR      4$              ;GO EXIT
                                ;;**************************************************************
                                ;ROUTINE SELECTS VTC1 DISPLAY - SETS UP BUS ADRS AND INTENSITY LEVEL -
                                ;THE CARRY BIT IS SET ON EXIT IF THE VTC1 IS NOT SEEN AT THE
                                ;ASSIGNED BUS ADDRESS
                                ;;**************************************************************
005262  013700  001172          SELCTB: MOV     VTADR,R0        ;GET VTC1 BASE ADRS
005266  012701  001234                  MOV     #VTCSR,R1       ;GET PTR ADRS
005272  010021                  1$:     MOV     R0,(R1)+        ;SET UP VTC1 REG ADRS PTRS
005274  062700  000002                  ADD     #2,R0           ;BUMP REG ADRS
005300  022701  001242                  CMP     #ARCSR,R1       ;ALL SET UP?
005304  001372                          BNE     1$              ;BR IF NOT
005306  012737  005350  000004          MOV     #3$,@#ERRVEC    ;SET UP FOR VTC1 TIMEOUT
005314  012777  000014  173712          MOV     #14,@VTCSR      ;SELECT STORE MODE
005322  012737  000040  001256          MOV     #32.,INTENS     ;MAX 32 INTENSITIES
005330  012737  177777  001254          MOV     #-1,CTYPE       ;CTYPE=-1 SAYS VTC1
005336  000241                          CLC                     ;ZERO CARRY SAYS VTC1 THERE
005340  012737  000006  000004  2$:     MOV     #ERRVEC+2,@#ERRVEC      ;RESTORE LOC 4 PTR TO 0 TO 5
005346  000207                          RTS     PC              ;EXIT
```

```
                                3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RETURN
                                        SEC                     ;CARRY SET SAYS NO VTO1 DISPLAY
                                        BR      2$              ;GO EXIT

                                ;;**********************************************************************
                                ;ROUTINE SELECTS AND SETS UP BUS ADRS FOR AR11 JOYSTICK CALIBRATION
                                ;THE CARRY SET ON EXIT IF THE AR11 IS NOT SEEN
                                ;;**********************************************************************
005356  013700  001174          SELCTC: MOV     ARADR,R0        ;GET AR11 BASE ADRS
005362  012701  001242                  MOV     #ARCSR,R1       ;GET PTR ADRS
005366  010021                  1$:     MOV     R0,(R1)+        ;SET UP REG ADRS PTRS
005370  062700  000002                  ADD     #2,R0           ;BUMP REG ADRS
005374  023701  001246                  CMP     #TEMP0,R1       ;ALL SET UP?
005400  001372                          BNE     1$              ;BR IF NOT
005402  012737  005434  000004          MOV     #3$,@#ERRVEC    ;SET UP TIMEOUT RETURN ADRS IF AR11 NOT THERE
005410  005077  173626                  CLR     @ARCSR          ;SEE IF THERE
005414  012737  177777  001346          MOV     #-1,JTYPE       ;JTYPE=-1 SAYS USE AR11 FOR JOYSTICK CAL
005422  000241                          CLC                     ;CARRY ZERO SAYS AR11 IS THERE
005424  012737  000006  000004  2$:     MOV     #ERRVEC+2,@#ERRVEC      ;RESTORE LOC 4 TO PT TO 6
005432  000207                          RTS     PC              ;EXIT
005434  000261                  3$:     SEC                     ;CARRY SET SAYS THAT AR11 IS NOT ADRS ASSIGNED
005436  000772                          BR      2$              ;GO EXIT

                                ;;**********************************************************************
                                ;ROUTINE STARTS NC11 AT SELECTED GAIN
                                ;;**********************************************************************
005440  012777  001001  173530  NCSTRT: MOV     #1001,@NCCSR    ;SET UP 64*64 MATRIX AND ADC ENABLE
005446  053777  001360  173522          BIS     GAIN,@NCCSR     ;SET UP GAIN
005454  052777  000002  173514          BIS     #BIT1,@NCCSR    ;SET GO BIT
005462  000207                          RTS     PC              ;RETURN

                                ;;**********************************************************************
                                ;ROUTINE STOPS NC11 AND SAVES NC11 STATUS
                                ;;**********************************************************************
005464  017737  173506  001316  NCSTP:  MOV     @NCCSR,COMSAV   ;SAVE THE INTERFACE ACTION
005472  042777  000003  173476          BIC     #3,@NCCSR       ;DISABLE NPR'S
005500  005077  173472                  CLR     @NCCSR          ;ZERO ALL STATUS
005504  000207                          RTS     PC              ;RETURN

                                ;;**********************************************************************
                                ;ROUTINE STOPS NC11, SAVES STATUS AND CLEARS MATRIX CORE AREA
                                ;;**********************************************************************
005506  004737  005464  NCSTP1: JSR     PC,NCSTP        ;GO STOP NC11 AND SAVE STATUS
005512  012777  000020  173472          MOV     #CLZ,@NCSFUN    ;ZERO Z REG COUNT
005520  012700  020000                  MOV     #MATRIX,R0      ;GET SET TO ZERO CORE MATRIX AREA
005524  005020                  1$:     CLR     (R0)+           ;ZERO LOC
005526  020027  060000                  CMP     R0,#MATRIX+40000        ;ALL DONE?
005532  001374                          BNE     1$              ;BR IF MORE
005534  000207                          RTS     PC              ;RETURN

                                ;;**********************************************************************
                                ;ROUTINE WILL ERASE DISPLAY (VSV01 OR VTO1)
                                ;;**********************************************************************
005536  005737  001254  ERASE:  TST     DTYPE           ;WHAT DISPLAY?
005542  100017                  BPL     2$              ;BR IF VSV01
005544  012777  000016  173462          MOV     #16,@VTCSR      ;ERASE DISPLAY
```

```
1434  005552  005037  001246                  CLR     TEMPO           ;SET UP VTO1 ERASE WAIT LOOPS
1435  005556  012737  000002  001250          MOV     #2,TEMP1        ;DO LOOP 2 TIMES
1436  005564  005337  001246          1$:     DEC     TEMPO           ;COUNT AWAY
1437  005570  001375                  BNE     1$              ;TILL 0
1438  005572  005337  001250          DEC     TEMP1           ;2ND LOOP DONE?
1439  005576  001372                  BNE     1$              ;BR IF NOT
1440  005600  000414                  BR      4$              ;ALL DONE - RETURN
1441  005602  052777  001000  173412  2$:     BIS     #1000,@VTVCSR   ;ERASE DISPLAY (CLR BIT MAP,
1442  005610  105777  173406          3$:     TSTB    @VTVCSR         ;LOOK FOR READY
1443  005614  100375                  BPL     3$              ;WAIT FOR IT
1444  005616  042777  001000  173376          BIC     #1000,@VTVCSR   ;TURN OFF ERASE DISPLAY
1445  005624  012777  002035  173362          MOV     #2035,@VTVCRG   ;CLR CHAR BUFFER & DISABLE CURSOR
1446  005632  000207                  4$:     RTS     PC              ;EXIT
1447
1448                                          ;;**************************************************************
1449                                          ;ROUTINE WILL LOAD BIT MAP (VSV01)
1450                                          ;R0 CONTAINS BIT MAP ADRS AND R1 THE BIT MAP DATA
1451                                          ;;**************************************************************
1452  005634  042777  000400  173360  DISPY:  BIC     #400,@VTVCSR    ;STOP DISPLAY
1453  005642  010077  173356          MOV     R0,@VTVMAP      ;LOAD BIT MAP ADRS
1454  005646  042777  000400  173346  DCONT:  BIC     #400,@VTVCSR    ;AGAIN IF ENTERING HERE
1455  005654  105777  173342          1$:     TSTB    @VTVCSR         ;READY?
1456  005660  100375                  BPL     1$              ;WAIT THEN
1457  005662  010177  173340          MOV     R1,@VTVPX       ;LOAD BIT MAP
1458  005666  052777  000400  173326          BIS     #400,@VTVCSR    ;RESUME DISPLAY
1459  005674  000207                  RTS     PC              ;RETURN
1460
1461                                          ;;**************************************************************
1462                                          ;ROUTINE WILL DISPLAY A POINT (VTO1)
1463                                          ;;**************************************************************
1464  005676  105777  173332          DISPY1: TSTB    @VTCSR          ;DISPLAY READY?
1465  005702  100375                  BPL     DISPY1          ;WAIT FOR IT
1466  005704  042700  174000          BIC     #174000,R0      ;RID GARBAGE
1467  005710  042701  174000          BIC     #174000,R1      ;RID GARBAGE
1468  005714  010077  173316          MOV     R0,@VTXDAC      ;SET UP X DAC
1469  005720  010177  173314          MOV     R1,@VTYDAC      ;SET UP Y DAC
1470  005724  000207                  RTS     PC              ;EXIT
1471
1472                                          ;;**************************************************************
1473                                          ;ROUTINE WILL DISPLAY X & Y CROSS HAIRS ON VSV01
1474                                          ;;**************************************************************
1475  005726  105777  173262          DISPY2: TST     @VTVCRG         ;READY?
1476  005732  100375                  BPL     DISPY2          ;WAIT IF NOT
1477  005734  105137  001246          COMB    TEMPO           ;X NEEDS TO BE INVERTED
1478  005740  013777  001246  173250  MOV     TEMPO,@VTVCHP   ;LOAD X & Y CROSS HAIRS
1479  005746  052777  016000  173240  BIS     #16000,@VTVCRG  ;ENABLE THE CROSS HAIRS
1480  005754  000207                  RTS     PC              ;RETURN
1481
1482                                          ;;**************************************************************
1483                                          ;ROUTINE WIIL FILL CORE WITH AN IMAGE THAT WHEN
1484                                          ;DISPLAYED WILL CONTAIN ALL THE INTENSITY LEVELS -
1485                                          ;ROWS AT THE BOTTOM OF THE SCREEN WILL APPEAR BRIGHTEST -
1486                                          ;NOTE THAT THIS IS ONLY A DISPLAY TEST PATTERN FOR
1487                                          ;POSSIBLE DISPLAY ADJUSTMENTS BY THE USER
1488                                          ;;**************************************************************
1489  005756  012700  000100          LDIMGE: MOV     #100,R0         ;COUNT 64 ROWS
```

```
1490  005762  013701  001340          MOV     TABLEX,R1       ;GET SELECTED ISOTOPE
1491  005766  012702  000100          MOV     #100.R2         ;COUNT 64 DATA POINTS PER ROW
1492  005772  012703  000100          MOV     #100,R3         ;100 WILL REPRESENT HIGHEST INTENSITY LEVEL(100-0)
1493  005776  010321          1$:     MOV     R3,(R1)+        ;LOAD CORE IMAGE
1494  006000  005302                  DEC     R2              ;DONE ROW?
1495  006002  001375                  BNE     1$              ;BR IF NOT
1496  006004  005300                  DEC     R0              ;DONE ROWS?
1497  006006  001405                  BEQ     2$              ;BR IF SO
1498  006010  162703  000001          SUB     #1,R3           ;LOWER NEXT CELL VALUE
1499  006014  012702  000100          MOV     #100.R2         ;RESET ROW LENGTH COUNTER
1500  006020  000766                  BR      1$              ;LOAD THIS ROW IMAGE
1501  006022  000207          2$:     RTS     PC              ;RETURN FOR DISPLAY
1502
1503
1504
1505                                  ;;**************************************************************
1506                                  ;ROUTINE WILL DO A CONVERSION AT CHAN # IN 'ARCHAN'
1507                                  ;EXIT WITH CONVERSION VALUE IN R3
1510                                  ;;**************************************************************
1511  006024  005777  173214  173204  ARCONV: TST     @ARBUF          ;CLR DONE FLAG
1512  006030  013777  001350  173204          MOV     ARCHAN,@ARCSR   ;LD CHAN & UNIPOLAR BITS
1513  006036  052777  000001  173176          BIS     #1,@ARCSR       ;START A/D
1514  006044  105777  173172          1$:     TSTB    @ARCSR          ;LOOK FOR DONE
1515  006050  100375                  BPL     1$              ;WAIT FOR IT
1516  006052  017703  173166                  MOV     @ARBUF,R3       ;GET VALUE
1517  006056  162703  000500                  SUB     #500,R3         ;OFFSET VALUE
1520  006062  032703  177400                  BIT     #177400,R3      ;SEE IF OUT OF RANGE
1521  006066  100402                  BMI     2$              ;BR IF TOO SMALL
1522  006070  001003                  BNE     3$              ;BR IF TO LARGE
1523  006072  000404                  BR      4$              ;NORMAL EXIT
1524  006074  005003          2$:     CLR     R3              ;LIMIT TO 0
1525  006076  000402                  BR      4$              ;GO EXIT
1526  006100  012703  000377          3$:     MOV     #377,R3         ;LIMIT TO 377
1527  006104  000207          4$:     RTS     PC              ;EXIT
1530
1531                                  ;;**************************************************************
1532                                  ;ROUTINE WILL DISPLAY A CHARACTER (VSV01)
1533                                  ;;**************************************************************
1534  006106  005777  173102          DCHAR:  TST     @VTVCRG         ;READY?
1535  006112  100375                  BPL     DCHAR           ;WAIT FOR IT
1536  006114  110377  173074          MOVB    R3,@VTVCRG      ;LOAD CHAR
1537  006120  000207                  RTS     PC              ;EXIT
1540
1541                                  ;;**************************************************************
1542                                  ;KEYBOARD INTERRUPT SERVICE ROUTINE
1543                                  ;;**************************************************************
1544  006122  017737  173020  001306  KBINT:  MOV     @$TKB,KBUFF     ;READ KEY BOARD
1545  006130  042737  000200  001306          BIC     #200,KBUFF      ;RID PARITY
1546  006136  022737  000003  001306          CMP     #3,KBUFF        ;CNTRL 'C'?
1547  006144  001002                  BNE     1$              ;BR IF NOT
1550  006146  000137  001712                  JMP     LISEN           ;ABORT WHATEVER $ LOOK FOR NEXT COMMAND
1551  006152  000002          1$:     RTI
1552
1553                                  ;;**************************************************************
1554                                  ;ROUTINE TYPES 'CR' AND 'LF' OR CHAR IN TTYOUT
1555                                  ;;**************************************************************
1556  006154  012737  000015  001310  TYPCR:  MOV     #15,TTYOUT              ;SET UP FOR A 'CR'
1557  006162  004737  006174          JSR PC,TYPO
```

```
1546  006166  012737  000012  001310          MOV #12,TTYOUT          ;SET UP FOR LF
1547  006174  105777  172750          TYPO:   TSTB @$TPS              ;WAIT FOR LAST CHARACTER
1548  006200  100375                          BPL TYPO
1549  006202  013777  001310  172742          MOV TTYOUT,@$TPB        ;SEND IT OUT
1550  006210  000207                          RTS PC
1551
1552                                  ;;*****************************************************************
1553                                  ;GET IS A SUBROUTINE TO GET AN X,Y COCRDINATE FROM THE
1554                                  ;CODED LEVEL MATRIX,DISPLAY IT AND EXIT.
1555                                  ;;*****************************************************************
1556  006212  111502          GET:    MOVB (R5),R2            ;MOVE A BYTE OF X AND Y
1557  006214  042702  177707          BIC #177707,R2          ;MASK ALL BUT X
1558  006220  006202                  ASR R2                  ;ABS VALUE IS 4*X
1559  006222  060200                  ADD R2,R0               ;ADD TO CORNER VALUE
1560  006224  112502                  MOVB (R5)+,R2           ;MOV SAME BYTE AND POKE POINTER
1561  006226  042702  177770          BIC #177770,R2          ;MASK ALL BUT Y
1562  006232  060201                  ADD R2,R1               ;YABS Y INC IS 3
1563  006234  006302                  ASL R2                  ;SO DO Y=Y*3
1564  006236  060201                  ADD R2,R1               ;ADD TO CORNER VALUE
1565  006240  004737  005576          JSR     PC,DISPY1       ;GO DISPLAY POINT
1566  006244  000207                  RTS PC                  ;RETURN TO MAIN
1567
1568                                  ;;*****************************************************************
1569                                  ;THIS SUBROUTINE IS CALLED WITH THE ADDRESS OF A
1570                                  ;MESSAGE TABLE FOLLOWING THE CALL. THE MESSAGE
1571                                  ;TABLE CONTAINS A LIST OF POINTERS TO PHRASES.
1572                                  ;EACH PHRASE IS TERMINATED WITH A 200 BYTE. THE TABLE
1573                                  ;IS TERMINATED WITH A 0.
1574                                  ;;*****************************************************************
1575
1576  006246  017605  000000          VTWRIT: MOV @(SP),R5            ;GET ADDRESS OF MESSAGE IN R5
1577  006252  062716  000002                  ADD #2,(SP)             ;ADJUST RETURN
1578  006256  005715                  VTPL:   TST (R5)                ;LOOK FOR ZERO AS TERMINATOR
1579  006260  001476                          BEQ VTWDON              ;BR IF MSG DONE
1580  006262  012504                          MOV (R5)+,R4            ;ADDRESS OF PHRASE TO R4
1581  006264  112403                  VTWL:   MOVB (R4)+,R3           ;THIS IS LETTER TO BE DISPLAYED
1582  006266  100773                          BMI     VTPL            ;200 IS THE TERMINATOR
1583  006270  005737  001254                  TST     DTYPE           ;WHAT DISPLAY?
1584  006274  001003                          BNE     DVTO1           ;BR IF VTO1
1585  006276  004737  006106                  JSR     PC,DCHAR        ;GO DISPLAY CHAR VSVO1
1586  006302  000770                          BR      VTWL            ;GO LOOK AT NEXT CHAR
1587  006304  022703  000015          DVTO1:  CMP     #15,R3          ;CR?
1588  006310  001004                          BNE     1$              ;BR IF NOT
1589  006312  012737  000000  001326          MOV     #0,BASX         ;RESET MARGIN
1590  006320  000761                          BR      VTWL            ;GO GET NEXT CHAR
1591  006322  022703  000012          1$:     CMP     #12,R3          ;LF?
1592  006326  001004                          BNE     VTEXCP          ;BR IF NOT
1593  006330  162737  000060  001330          SUB     #60,BASY        ;GO TO NEXT LINE
1594  006336  000752                          BR      VTWL            ;GO GET NEXT CHAR
1595  006340  110337  001304          VTEXCP: MOVB R3,KBDBUF          ;THERE ARE A FEW CHARACTERS
1596  006344  012737  000034  006460          MOV #34,TEMCHR          ;THAT REQUIRE CONVERSION
1597  006352  004737  006462                  JSR PC,BRAN             ;BEFORE DISPLAY.
1598  006356  006373                          VTEXCT-1
1599  006360  006430                          CH?4                    ;CHARCTER IS
```

```
 1600  006362  006422                          CH75                    ;=
 1601  006364  006416                          CH72                    ;:
 1602  006366  006414                          CH76                    ;>
 1603  006370  006402                          CH77                    ;?
 1604  006372  000420                          BR VTNOSP               ;CHARACTER IS OK AS IS
 1605  006374     074     075     072  VTEXCT: .BYTE 74.75.72.76.77.0
 1606  006377     076     077     000
 1607                                          .EVEN
 1608  006402  062737  000004  006460  CH77:   ADD #4,TEMCHR           ;CONVERT OCTAL TO 42
 1609  006410  062737  000007  006460  CH76:   ADD #7,TEMCHR           ;CONVERT TO 46
 1610  006416  005237  006460          CH72:   INC TEMCHR              ;CONVERT TO 37
 1611  006422  062737  000002  006460  CH75:   ADD #2,TEMCHR           ;CONVERT TO 36
 1612  006430  013703  006460          CH74:   MOV TEMCHR,R3           ;R3 NOW CONTAINS PROPER VALUE
 1613  006434  042703  177700          VTNOSP: BIC #177700,R3          ;CLEAR GARBAGE
 1614  006440  006303                          ASL R3                  ;MULTIPLY 6 BIT OCTAL BY 4
 1615  006442  006303                          ASL R3                  ;TO FIND RELATIVE POSITION IN TABLE
 1616  006444  062703  012652                  ADD #TABLE-4,R3         ;THIS IS ABSOLUTE POSITION
 1617  006450  004737  006532                  JSR PC,TIXDIS           ;DRAW CHARACTER
 1618  006454  000703                          BR VTWL                 ;AND CONSIDER NEXT
 1619  006456  000207                  VTWDON: RTS PC
 1620  006460  000000                  TEMCHR: 0                       ;SPECIAL CHARS ARE BUILT HERE
 1621
 1622                                  ;;*****************************************************************
 1623                                  ;THIS  ROUTINE COMPARES THE CHARACTER IN KBDBUF (BYTE)
 1624                                  ;AGAINST A LIST POINTED TO BY CONTENTS+1 OF CALL+2.
 1625                                  ;IF A MATCH IS FOUND, CONTROL IS TRANSFERED TO ADDRESS
 1626                                  ;CONTAINED IN CALL +2+2N WHERE N IS THE NUMBER OF
 1627                                  ;THE ENTRY IN THE MATCH TABLE THAT MATCHED.
 1628                                  ;IF NO MATCH IS FOUND, CONTOL IS TRANSFERED TO THE LOCATION FOLLOWING
 1629                                  ;THE LAST TRANSFER ADDRESS.
 1630                                  ;A ZERO IS THE MATCH TABLE TERMINATOR.
 1631                                  ;THE N AND C BITS ARE CLEARED ON EXIT
 1632                                  ;;*****************************************************************
 1633  006462  017637  000000  006530  BRAN:   MOV @(SP),MACHER         ;GET ADDRESS OF MATCH LIST
 1634  006470  062716  000002          BRANL:  ADD #2,(SP)             ;ADJUST RETURN POINTER
 1635  006474  005237  006530                  INC MACHER              ;MOVE MATCH POINTER
 1636  006500  105777  000024                  TSTB @MACHER            ;A ZERO INDICATES END OF LIST
 1637  006504  001406                          BEQ NOMACH
 1638  006506  123777  001304  000014          CMPB KBDBUF,@MACHER
 1639  006514  001365                          BNE BRANL               ;NO MATCH. TRY AGAIN
 1640  006516  017616  000000                  MOV @(SP),(SP)          ;PUT TRANSFER ADDRESS ON STACK
 1641  006522  000241                  NOMACH: CLC                     ;CLEAR CARRY BIT
 1642  006524  000250                          CLN                     ;AND N BIT
 1643  006526  000207                          RTS PC                  ;RETURN APPROPRIATELY
 1644  006530  000000                  MACHER: 0
 1645
 1646                                  ;;*****************************************************************
 1647                                  ;THIS SUBROUTINE IS USED TO DISPLAY A MATRIX OF POINTS
 1648                                  ;THE ROUTINE IS GENERALIZED AND MUST HAVE VARIOUS
 1649                                  ;CONSTANTS SETUP BY THE CALLING PROGRAMS.
 1650                                  ;R3 POINTS TO THE MATRIX CODE WORDS
 1651                                  ;;*****************************************************************
 1652
 1653
 1654  006532  013737  006660  006662  TIXDIS: MOV SHFTK,SHFTCT         ;SET SHIFT COUNT FOR DECODER
 1655  006540  013737  006664  006666          MOV ROWK,ROW            ;SETUP THE NUMBER OF ROWS
```

```
1656  006546  013737  006670  006672          MOV COLK,COL            ;AND THE NUMBER OF COLUMNS
1657  006554  012302                          MOV (R3)+,R2            ;GET THE WORD TO BE DECODED
1658  006556  013700  001326                  MOV BASX,R0             ;R0 WILL CONTAIN THE X COORD.
1659  006562  013701  001330                  MOV BASY,R1             ;R1 NOW CONTAINS THE Y COORDINATE
1660  006566  006002                  TIXL:   ROR R2                  ;CHECK CODE WORD FOR BIT
1661  006570  103002                          BCC TIXBNO              ;IF CARRY BIT CLEAR NO POINT TO DISPLAY
1662  006572  004737  005676                  JSR     PC,DISPY1       ;GO DISPLAY POINT
1663  006576  005337  006662          TIXBNO: DEC SHFTCT              ;COUNT INFO BITS IN WORD
1664  006602  001001                          BNE TIXWOK              ;CODE WORD OK IF NON ZERO RESULT
1665  006604  011302                          MOV (R3),R2             ;GET NEW CODE WORD
1666  006606  063700  001332          TIXWOK: ADD INCXY,R0            ;REFERENCE NEXT POINT
1667  006612  005337  006666                  DEC ROW                 ;DONT GO PAST END OF ROW
1668  006616  001363                          BNE TIXL                ;NON ZERO OK
1669  006620  013737  006664  006666          MOV ROWK,ROW            ;RESET
1670  006626  063701  001332                  ADD INCXY,R1            ;MOV Y UP A ROW
1671  006632  005337  006672                  DEC COL                 ;WHEN COL=0 WE ARE DONE
1672  006636  001403                          BEQ TIXDON
1673  006640  013700  001326                  MOV BASX,R0             ;RESET X TO LEFT HAND EDGE
1674  006644  000750                          BR TIXL
1675  006646  063700  001332          TIXDON: ADD INCXY,R0            ;MAKE A SPACE BETWEEN LETTERS
1676  006652  010037  001326                  MOV R0,BASX             ;AND RESET BASX TO NEW VALUE
1677  006656  000207                          RTS PC
1678  006660  000017                  SHFTK:  15.
1679  006662  000000                  SHFTCT: 0
1680  006664  000005                  ROWK:   5
1681  006666  000000                  ROW:    0
1682  006670  000006                  COLK:   6
1683  006672  000000                  COL:    0
1684
1685                                  ;;*****************************************************************
1686                                  ;THIS ROUTINE CALLS 'ASCIZP WHICH CONVERTS A SINGLE OR DOUBLE
1687                                  ;PRECISION OCTAL NUMBER TO ASCII DECIMAL AND THEN WRITES IT TO SCREEN
1688                                  ;;*****************************************************************
1689  006674  007105                  BLANK:  ADUMMY
1690  006676  000000                          0
1691  006700  005037  006726          AWRIT:  CLR DUMMY2
1692  006704  004737  006732          OTTY:   JSR PC,ASCZSP
1693  006710  006724                          DUMMY1
1694  006712  007105                          ADUMMY
1695  006714  004737  006246                  JSR PC,VTWRIT
1696  006720  006674                          BLANK
1697  006722  000207                          RTS PC
1698
1699                                  ;;*****************************************************************
1700                                  ;THIS ROUTINE CONVERTS A DOUBLE PRECISION OCTAL NUMBER IN CORE
1701                                  ;TO AN ASCII DECIMAL NUMBER.
1702                                  ;THE CALL IS JSR PC,ASCIZ
1703                                  ;WITH THE ADDRESS OF THE LOW ORDER OCTAL WORD IN
1704                                  ;CALL+2
1705                                  ;AND THE ADDRESS OF THE PLACE THE ASCII DECIMAL IS TO BE STORED
1706                                  ;IN CALL+4
1707                                  ;THE HIGH ORDER OCTAL WORD MUST BE IN THE LOCATION FOLLOWING
1708                                  ;THE LOW ORDER WORD.
1709                                  ;THE SUBROUTINE SHOULD BE CALLED BY
1710                                  ;JSR PC,ASCZSP
1711                                  ;TO SUPPRESS LEADING ZEROES AND TO LEFT JUSTIFY THE
```

```
                                        ;ANSWER. UNLESS THIS LAST CALL IS MADE THE OUTPUT
                                        ;WILL ALWAYS BE 11 BYTES.
                                        ;*****************************************************************
006724  000000                DUMMY1: 0
006726  000000                DUMMY2: 0
006730  000000                ZESUP:  0                       ;ZERO SUPPRESS FLAG
006732  005237  006730        ASCZSP: INC ZESUP               ;SET FLAG
006736  017601  000000        ASCIZ:  MOV @(SP),R1            ;GET LOW ORDER ADDRESS
006742  062716  000002                ADD #2,(SP)             ;ADJUST POINTER
006746  012103                        MOV (R1)+,R3            ;GET LOW ORDER
006750  011102                        MOV (R1),R2             ;GET HIGH ORDER
006752  012737  000012  007070        MOV #10.,DIVCNT         ;10 POSSIBLE DIGITS
006760  012705  007072                MOV #ASCRES,R5          ;ADDRESS OF DIGITS
006764  112725  000200                MOVB #200,(R5)+         ;CHGEN TERMINATOR
006770  012704  000012        ASCLP:  MOV #10.,R4             ;DIVISOR
006774  010546                        MOV R5,-(SP)            ;SAVE R5 DURING DIVISION
006776  004737  007120                JSR PC,DIVIDE           ;DIVIDE BY 10
007002  012605                        MOV (SP)+,R5            ;RESTORE R5
007004  062702  000060                ADD #60,R2              ;MAKE IT ASCII
007010  110225                        MOVB R2,(R5)+           ;PUT IT IN THE ASCII STRING
007012  010103                        MOV R1,R3
007014  010002                        MOV R0,R2               ;PUT ANSWER IN DIVIDEND SLOT
007016  005337  007070                DEC DIVCNT
007022  001362                        BNE ASCLP
                                ;NOW TRANSFER THE ASCII TO PROPER PLACE IN CORRECT ORDER
007024  005737  006730                TST ZESUP               ;SUPPRESS ZEROES?
007030  001410                        BEQ MOVALL              ;NO MOVE ALL
007032  005037  006730                CLR ZESUP               ;INIT THE SWITCH
007036  124527  000060        SKZE:   CMPB -(R5),#60          ;LOOK FOR ZEROES
007042  001775                        BEQ SKZE                ;AND PASS BY THEM
007044  105725                        TSTB (R5)+              ;BACK-UP
007046  100001                        BPL MOVALL              ;AT LEAST ONE ZERO
007050  005205                        INC R5                  ;POINT TO FIRST NUMBER
007052  017600  000000        MOVALL: MOV @(SP),R0            ;ADDRESS OF RESULTS
007056  062716  000002                ADD #2,(SP)             ;ADJUST RETURN
007062  114520          MOVAL:  MOVB -(R5),(R0)+
007064  100376                        BPL MOVAL               ;200 BYTE IS TERMINATOR
007066  000207                        RTS PC
007070  000000          DIVCNT: 0
007072  007105          ASCRES: .=.+11.
007105  007120          ADUMMY: .=.+11.
                                        .EVEN


                                        ;*****************************************************************
                                        ;THIS ROUTINE DIVIDES A POSITIVE DOUBLE PRECISION NUMBER
                                        ;FOUND IN R2 (HI) AND R3 (LO) BY THE POSITIVE NUMBER IN
                                        ;R4. THE ANSWER IS PLACED IN R0 (HI) AND R1 (LO).
                                        ;THE ROUTINE LEFT JUSTIFIES DIVISOR KEEPING TRACK OF SHIFTS
                                        ;SO THAT FIRST BIT IS IN BIT 14 OF R4. THE SHIFT COUNT IS THEN
                                        ;ADDED TO 16. AND THE DIVISION IS STARTED BY COMPARING THE
                                        ;DIVISOR (SHIFTED) TO THE HI ORDER OF THE DIVIDEND.
                                        ;IF COMPARISON SHOWS DIVISOR SMALLER, A SUBTRACTION IS DONE
                                        ;BETWEEN DIVSOR AND HI ORDER DIVIDEND AND THE ANSWER IS INCREMENTED.
                                        ;REGUARDLESS OF COMPARISON RESULT, BOTH THE ANSWER AND
                                        ;THE DIVIDEND ARE MULTIPLIED BY 2.
                                        ;THIS SEQUENCE IS PERFORMED THE NUMBER OF TIMES INDICATED BY
```

```
                                        ;SHIFTCOUNT.
                                        ;R2R3 / R4=ROR1
                                        ;;*****************************************************************
007120  005005                  DIVIDE: CLR R5                  ;INIT SHIFT COUNT
007122  005000                          CLR R0
007124  005001                          CLR R1                  ;ZERO THE ANSWER
007126  005205                  DIVJUS: INC R5                  ;THIS IS THE SHIFT COUNT
007130  006304                          ASL R4                  ;DIVSOR X 2
007132  100375                          BPL DIVJUS              ;GO UNTIL BIT 15 SETS
007134  006004                          ROR R4                  ;MOV IT BACK ONE
007136  010537  007206                  MOV R5,RMJUST           ;SAVE SIFT COUNT FOR REMAN. LST
007142  062705  000020                  ADD #16.,R5             ;SIMULATE 16 BIT SHIFT
007146  006301                  DIVLOO: ASL R1                  ;ANSWER X 2
007150  006100                          ROL R0
007152  020204                          CMP R2,R4               ;COMPARE DIVISOR AND DIVIDEND
007154  100402                          BMI NOSUB               ;IF DIVSOR LARGER NO SUBTRCTION
007156  160402                          SUB R4,R2
007160  005201                          INC R1                  ;NOTE SUBTRACTION IN ANSWER
007162  006303                  NOSUB:  ASL R3                  ;MAKE DIVIDEND LARGER
007164  006102                          ROL R2                  ;BY 2
007166  005305                          DEC R5                  ;DO ALL THIS SHIFCNT TIMES
007170  001366                          BNE DIVLOO
007172  000241                          CLC
007174  006002                  RLJL:   ROR R2                  ;NOW RIGHT JUSTIFY THE REMAINDER
007176  005337  007206                  DEC RMJUST
007202  001374                          BNE RLJL
007204  000207                          RTS PC
007206  000000                  RMJUST: 0

                                        ;;*****************************************************************
                                        ;THIS ROUTINE AVERAGES THE LAST 32. X-Y JOYSTICK VALUES
                                        ;;*****************************************************************
007210  010046                  XYAVE:  MOV     R0,-(SP)        ;SAVE R0
007212  013700  001246                  MOV     TEMPO,R0        ;GET X & Y
007216  013702  007436                  MOV     XYBUFP,R2       ;GET CURRENT BUFFER POINTER
007222  010022                          MOV     R0,(R2)+        ;SAVE NEW X-Y VALUE
007224  020227  007436                  CMP     R2,#XYBUFE      ;END OF BUFFER?
007230  103402                          BLO     1$              ;NO
007232  012702  007336                  MOV     #XYBUF,R2       ;YES, GO BACK TO BEGINING OF BUFFER
007236  010237  007436          1$:     MOV     R2,XYBUFP       ;SAVE NEW BUFFER POINTER
007242  012702  007336                  MOV     #XYBUF,R2       ;CALC AVE X
007246  004737  007300                  JSR     PC,10$
007252  110046                          MOVB    R0,-(SP)        ;SAVE IT
007254  012702  007337                  MOV     #XYBUF+1,R2     ;CALC AVE Y
007260  004737  007300                  JSR     PC,10$          ;GO DO IT
007264  000300                          SWAB    R0              ;GET X-Y IN R0
007266  152600                          BISB    (SP)+,R0        ;GET SAVED X
007270  010037  001246                  MOV     R0,TEMPO        ;PUT IN TEMPO
007274  012600                          MOV     (SP)+,R0        ;RESTORE R0
007276  000207                          RTS     PC              ;EXIT WITH AVE X-Y IN TEMPO

007300  005000                  10$:    CLR     R0              ;ZERO SUM
007302  005005                  11$:    CLR     R5              ;DO A MOVB TO A REG (UNSIGNED)
007304  152205                          BISB    (R2)+,R5
007306  060500                          ADD     R5,R0           ;ADD IT IN
007310  005202                          INC     R2              ;SKIP OTHER VALUE
```

```
1824  007312  020227  007436          CMP     R2,#XYBUFE      ;END OF BUFFER?
1825  007316  103771                  BLO     11$             ;NO
1826  007320  006300                  ASL     R0              ;DIVIDE BY 32.
1827  007322  006300                  ASL     R0
1828  007324  006300                  ASL     R0
1829  007326  000300                  SWAB    R0
1830  007330  042700  177400          BIC     #177400,R0      ;CLR HI BYTE
1831  007334  000207                  RTS     PC
1832
1833  007336                  XYBUF:
1834  007336  000000                  0
1835  007340  000000                  0
1836  007342  000000                  0
1837  007344  000000                  0
1838  007346  000000                  0
1839  007350  000000                  0
1840  007352  000000                  0
1841  007354  000000                  0
1842  007356  000000                  0
1843  007360  000000                  0
1844  007362  000000                  0
1845  007364  000000                  0
1846  007366  000000                  0
1847  007370  000000                  0
1848  007372  000000                  0
1849  007374  000000                  0
1850  007376  000000                  0
1851  007400  000000                  0
1852  007402  000000                  0
1853  007404  000000                  0
1854  007406  000000                  0
1855  007410  000000                  0
1856  007412  000000                  0
1857  007414  000000                  0
1858  007416  000000                  0
1859  007420  000000                  0
1860  007422  000000                  0
1861  007424  000000                  0
1862  007426  000000                  0
1863  007430  000000                  0
1864  007432  000000                  0
1865  007434  000000                  0
1866          007436          XYBUFE=.
1867  007436  007336          XYBUFP: XYBUF
1868
1869                          ;;*****************************************************************
1870                          .SBTTL  TTY INPUT ROUTINE
1871
1872                          ;;*****************************************************************
1873                          .ENABL  LSB
1874
1875                          ;;*****************************************************************
1876                          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
1877                          ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
1878                          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
1879                          ;*WHEN OPERATING IN TTY FLAG MODE.
```

```
1880  007440  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
1881  007446  001074                          BNE     15$             ;;BRANCH IF NO
1882  007450  105777  171470                  TSTB    @$TKS           ;;CHAR THERE?
1883  007454  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
1884  007456  117746  171464                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
1885  007462  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
1886  007466  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
1887  007472  001062                          BNE     15$             ;;NO, RETURN TO USER
1888  007474  123727  001134  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
1889  007502  001456                          BEQ     15$             ;;BRANCH IF YES
1890
1891  007504  104400  010312                  TYPE    .$CNTLG         ;;ECHO THE CONTROL-G (^G)
1892  007510  104400  010317          $GTSWR: TYPE    .$MSWR          ;;TYPE CURRENT CONTENTS
1893  007514  013746  000176                  MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
1894  007520  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1895  007522  104400  010330                  TYPE    .$MNEW          ;;PROMPT FOR NEW SWR
1896  007526  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
1897  007530  005046                          CLR     -(SP)           ;;THE NEW SWR
1898  007532  105777  171406          7$:     TSTB    @$TKS           ;;CHAR THERE?
1899  007536  100375                          BPL     7$              ;;IF NOT TRY AGAIN
1900
1901  007540  117746  171402                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
1902  007544  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
1903
1904
1905
1906  007550  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
1907  007554  001005                          BNE     10$             ;;BRANCH IF NOT
1908  007556  104400  010305                  TYPE    .$CNTLU         ;;YES, ECHO CONTROL-U (^U)
1909  007562  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
1910  007566  000757                          BR      19$             ;;LET'S TRY IT AGAIN
1911
1912
1913  007570  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
1914  007574  001022                          BNE     16$             ;;BRANCH IF NO
1915  007576  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
1916  007602  001403                          BEQ     11$             ;;BRANCH IF YES
1917  007604  016677  000002  171326          MOV     2(SP),@SWR      ;;SAVE NEW SWR
1918  007612  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
1919  007616  104400  001161          14$:    TYPE    .$CRLF          ;;ECHO <CR> AND <LF>
1920  007622  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
1921  007630  001003                          BNE     15$             ;;BRANCH IF NOT
1922  007632  012777  000100  171304          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
1923  007640  000002                  15$:    RTI                     ;;RETURN
1924  007642  004737  010512          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
1925  007646  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
1926  007652  002420                          BLT     18$             ;;BRANCH IF YES
1927  007654  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
1928  007660  003015                          BGT     18$             ;;BRANCH IF YES
1929  007662  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
1930  007666  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
1931  007672  001403                          BEQ     17$             ;;BRANCH IF YES
1932  007674  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
1933  007676  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
1934  007700  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
1935  007702  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
```

B04


```
        007706  056616  177776                          BIS     -2(SP),(SP)             ;;SET IN NEW CHAR
        007712  000767                                  BR      7$                      ;;GET THE NEXT ONE
        007714  104403  001160          18$:            TYPE    ,$QUES                  ;;TYPE ?(CR)(LF)
        007720  000720                                  BR      20$                     ;;SIMULATE CONTROL-U
                                                .DSABL  LSB

                                                ;;*************************************************************
                                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                                ;*CALL:
                                                ;*      RDCHR                           ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                                ;*      RETURN HERE                     ;;CHARACTER IS ON THE STACK
                                                ;*                                      ;;WITH PARITY BIT STRIPPED OFF
                                                ;

        007722  011646                          $RDCHR: MOV     (SP),-(SP)              ;;PUSH DOWN THE PC
        007724  016666  000004  000002                  MOV     4(SP),2(SP)             ;;SAVE THE PS
        007732  105777  171206                  1$:     TSTB    @$TKS                   ;;WAIT FOR
        007736  100375                                  BPL     1$                      ;;A CHARACTER
        007740  117766  171202  000004                  MOVB    @$TKB,4(SP)             ;;READ THE TTY
        007746  042766  177600  000004                  BIC     #^C<177>,4(SP)          ;;GET RID OF JUNK IF ANY
        007754  026627  000004  000023                  CMP     4(SP),#23               ;;IS IT A CONTROL-S?
        007762  001013                                  BNE     3$                      ;;BRANCH IF NO
        007764  105777  171154                  2$:     TSTB    @$TKS                   ;;WAIT FOR A CHARACTER
        007770  100375                                  BPL     2$                      ;;LOOP UNTIL ITS THERE
        007772  117746  171150                          MOVB    @$TKB,-(SP)             ;;GET CHARACTER
        007776  042716  177600                          BIC     #^C177,(SP)             ;;MAKE IT 7-BIT ASCII
        010002  022627  000021                          CMP     (SP)+,#21               ;;IS IT A CONTROL-Q?
        010006  001366                                  BNE     2$                      ;;IF NOT DISCARD IT
        010010  000750                                  BR      1$                      ;;YES, RESUME
        010012  026627  000004  000140          3$:     CMP     4(SP),#140              ;;IS IT UPPER CASE?
        010020  002407                                  BLT     4$                      ;;BRANCH IF YES
        010022  026627  000004  000175                  CMP     4(SP),#175              ;;IS IT A SPECIAL CHAR?
        010030  003003                                  BGT     4$                      ;;BRANCH IF YES
        010032  042766  000040  000004                  BIC     #40,4(SP)               ;;MAKE IT UPPER CASE
        010040  000002                          4$:     RTI                             ;;GO BACK TO USER
                                                ;;*************************************************************
                                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                                ;*CALL:
                                                ;*      RDLIN                           ;;INPUT A STRING FROM THE TTY
                                                ;*      RETURN HERE                     ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                                ;*                                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

        010042  010346                          $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
        010044  005046                                  CLR     -(SP)                   ;;CLEAR THE RUBOUT KEY
        010046  012703  010276                  1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
        010052  022703  010305                  2$:     CMP     #$TTYIN+7,R3            ;;BUFFER FULL?
        010056  101456                                  BLOS    4$                      ;;BR IF YES
        010060  104406                                  RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
        010062  112613                                  MOVB    (SP)+,(R3)              ;;GET CHARACTER
        010064  122713  000177                  10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
        010070  001022                                  BNE     5$                      ;;BR IF NO
        010072  005716                                  TST     (SP)                    ;;IS THIS THE FIRST RUBOUT?
        010074  001007                                  BNE     6$                      ;;BR IF NO
        010076  112737  000134  010274                  MOVB    #'\,9$                  ;;TYPE A BACK SLASH
        010104  104403  010274                          TYPE    ,9$
```

```
 1992  010110  012716  177777          MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
 1993  010114  005303                  5$:     DEC     R3              ;;BACKUP BY ONE
 1994  010116  020327  010276          CMP     R3,#$TTYIN      ;;STACK EMPTY?
 1995  010122  103434                  BLO     4$              ;;BR IF YES
 1996  010124  111337  010274          MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
 1997  010130  104400  010274          TYPE    ,9$             ;;GO TYPE
 2000  010134  000746                  BR      2$              ;;GO READ ANOTHER CHAR.
 2001  010136  005716                  6$:     TST     (SP)            ;;RUBOUT KEY SET?
 2002  010140  001406                  BEQ     7$              ;;BR IF NO
 2003  010142  112737  000134  010274  MOVB    #'\,9$          ;;TYPE A BACK SLASH
 2004  010150  104400  010274          TYPE    ,9$
 2005  010154  005016                  CLR     (SP)            ;;CLEAR THE RUBOUT KEY
 2006  010156  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
 2007  010162  001003                  BNE     8$              ;;BR IF NO
 2010  010164  104400  010305          TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
 2011  010170  000726                  BR      1$              ;;GO START OVER
 2012  010172  122713  000022          9$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
 2013  010176  001011                  BNE     3$              ;;BRANCH IF NO
 2014  010200  105013                  CLRB    (R3)            ;;CLEAR THE CHARACTER
 2015  010202  104400  001161          TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
 2016  010206  104400  010276          TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
 2017  010212  000717                  BR      2$              ;;GO PICKUP ANOTHER CHACTER
 2020  010214  104405  001160          4$:     TYPE    ,$QUES          ;;TYPE A '?'
 2021  010220  000712                  BR      1$              ;;CLEAR THE BUFFER AND LOOP
 2022  010222  111337  010274          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
 2023  010226  104400  010274          TYPE    ,9$
 2024  010232  122723  000015          CMPB    #15,(R3)+       ;;CHECK FOR RETURN
 2025  010236  001305                  BNE     2$              ;;LOOP IF NOT RETURN
 2026  010240  105063  177777          CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
 2027  010244  104400  001162          TYPE    ,$LF            ;;TYPE A LINE FEED
 2030  010250  005726                  TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
 2031  010252  012603                  MOV     (SP)+,R3        ;;RESTORE R3
 2032  010254  011646                  MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 2033  010256  016666  000004  000002  MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 2034  010264  012766  010276  000004  MOV     #$TTYIN,4(SP)
 2035  010272  000002                  RTI                     ;;RETURN
 2036  010274     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
 2037  010275     000                  .BYTE   0               ;;TERMINATOR
 2040  010276  000007                  $TTYIN: .BLKB   7               ;;RESERVE 7 BYTES FOR TTY INPUT
 2041  010305     136  006525  000012  $CNTLU: .ASCIZ  /^U<15><12>/    ;;CONTROL "U"
 2042  010312  043536  005015     000  $CNTLG: .ASCIZ  /^G<15><12>/    ;;CONTROL "G"
 2043  010317     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
 2044  010324  036440  000040
 2045  010330  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
 2046  010336  020075     000
 2047  010342                          .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:        $FILLS 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
                                        ;*

010342  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
010346  100002                          BPL     1$              ;;BR IF YES
010350  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
010352  000407                          BR      3$              ;;LEAVE
010354  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
010356  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
010362  112046                  2$:     MOVB    (R0)+,-(SP      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
010364  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
010366  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
010370  012600                  50$:    MOV     (SP)+,R0        ;;RESTORE R0
010372  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
010376  000002                          RTI                     ;;RETURN
010400  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF (HT)
010404  001430                          BEQ     8$
010406  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT (CRLF)
010412  001006                          BNE     5$
010414  005726                          TST     (SP)+           ;;POP  (CR)(LF) EQUIV
010416  104400                          TYPE                    ;;TYPE A CR AND LF
010420  001161                          $CRLF
010422  105037  010556                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
010426  000755                          BR      2$              ;;GET NEXT CHARACTER
010430  004737  010512          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
010434  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
010440  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
010442  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                        ;;AND THE NULL CHAR.
010446  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
010452  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
010454  004737  010512                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
010460  105237  010556                  INCB    $CHARCNT        ;;DO NOT COUNT AS A CHAR
010464  000770                          BR      7$              ;;LOOP

                                        ;HORIZONTAL TAB PROCESSOR

010466  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
010472  004737  010512          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
010476  132737  000007  010556          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
010504  001372                          BNE     9$              ;;TAB STOP
010506  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
010510  000724                          BR      2$              ;;GET NEXT CHARACTER
010512  105777  170432          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
010516  100375                          BPL     $TYPEC
010520  116677  000002  170424          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
010526  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
010534  001003                          BNE     1$              ;;BRANCH IF NO
010536  105037  010556                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
010542  000406                          BR      $TYPEC          ;;EXIT
010544  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
```

```
                        BEQ     $TYPEX          ;;BRANCH IF YES
                        INCB    (PC)+           ;;COUNT THE CHARACTER
$CHARCNT:: .WORD  0             ;;CHARACTER COUNT STORAGE
$TYPEX: RTS     PC

;;*****************************************************************
.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

010562  017646  000000          $TYPOS: MOV     @(SP)+,-(SP)    ;;PICKUP THE MODE
010566  116637  000001  011005          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
010574  112637  011007                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
010600  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
010604  000406                          BR      $TYPON
010606  112737  000001  011005  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
010614  112737  000006  011007          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
010622  112737  000005  011004  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
010630  010346                          MOV     R3,-(SP)        ;;SAVE R3
010632  010446                          MOV     R4,-(SP)        ;;SAVE R4
010634  010546                          MOV     R5,-(SP)        ;;SAVE R5
010636  113704  011007                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
010642  005404                          NEG     R4
010644  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
010650  110437  011006                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
010654  113704  011005                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
010660  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
010664  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
010666  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
010670  000404                  BR      3$              ;;GO DO MSB
010672  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
010674  006105                  ROL     R5
010676  006105                  ROL     R5
010700  010503                  MOV     R5,R3
010702  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
```

```
 2160  010704  105337  011006                DECB    $OMODE          ;;TYPE THIS DIGIT?
 2161  010710  100016                        BPL     7$              ;;BR IF NO
 2162  010712  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
 2163  010716  001002                        BNE     4$              ;;TEST FOR 0
 2164  010720  005704                        TST     R4              ;;SUPPRESS THIS 0?
 2165  010722  001403                        BEQ     5$              ;;BR IF YES
 2166  010724  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 2167  010726  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 2168  010732  052703  000040        5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 2169  010736  110337  011002                MOVB    R3,9$           ;;SAVE FOR TYPING
 2170  010742  104400  011002                TYPE    ,9$             ;;GO TYPE THIS DIGIT
 2171  010746  105337  011004        7$:     DECB    $OCNT           ;;COUNT BY 1
 2172  010752  003347                        BGT     2$              ;;BR IF MORE TO DO
 2173  010754  002402                        BLT     6$              ;;BR IF DONE
 2174  010756  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 2175  010760  000744                        BR      2$              ;;GO DO THE LAST DIGIT
 2176  010762  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
 2177  010764  012604                        MOV     (SP)+,R4        ;;RESTORE R4
 2178  010766  012603                        MOV     (SP)+,R3        ;;RESTORE R3
 2179  010770  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 2180  010776  012616                        MOV     (SP)+,(SP)
 2181  011000  000002                        RTI                     ;;RETURN
 2182  011002     000                9$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 2183  011003     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 2184  011004     000                $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 2185  011005     000                $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 2186  011006  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
 2187                                ;;**************************************************************
 2188                                .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
 2189
 2190                                ;;**************************************************************
 2191                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
 2192                                ;*CHANGE IT TO BINARY.
 2193                                ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
 2194                                ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
 2195                                ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
 2196                                ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
 2197                                ;*CALL:
 2198                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
 2199                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
 2200                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
 2201
 2202  011010  011646                $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
 2203  011012  016665  000004  000002        MOV     4(SP),2(SP)     ;;INPUT NUMBER
 2204  011020  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 2205  011022  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 2206  011024  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 2207  011026  104407                1$:     RDLIN                   ;;READ AN ASCII LINE
 2208  011030  012600                        MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
 2209  011032  010037  011136                MOV     R0,5$           ;;AND SAVE IT
 2210  011036  005001                        CLR     R1              ;;CLEAR DATA WORD
 2211  011040  005002                        CLR     R2
 2212  011042  112046                2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
 2213  011044  001420                        BEQ     3$              ;;IF ZERO GET OUT
 2214  011046  122716  000060                CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
 2215  011052  003026                        BGT     4$              ;;IS AN OCTAL DIGIT
```

```
002216  011054  122716  000067          CMPB    #'7,(SP)
002217  011060  002423                  BLT     4$
002220  011062  006301                  ASL     R1              ;;*2
002221  011064  006102                  ROL     R2
002222  011066  006301                  ASL     R1              ;;*4
002223  011070  006102                  ROL     R2
002224  011072  006301                  ASL     R1              ;;*8
002225  011074  006102                  ROL     R2
002226  011076  042716  177770          BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
002227  011102  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
002230  011104  000756                  BR      2$              ;;LOOP
002231  011106  005726          3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
002232  011110  010166  000012          MOV     R1,12(SP)       ;;SAVE THE RESULT
002233  011114  010237  011146          MOV     R2,$HIOCT
002234  011120  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
002235  011122  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
002236  011124  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
002237  011126  000002                  RTI                     ;;RETURN
002240  011130  005726          4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
002241  011132  105010                  CLRB    (R0)            ;;SET A TERMINATOR
002242  011134  104400                  TYPE                    ;;TYPE UP THRU THE BAD CHAR.
002243  011136  000000          5$:     .WORD   0
002244  011140  104400  001160          TYPE    ,$QUES          ;;"?" "CR" & "LF"
002245  011144  000730                  BR      1$              ;;TRY AGAIN
002246  011146  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.

002250  011150  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
002251  011152  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
002252  011156  005740                  TST     -(R0)           ;;BACKUP BY 2
002253  011160  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
002254  011162  006300                  ASL     R0              ;;POSITION FOR INDEXING
002255  011164  016000  011172          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
002256  011170  000200                  RTS     R0              ;;GO TO ROUTINE
002257
002258                                  .SBTTL  TRAP TABLE

002260                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
002261                                  ;*BY THE "TRAP" INSTRUCTION.

002262
002263                                  ;       ROUTINE
002264                                  ;       -------
002265  011172                  $TRPAD:
002266  011172  010342                  $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
002267  011174  010606                  $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
002270  011176  010662                  $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
002271  011200  010622                  $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
002272
002273  011202  007510                  $GTSWR  ;;CALL=GTSWR    TRAP+4,104404)  GET SOFT-SWR SETTING
```

```
                                                $CKSWR  ;;CALL=CKSWR    TRAP+5(104405)  TEST FOR CHANGE IN SOFT-SWR
                                                $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
                                                $RDLIN  ;;CALL=RDLIN    TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
                                                $RDOCT  ;;CALL=RDOCT    TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
                                        ;;**************************************************************
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;**************************************************************
                                        ;POWER DOWN ROUTINE
011214  012737  011360  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
011222  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
011230  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
011232  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
011234  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
011236  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
011240  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
011242  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
011244  017746  167670                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
011250  010637  011364                  MOV     SP,$SAVR6       ;;SAVE SP
011254  012737  011266  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
011262  000000                          HALT
011264  000776                          BR      .-2             ;;HANG UP

                                        ;;**************************************************************
                                        ;POWER UP ROUTINE
011266  012737  011360  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
011274  013706  011364                  MOV     $SAVR6,SP       ;;GET SP
011300  005037  011364                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
011304  005237  011364          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
011310  001375                          BNE     1$              ;;OF  WORD
011312  012677  167622                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
011316  012505                          MOV     (SP)+,R5        ;;POP STACK INTO R5
011320  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
011322  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
011324  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
011326  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
011330  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
011332  012737  011214  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
011340  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
011346  104400                          TYPE                    ;;REPORT THE POWER FAILURE
011350  013222                  $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
011352  012716                          MOV     (PC)+,(SP)      ;;RESTART AT START1
011354  001615                  $PWRAD: .WORD   START1          ;;RESTART ADDRESS
011356  000002                          RTI
011360  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
011362  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
011364  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
```

```
                                        .SBTTL  VTO1 INTENSITY POINTERS & DATA
                                ;TABLE OF POINTERS TO X AND Y INTENSITY PATTERNS FOR VTO1 DISPLAY
                                
2323  011366  011466           LEVTAB: L1
2324  011370  011467                   L2
2325  011372  011471                   L3
2326  011374  011474                   L4
2327  011376  011500                   L5
2328  011400  011505                   L6
2329  011402  011513                   L7
2330  011404  011522                   L8
2331  011406  011532                   L9
2332  011410  011543                   L10
2333  011412  011555                   L11
2334  011414  011570                   L12
2335  011416  011604                   L13
2336  011420  011621                   L14
2337  011422  011637                   L15
2338  011424  011656                   L16
2339  011426  011676                   L17
2340  011430  011717                   L18
2341  011432  011741                   L19
2342  011434  011764                   L20
2343  011436  012010                   L21
2344  011440  012035                   L22
2345  011442  012063                   L23
2346  011444  012112                   L24
2347  011446  012142                   L25
2348  011450  012173                   L26
2349  011452  012225                   L27
2350  011454  012260                   L28
2351  011456  012314                   L29
2352  011460  012351                   L30
2353  011462  012407                   L31
2354  011464  012446                   L32
2355
2356                            ;THIS IS THE CODED MATRIX TABLE FOR INTENSITY DISPLAY
2357                            ;THE ENTRIES ARE PACKED 2X,Y COORDINATES TO A WORD
2358                            ;THE EVEN BYTE CONTAINS THE FIRST PAIR,3 BITS PER
2359                            ;COORDINATE.
2360
2361  011466    033             L1:     .BYTE 33
2362  011467    016   062       L2:     .BYTE 16,62
2363  011471    026   032   064 L3:     .BYTE 26,32,64
2364  011474    021   015   056 L4:     .BYTE 21,15,56,62
2365  011477    062
2366  011500    005   020   043 L5:     .BYTE 05,20,43,57,71
2367  011503    057   071
2368  011505    024   000   032 L6:     .BYTE 24,00,32,47,65,51
2369  011510    047   065   051
2370  011513    002   015   020 L7:     .BYTE 02,15,20,43,46,61,65
2371  011516    043   046   061
2372  011521    065
2373  011522    005   012   024 L8:     .BYTE 05,12,24,37,41,56,70,63
2374  011526    037   041   056
2375  011530    070   063
```

```
2376  011532  000  013  016  L9:    .BYTE 00,13,16,31,35,43,57,61,74
2377  011535  031  035  043
2378  011540  057  061  074
2379  011543  000  013  016  L10:   .BYTE 00,13,16,31,43,55,57,61,74,35
2380  011546  031  043  055
2381  011551  057  061  074
2382  011554  035
2383  011555  012  015  027  L11:   .BYTE 12,15,27,33,41,45,62,57,74,00,76
2384  011560  033  041  045
2385  011563  062  057  074
2386  011566  000  076
2387  011570  000  003  016  L12:   .BYTE 00,03,16,21,24,37,42,45,57,61,64,76
2388  011573  021  024  037
2389  011576  042  045  057
2390  011601  061  064  076
2391  011604  007  013  015  L13:   .BYTE 07,12,15,20,34,36,42,50,54,56,71,73,75
2392  011607  020  034  036
2393  011612  042  050  054
2394  011615  056  071  073
2395  011620  075
2396  011621  001  005  007  L14:   .BYTE 01,05,07,13,20,26,32,34,46,40,52,60,65,73
2397  011624  013  020  026
2398  011627  032  034  046
2399  011632  040  052  060
2400  011635  065  073
2401  011637  003  006  027  L15:   .BYTE 03,06,27,11,25,30,33,45,47,50,54,52,71,74,66
2402  011642  011  025  030
2403  011645  033  045  047
2404  011650  050  054  052
2405  011653  071  074  066
2406  011656  006  010  023  L16:   .BYTE 06,10,23,25,27,30,32,44,46,50,53,64,71,77,02,04
2407  011661  025  027  030
2408  011664  032  044  046
2409  011667  050  053  064
2410  011672  071  077  002
2411  011675  004
2412  011676  002  004  006  L17:   .BYTE 02,04,06,10,22,25,27,30,33
2413  011701  010  022  025
2414  011704  027  030  033
2415  011707  045  047  050          .BYTE 45,47,50,52,65,71,73,77
2416  011712  052  065  071
2417  011715  073  077
2418  011717  002  004  006  L18:   .BYTE 02,14,06,10,22,25,27,30,33
2419  011722  010  022  025
2420  011725  027  030  033
2421  011730  045  047  050          .BYTE 45,47,50,52,64,66,71,73,77
2422  011733  052  064  066
2423  011736  071  073  077
2424  011741  002  004  006  L19:   .BYTE 02,04,06,10,22,25,27,30,34,41,43,46,50,57
2425  011744  010  022  025
2426  011747  027  030  034
2427  011752  041  043  046
2428  011756  050  057
2429  011757  063  065  071          .BYTE 63,65,71,74,77
2430  011762  074  077
2431  011764  001  003  005  L20:   .BYTE 01,03,05,07,12,30,23,64,25,42,27,34,36,51
```

```
011767   007   012   030
011772   023   064   025
011775   042   027   034
012000   036   051
012002   055   057   070         .BYTE 55,57,70,53,66,72
012005   053   066   072
012010   001   003   005   L21:  .BYTE 01,03,05,07,10,40,16,21,23,25,42,27,34,45
012013   007   010   040
012016   016   021   023
012021   025   042   027
012024   034   045
012026   051   055   057         .BYTE 51,55,57,60,64,66,72
012031   060   064   066
012034   072
012035   001   003   005   L22:  .BYTE 01,03,05,07,10,40,16,21,23,25,42,27,34,46
012040   007   010   040
012043   016   021   023
012046   025   042   027
012051   034   046
012053   051   053   055         .BYTE 51,53,55,57,60,64,66,72
012056   057   060   064
012061   066   072
012063   000   002   005   L23:  .BYTE 00,02,05,11,13,16,22,34,27,31,25,40,42,47
012066   011   013   016
012071   022   034   027
012074   031   025   040
012077   042   047
012101   051   054   056         .BYTE 51,54,56,60,63,65,71,74,76
012104   060   063   065
012107   071   074   076
012112   000   003   007   L24:  .BYTE 00,03,07,12,14,16,21,23,25,27,32,34,30,43
012115   012   014   016
012120   021   023   025
012123   027   032   034
012126   030   043
012130   046   050   052         .BYTE 46,50,52,55,63,65,67,71,74,76
012133   055   063   065
012136   067   071   074
012141   076
012142   003   005   007   L25:  .BYTE 03,05,07,10,12,14,16,21,23,27,32,34,36,41
012145   010   012   014
012150   016   021   023
012153   027   032   034
012156   036   041
012160   045   047   050         .BYTE 45,47,50,52,54,63,65,67,70,72,76
012163   052   054   063
012166   065   067   070
012171   072   076
012173   003   005   007   L26:  .BYTE 03,05,07,10,12,16,21,23,25,30,32,34,36,41
012176   010   012   016
012201   021   023   025
012204   030   032   034
012207   036   041
012211   045   047   052         .BYTE 45,47,52,54,56,61,63,67,70,72,74,76
012214   054   056   061
012217   063   067   070
```

```
2488  012222  072  074  076
2489  012225  001  003  007  L27:   .BYTE 01,03,07,12,14,16,21,23,25,27,30,32,34,36
2490  012230  012  014  016
2491  012233  021  023  025
2492  012236  027  030  032
2493  012241  034  036
2494  012243  041  045  047        .BYTE 41,45,47,50,52,54,61,63,65,67,70,74,76
2495  012246  050  052  054
2496  012251  061  063  065
2497  012254  067  070  074
2498  012257  076
2499  012260  003  005  007  L28:   .BYTE 03,05,07,10,12,14,16,21,23,27,30,32,34,36
2500  012263  010  012  014
2501  012266  016  021  023
2502  012271  027  030  032
2503  012274  034  036
2504  012276  043  045  047        .BYTE 43,45,47,50,52,54,56,61,63,65,67,70,72,76
2505  012301  050  052  054
2506  012304  056  061  063
2507  012307  065  067  070
2508  012312  072  076
2509  012314  001  003  005  L29:   .BYTE 01,03,05,07,10,12,16,14,21,23,27,30,32,34,36,43
2510  012317  007  010  012
2511  012322  016  014  021
2512  012325  023  027  030
2513  012330  032  034  036
2514  012333  043
2515  012334  045  047  050        .BYTE 45,47,50,52,54,56,61,63,67,70,72,74,76
2516  012337  052  054  056
2517  012342  061  063  067
2518  012345  070  072  074
2519  012350  076
2520  012351  001  003  005  L30:   .BYTE 01,03,05,07,10,12,14,16,21,23,25,27,30,32,36,41
2521  012354  007  010  012
2522  012357  014  016  021
2523  012362  023  025  027
2524  012365  030  032  036
2525  012370  041
2526  012371  043  045  047        .BYTE 43,45,47,50,54,56,61,63,65,67,70,72,74,76
2527  012374  050  054  056
2528  012377  061  063  065
2529  012402  067  070  072
2530  012405  074  076
2531  012407  001  003  005  L31:   .BYTE 01,03,05,07,10,12,14,16,21,23,25,27,30,32,36,41
2532  012412  007  010  012
2533  012415  014  016  021
2534  012420  023  025  027
2535  012423  030  032  036
2536  012426  041
2537  012427  043  045  047        .BYTE 43,45,47,50,52,54,56,61,63,65,67,70,72,74,76
2538  012432  050  052  054
2539  012435  056  061  063
2540  012440  065  067  070
2541  012443  072  074  076
2542  012446  001  003  005  L32:   .BYTE 01,03,05,07,10,12,14,16,21,23,25,27,30,32,34,36
2543  012451  007  010  012
```

```
2544  012454  014  016  021
2545  012457  023  025  027
2546  012462  030  032  034
2547  012465  036
2548  012466  041  043  045          .BYTE 41,43,45,47,50,52,54,56,61,63,65,67,70,72,74,76
2549  012471  047  050  052
2550  012474  054  056  061
2551  012477  063  065  067
2552  012502  070  072  074
2553  012505  076
2554                                 .EVEN
```

```
                                                        .SBTTL  DISPLAY MESSAGES
                                                ;THIS IS THE MESSAGE FOR THE BOTTOM OF THE SCOPE
                    000000                              TERM=0
012506  012536                                MESS1:  WD001
012510  000000                                        TERM
012512  012561                                MESS2:  WD002
012514  000000                                        0
012516  012604                                MESS3:  WD003
012520  000000                                        0
012522  012617                                MESS4:  WD004
012524  000000                                        0
012526  012636                                MESS5:  WD005
012530  000000                                        0
012532  012645                                MESS6:  WD006
012534  000000                                        0
012536  005015  047514  042527                WD001:  .ASCII  <15><12>/LOWER THRESHOLD /<END>
012544  020122  044124  042522
012552  044123  046117  020104
012560     200
012561     015  052412  050120                WD002:  .ASCII  <15><12>/UPPER THRESHOLD /<END>
012566  051105  052040  051110
012574  051505  047510  042114
012602  100040
012604  005015  020132  047503                WD003:  .ASCII  <15><12>/Z COUNT=/<END>
012612  047125  036524     200
012617     040  040515  051124                WD004:  .ASCII  / MATRIX COUNT=/<END>
012624  054111  041440  052517
012632  052116  100075
012636  020040  047532  046517                WD005:  .ASCII  /  ZOOM/<END>
012644     200
012645     040  026502  040507                WD006:  .ASCII  / B-GAMMA/<END>
012652  046515  100101
                                                        .EVEN
                    000200                      END=200         ;MSG TERMINATOR FOR DISPLAY CHARS
```

```
                                        .SBTTL  VT01 CHARACTER TABLE
                                ;THIS IS THE TABLE OF CODED LETTERS AVAILABLE FOR THE
                                ;VT01 SCOPE.

                                TABLE:  77061           ;A
                                        35061           ;B
                                        45111           ;C
                                        37116           ;D
                                        02056           ;E
                                        34041           ;F
                                        45111           ;G
                                        37122           ;H
                                        02041           ;I
                                        76047           ;J
                                        02041           ;K
                                        76047           ;L
                                        73056           ;M
                                        35041           ;N
                                        43061           ;O
                                        43077           ;P
                                        10216           ;Q
                                        34204           ;R
                                        60446           ;S
                                        14411           ;T
                                        12461           ;U
                                        42447           ;V
                                        02077           ;W
                                        02041           ;X
                                        53061           ;Y
                                        43565           ;Z
                                        63461           ;[
                                        43165           ;\
                                        43056           ;]
                                        35061           ;^
                                        36041           ;_
                                        37061           ;`
                                        60466           ;a
                                        35061           ;b
                                        12461           ;c
                                        37057           ;d
                                        41017           ;e
                                        70114           ;f
                                        10204           ;g
                                        76204           ;h
                                        43056           ;i
                                        43061           ;j
                                        42504           ;k
                                        43061           ;l
                                        53561           ;m
                                        43065           ;n
                                        25061           ;o
                                        42504           ;p
                                        10204           ;q
                                        43052           ;r
                                        02077           ;s
```

C05


```
2645  013026  002057        02057   :[
2646  013030  036041        36041
2647  013032  010400        10400   :\
2648  013034  020202        20202
2649  013036  020417        20417   :]
2650  013040  036410        36410
2651  013042  001740        01740   :=
2652  013044  000037        00037
2653  013046  000200        00200   ::
2654  013050  000004        00004
2655  013052  000000        00000   : SPACE
2656  013054  000000        00000
2657  013056  010004        10004   :!
2658  013060  010204        10204
2659  013062  000000        00000   :"
2660  013064  024512        24512
2661  013066  076512        76512   :#
2662  013070  025752        25752
2663  013072  035217        35217   :$
2664  013074  011705        11705
2665  013076  005470        05470   :%
2666  013100  046544        46544
2667  013102  010100        10100   :
2668  013104  004210        04210
2669  013106  000000        00000   :'
2670  013110  004101        04101
2671  013112  004104        04104   :
2672  013114  010102        10102
2673  013116  020404        20404   :
2674  013120  010410        10410
2675  013122  010004        10004   :*
2676  013124  035050        35050
2677  013126  010200        10200   :+
2678  013130  010237        10237
2679  013132  004101        04101   :,
2680  013134  000000        00000
2681  013136  000000        00000   :-
2682  013140  000017        00017
2683  013142  000004        00004   :.
2684  013144  000000        00000
2685  013146  010101        10101   :
2686  013150  001010        01010
2687  013152  047056        47056   :0
2688  013154  075465        75465
2689  013156  010216        10216   :1
2690  013160  010304        10304
2691  013162  004077        04077   :2
2692  013164  035054        35054
2693  013166  041056        41056   :3
2694  013170  035054        35054
2695  013172  076410        76410   :4
2696  013174  020612        20612
2697  013176  041017        41017   :5
2698  013200  076057        76057
2699  013202  037056        37056   :6
2700  013204  034041        34041
```

```
013206  010102                                  10102           ;7
013210  007010                                  07010
013212  043056                                  43056           ;8
013214  035056                                  35056
013216  041016                                  41016           ;9
013220  035077                                  35077

                                        .SBTTL  ASCII MESSAGES

013222  005015  051012  051505  PWRMSG: .ASCIZ  <15><12><12>/RESTARTED AFTER POWER FAILURE/
013230  040524  052122  042105
013236  040440  052106  051105
013244  050040  053517  051105
013252  043040  044501  052514
013260  042522      000
013263      015  005012  042115  MSG1:   .ASCIZ  <15><12><12>/MD-11-DZNCB-C  GAMMA 11 EXERCISER/
013270  030455  026461  055104
013276  041516  026502  020103
013304  020040  040507  046515
013312  020101  030461  042440
013320  042530  041522  051511
013326  051105      000
013331      015  042412  052116  MSG2:   .ASCIZ  <15><12>/ENTER THRESHOLD VALUE IN OCTAL - THEN RETURN =/
013336  051105  052040  051110
013344  051505  047510  042114
013352  053040  046101  042525
013360  044440  020116  041517
013366  040524  020114  020055
013374  044124  047105  051040
013402  052105  051125  020116
013410  000075
013412  005015  052502  020123  MSG3:   .ASCIZ  <15><12>/BUS TIMEOUT ER - VSV01 DISPLAY/
013420  044524  042515  052517
013426  020124  051105  026440
013434  053040  053123  030460
013442  042040  051511  046120
013450  054501      000
013453      015  041012  051525  MSG4:   .ASCIZ  <15><12>/BUS TIMEOUT ER - VTO1 DISPLAY/
013460  052040  046511  047505
013466  052125  042440  020122
013474  020055  052126  030460
013502  042040  051511  046120
013510  054501      000
013513      015  041012  051525  MSG5:   .ASCIZ  <15><12>/BUS TIMEOUT ER - NC11/
013520  052040  046511  047505
013526  052125  042440  020122
013534  020055  041516  030461
013542      000
013543      015  042412  052116  MSG6:   .ASCIZ  <15><12>/ENTER KEYBOARD COMMAND(S)
013550  051105  045440  054505
013556  047502  051101  020104
013564  047503  046515  047101
013572  024104  024523      000
013577      015  041012  051525  MSG7:   .ASCIZ  <15><12>/BUS TIMEOUT ER - AR11
013604  052040  046511  047505
013612  052125  042440  020122
```

EOS


```
2757 013620 020055 051101 030461
2758 013626    000
2759         000001                .END
```

F05


```
ADUMMY   007105          1689    1694    1752*
ARADR    001174           617*   1386
ARBUF    001244           652*   1507    1512
ARCHAN   001350           689*   1289*   1292*   1295*   1508
ARCONV   006024          1290    1293    1296    1507*
ARCSR    001242           651*   1369    1387    1393*   1508*   1509*   1510
ASCII2   006736          1719*
ASCLP    006770          1726*   1735
ASCRES   007072          1724    1751*
ASCZSP   006732          1692    1718*
AWAIT    006700          1118    1122    1691*
BASX     001326           680*   1113*   1589*   1659    1673    1676*
BASY     001330           681*   1114*   1593*   1659
BELLEN   001354           691*   1253*   1254
BIT0   = 000001           501*
BIT00  = 000001           491*    501
BIT01  = 000002           490*    500
BIT02  = 000004           499*    499
BIT03  = 000010           498*    498
BIT04  = 000020           497*    497
BIT05  = 000040           496*    496
BIT06  = 000100           495*    495
BIT07  = 000200           494*    494
BIT08  = 000400           493*    493
BIT09  = 001000           492*    492
BIT1   = 000002           500*   1405
BIT10  = 002000           481*
BIT11  = 004000           480*
BIT12  = 010000           479*
BIT13  = 020000           478*
BIT14  = 040000           477*
BIT15  = 100000           476*
BIT2   = 000004           499*
BIT3   = 000010           498*
BIT4   = 000020           497*
BIT5   = 000040           496*    951
BIT6   = 000100           495*
BIT7   = 000200           494*
BIT8   = 000400           493*    979     910
BIT9   = 001000           492*
BLANK    006674          1689*   1696
BCOBCO   002064           790     793     796     830*
BOX      002626           927*
BOX1     003012           929     960*
BPTVEC=  000014           509*
BRAN     006462          1597    1633*
BRANL    006470          1634*   1639
CHANGE   002616           836     838     841     859     874     894     903     919*
CHARCT   001324           679*    986*    995*
CHKON    003436          1022    1050    1059*
CH72     006416          1601    1610*
CH74     006430          1599    1612*
CH75     006422          1600    1611*
CH76     006410          1602    1609*
CH77     006402          1603    1608*
CLEAR  = [illegible]     2273*
```

```
CKVALU  003270          1014    1017#
CLALL = 000100           521#
CLZ   = 000020           520#   1421
CSL     006672          1556*   1671*   1683#
CSLK    006670          1555    1682#
CSMSAV  001316           676#    884    1152    1412*
CPERL   001334           683#   1003*   1025
CR    = 000015           416#   2099    2109
CRLF  = 000200           417#   2070    2109
CSCHAR  006106          1526#   1527    1685
DCONT   005646           935     942    1173    1180    1454#
DDISP = 177570           423#    563     712
DISBG   004462          1192    1216    1241#
DISBG1  004730          1242    1298#
DISDAT  002622           923#
DISDON  003500          1002    1067#
DISLOP  003236          1011#   1061    1066    1091    1103
DISPLA  001142           563#    712*    720*
DISPRE  000174           531#    720
DISPY   005634           931     939     949     952    1089    1163    1176    1184    1187    1452#
DISPY1  005676           962     966     971     976    1139    1204    1209    1213    1221    1225    1231    1235    1282
                        1319    1464#   1465    1565    1662
DISPY2  005726          1284    1320    1475#   1476
DIVCNT  007070          1723*   1734*   1753#
DIVIDE  007120          1728    1771#
DIVJUS  007126          1774#   1776
DIVLOO  007146          1780#   1789
DMATRX  003212          1008#
DSWR  = 177570           422#    562     711
DTYPE   001254           655#    927    1021    1027    1045    1107    1163    1265    1269    1303    1305    1351*   1374*
                        1431    1583
DUMMY1  006724          1117*   1121*   1125*   1138*   1693    1715#
DUMMY2  006726          1126*   1139*   1691*   1716#
DVLOOP  003342          1031#   1041
DVTO1   006304          1584    1587#
EMTVEC= 000030           511#
END   = 000200          2570    2574    2578    2580    2583    2595    2589#
ERASE   005536           942     960     919    1162    1431#
ERRVEC= 000004           504#    709     710*    721*    737*    743*   1343*   1354*   1371*   1376*   1382*   1395*
FREERN  001342           686#    782*    844*   1149
GAIN    001320           677#    747*    886*    914*    915    1141    1405
GET     006212          1056    1556#
GNS   = ******  U        530    2266    2267    2268    2269    2271    2273    2274    2275    2276
GOON    003376          1043    1045#
GSG     003166           990     995#
GTSWR = 104404          2271#
HORIZ   001274           667#    980*   1060*   1062*
HT    = 000011           414#   2068    2109
INCXY   001332           682#   1111*   1666    1670    1675
INTENS  001256           660#   1352*   1373*
INTOUT  001260           661#   1346
IOTVEC= 000020           509#
JBCX    004160          1167#
JBCX1   004306          1164    1195#
JBCX2   004400          1196    1219#
JTYPE   001346           688#    867*   1135    1241    1394*
```

```
KBDBUF  001304   671*  1595*  1638
KBINT   006122   751   1534#
KBUFF   001306   672#   784*   796   798*   951*   852   1250   1252*   1534*   1535*   1536
LARGEL  003150   989#   996
LARGES  003132   959    996#
LDIMGE  005756   993   1489#
LEVTAB  011366  1048   2383#
LF    = 000012   415#  2103   2109
LISEN   001712   792#   856   1538
LISN    001732   786#   787    933   840   943   966   890   995   959   991   903   911   916
                1153
L1      011466  2323   2361#
L2      011543  2332   2379#
L3      011565  2333   2393#
L4      011570  2334   2387#
L5      011604  2335   2391#
L6      011621  2336   2396#
L7      011637  2337   2401#
L8      011656  2338   2406#
L9      011676  2339   2412#
L10     011717  2340   2418#
L11     011741  2341   2424#
L12     011467  2324   2362#
L13     011764  2342   2431#
L14     012010  2343   2438#
L15     012035  2344   2446#
L16     012063  2345   2454#
L17     012112  2346   2462#
L18     012142  2347   2471#
L19     012173  2348   2480#
L20     012225  2349   2489#
L21     012260  2350   2499#
L22     012314  2351   2509#
L23     011471  2325   2363#
L24     012351  2352   2520#
L25     012407  2353   2531#
L26     012446  2354   2542#
L27     011474  2326   2364#
L28     011500  2327   2366#
L29     011505  2328   2368#
L30     011513  2329   2370#
L31     011522  2330   2373#
L32     011532  2331   2376#
MAINCHER 006530  1633*  1635*  1636   1638   1644#
MAINPADR 001266   664#   955*  1071   1094*  1095   1102*
MAINPLD  003504  1034   1046  1071#
MATRIX=  020000   522#   685   746    748    935    937   1145   1422   1424
MESS1   012506  1116   2558#
MESS2   012512  1120   2560#
MESS3   012516  1124   2562#
MESS4   012522  1129   2564#
MESS5   012526  1144   2566#
MESS6   012532  1148   2568#
MLOOP   003416  1054#  1056
MOVAL   007062  1747#  1748
MOVALL  007052  1738   1743   1745#
```

DZNCB.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
RDLIN = 104407          2207    2275#
RDOCT = 104410           871     897    2276#
RESVEC= 000010           505#
RLJL    007174          1791#   1793
RMJUST  007206          1779*   1792*   1795#
ROUTA   002104           901     835#
ROUTB   002116           802     837#
ROUTC   002130           803     839#
ROUTD   002140           804     841#
ROUTE   002144           805     842#
ROUTF   002154           806     844#   1151
ROUTG   002250           807     860#
ROUTJ   002260           810     862#
ROUTK   002302           811     867#
ROUTL   002312           812     869#
ROUTM   002342           813     875#
ROUTN   002402           814     893#
ROUTR   002420           818     886#
ROUTS   002436           819     889#
ROUTT   002454           820     892#
ROUTU   002470           821     895#
ROUTV   002520           822     901#
ROUTW   002540           823     906#
ROUTZ   002576           826     914#
ROW     006666          1655*   1667*   1669*   1681#
ROWCNT  001336           684#   1010*   1013*   1015*
RCWK    006664          1655    1669    1680#
RTABLE  002000           795     797     801#
R0      =%000000         426#    731*    733     734*    786*    789     791*    792     794*    795     797     847*    849*
                         878*    879*    909*    910*    929*    932*    933     937*    939*    940     944*    945*    947
                         951*    960*    963*    965*    972*    973     975*    987*    989     993*   1000*   1003    1054*
                        1071*   1095*   1096    1099*   1100    1167*   1170*   1171    1175*   1177*   1179    1192*   1196*
                        1189*   1189    1197*   1200*   1201    1203*   1210*   1212*   1219*   1222*   1223    1225*   1232*
                        1234*   1273*   1275*   1276*   1277*   1309*   1311*   1312*   1313*   1329*   1331    1332*   1335*
                        1338*   1339    1340*   1346*   1347    1348*   1349    1365*   1367    1368*   1386*   1388    1389*
                        1422*   1423*   1424    1453    1466*   1468    1489*   1496*   1559*   1658*   1666*   1673*   1675*
                        1676    1733    1745*   1747*   1772*   1781*   1800    1801*   1803    1910    1913*   1814*   1915
                        1816*   1819*   1822*   1826*   1827*   1828*   1829*   1830*   2060    2061*   2062    2065*   2204
                        2208*   2209    2212    2232*   2235*   2250    2251*   2252    2253*   2254*   2255*   2256*   2294
                        2309*
R1      =%000001         427#    732*    733*    735     848*    957*    930*    946*    950*    961*    967*    969     970*
                         977*    978     988*    989     991     993     994    1025*   1032    1034    1037    1039*   1042
                        1055*   1088*   1133*   1134    1168*   1183*   1185*   1198*   1205*   1206    1208*   1214*   1220*
                        1227*   1228    1230*   1236*   1274*   1278*   1279*   1280*   1310*   1314*   1315*   1316*   1330*
                        1331*   1333    1339*   1341    1366*   1367*   1369    1387*   1388*   1390    1457    1467*   1469
                        1490*   1493*   1562*   1564*   1659*   1670*   1719*   1721    1722    1732    1773*   1780*   1795*
                        2205    2210*   2218*   2220*   2222*   2225*   2228    2231*   2285    2308*
R2      =%000002         428#    1011*   1017    1019*   1032    1037*   1042*   1131*   1135*   1139    1491*   1494*   1499*
                        1556*   1557*   1558*   1559    1560*   1561*   1562    1563*   1564    1657*   1660*   1665*   1722*
                        1730*   1731    1733*   1782    1784*   1787*   1791*   1902*   1803*   1804    1806*   1807    1809*
                        1811*   1821    1823*   1824    2206    2211*   2219*   2221*   2223*   2229    2230*   2296    2307*
R3      =%000003         429#    1026*   1029*   1040*   1132*   1134*   1138    1291    1294    1297    1492*   1493    1498*
                        1512*   1513*   1514    1518*   1520*   1528    1581*   1587    1591    1595    1612*   1613*   1614*
                        1615*   1616*   1657    1665    1721*   1732*   1786*   1979    1981*   1982    1985*   1995    1993*
                        1994    1996    2004    2008    2010*   2016    2018    2020*   2023*   2143    2152*   2159*   2159*
                        2162*   2167*   2168*   2169    2179*   2287    2306*
```

DZNCB.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

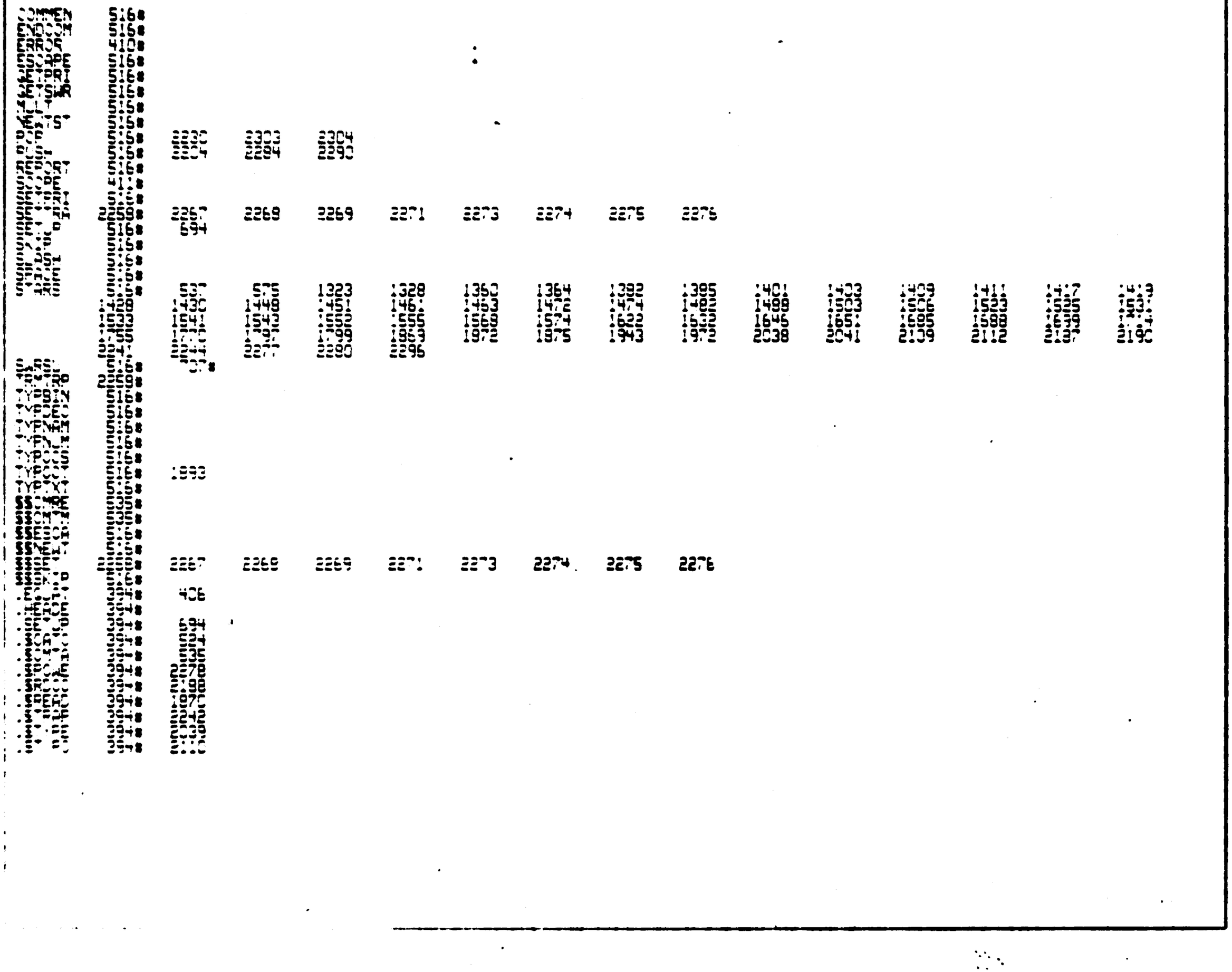
```
R4      =:000004   430#  1023*  1030*  1031*  1036*  1044*  1047*  1049   1049*  1057*  1072   1074*  1075*
                  1076*  1077*  1078*  1079*  1085   1580*  1581   1726*  1775*  1777*  1782   1784   2144
                  2146*  2147*  2148*  2149   2150*  2164   2166*  2174*  2177*  2298   2305*
R5      =:000005   431#  1048*  1556   1560   1576*  1578   1580   1724*  1725*  1727   1729*  1731*  1740
                  1742   1744*  1747   1771*  1774*  1778   1779*  1788*  1820*  1921*  1822   2145   2151*
                  2153*  2155*  2156*  2157*  2158   2176*  2289   2304*
R6      =:000006   432#   434    697*   698*   699
R7      =:000007   433#   435
SCLOEL   003320   1020   1025#
SELCTA   005112    724    876    907   1329#
SELCTB   005262    726    901   1365#
SELCTC   005356    962   1386#
SHFTCT   006662   1654*  1663*  1679#
SHFTK    006660   1654   1678#
SKZE     007036   1740#  1741
SP      =:000006   434#   701*   709*   717*   721    740    783*   872    898    1356   1378   1576   1577*
                  1633   1634*  1640*  1719   1720*  1727*  1729   1745   1746*  1800*  1810*  1814   1916
                  1884*  1885*  1886   1893*  1896*  1897*  1901*  1902*  1906   1909*  1913   1915   1917
                  1918*  1925   1927   1929*  1930   1932*  1933*  1934*  1935*  1936*  1951*  1952*  1955*
                  1956*  1957   1961*  1962*  1963   1966   1968   1970*  1979*  1980*  1985   1998   1992*
                  1999   2003*  2022   2023   2024*  2025*  2026*  2060*  2061   2062*  2064   2065   2066*
                  2068   2070   2072   2078   2080*  2082*  2090*  2094   2098   2099   2103   2135*  2136
                  2137   2138*  2143*  2144*  2145*  2151   2176   2177   2178   2179*  2180*  2202*  2203*
                  2204*  2205*  2206*  2208   2212*  2214   2216   2224*  2225   2227   2228*  2230   2231
                  2232   2234   2250*  2251   2294*  2285*  2296*  2287*  2298*  2289*  2290*  229?   2299*
                  2303   2304   2305   2306   2307   2308   2309   2314*
STACK = 001100    409#   701    783
START    001356    534    694#
START1   001616    739    743#   861   2315
STKLMT= 177774    420#
SWR      001140    562#   699    711*   713    719*  1880   1917*  2290   2303*
SWREG    000176    532#   719   1880   1893
SW0   = 000001    473#
SW00  = 000001    463#   473
SW01  = 000002    462#   472
SW02  = 000004    461#   471
SW03  = 000010    460#   470
SW04  = 000020    459#   469
SW05  = 000040    458#   468
SW06  = 000100    457#   467
SW07  = 000200    456#   466
SW08  = 000400    455#   465
SW09  = 001000    454#   464
SW1   = 000002    472#
SW10  = 002000    453#
SW11  = 004000    452#
SW12  = 010000    451#
SW13  = 020000    450#
SW14  = 040000    449#
SW15  = 100000    448#
SW2   = 000004    471#
SW3   = 000010    470#
SW4   = 000020    469#
SW5   = 000040    468#
SW6   = 000100    467#
SW7   = 000200    466#
```

```
SW8    = 000400            465*
SW9    = 001000            464*
TABLE    012656           1616    2593*
TABLEX   001340            685*    748*    835*    837*    987     988     1008    1133    1145    1490
TBITVE=  000014            506*
TEMCHR   005460           1596*   1608*   1609*   1610*   1611*   1612    1620*
TEMPO    001246            656*   1130*   1136*   1259*   1260*   1268*   1271*   1272*   1273    1274    1291*   1298*   1299*
                          1308*   1309    1310    1390    1434*   1436*   1477*   1478    1801    1915*
TEMP1    001250            657*   1294*   1299    1435*   1438*
TEMP2    001252            658*   1297*   1301
TERM   = 000000           2557*   2559
THHI     001314            675*    750*    898*    991     1017    1121
THLO     001312            674*    749*    872*   1000     1019    1117
TIME     001344            687*    848
TIXBNO   006576           1561    1663*
TIXDIS   006532           1617    1654*
TIXDON   006646           1672    1675*
TIXL     006566           1660*   1668    1674
TIXWOK   006606           1664    1666*
TJOY     004146            864     868    1162*
TKVEC  = 000060            513*    751*    752*
TOTSIZ   001322            678*    986    1130
TPVEC  = 000064            514*
TRAPVE=  000034            512*    703*    704*
TRTVEC=  000014            507*
TTYOUT   001310            673*    830*   1256*   1544*   1546*   1549
TYPCR    006154            832    1544*
TYPE   = 104400            723     729     729     741     754     865     870     881     896     904     912    1891    1892
                          1895    1908    1919    1938    1991    1997    2002    2006    2011    2012    2014    2017    2021
                          2073    2170    2236    2238    2266*   2312
TYPO     006174            831    1257    1545    1547*   1548
TYPCC  = 104401           1894    2267*
TYPCN  = 104403           2269*
TYPCS  = 104402           2268*
VERT     001276            668*    981*   1065*
VTADR    001172            613*   1365
VTCSR    001234            645*   1112*   1281*   1317*   1341    1366    1372*   1433*   1464
VTEXCP   006340           1592    1595*
VTEXCT   006374           1598    1605*
VTMADR   001170            609*   1335
VTNOSP   006434           1604    1613*
VTPL     006256           1578*   1582
VTVADR   001166            605*   1329
VTVCHP   001216            632*   1478*
VTVCRG   001214            631*    735    1330    1344    1445*   1475    1479*   1526    1528*
VTVCSR   001222            637*    878     909     954*   1191*   1333    1345    1441*   1442    1444*   1452*   1454*   1455
                          1458*
VTVINT   001232            641*   1347*
VTVMAP   001224            638*   1453*
VTVPOS   001220            633*   1109*
VTVPX    001226            639*   1457*
VTVPX1   001230            640*
VTVSAV   001262            662*    954    1191
VTWDON   006456           1579    1619*
VTWL     006264           1581*   1586    1590    1594    1618
VTWRIT   006246           1115    1119    1123    1128    1143    1147    1576*   1695
```

```
VTXDAC  001236            646*   1468*
VTYDAC  001240            647*   1469*
WD001   012536           2558    2570*
WD002   012561           2560    2574*
WD003   012604           2562    2578*
WD004   012617           2564    2580*
WD005   012636           2566    2583*
WD006   012645           2568    2585*
XREF    001300            669*    982*   1054    1059*   1063*
XYAVE   007210           1261    1300    1800*
XYBUF   007336           1806    1808    1811    1833*   1867
XYBUFE= 007436           1804    1824    1866*
XYBUFP  007436           1802    1807*   1867*
YREF    001302            670*    983*   1055    1064*
ZESUP   006730           1717*   1718*   1737    1739*
$AUTOB  001134            559*   1889    2037
$BDADR  001122            554*
$BDDAT  001126            556*
$CHARC  010556           2075*   2085*   2092    2101*   2106*
$CKSWR  007440           1880*   2273
$CMTAG  001100            542*    696     697
$CM3  = 000000            572*
$CNTLC  010312           1891    2032*
$CNTLU  010305           1908    2006    2031*
$CRLF   001161            573*   1919    2011    2031    2074    2103    2241
$ERFLG  001103            545*
$ERMAX  001115            551*
$ERRPC  001116            552*
$ERRTB  001164            590*
$ERTTL  001112            549*
$FILLC  001156            570*   2078    2109
$FILLS  001155            569*   2109
$GDADR  001120            553*
$GDDAT  001124            555*
$GTSWR  007510           1892*   2271
$HD   = 000003             404     405
$HIOCT  011146           2229*   2240*
$ICNT   001104            546*
$ILLUP  011360           2282    2298    2317*
$INTAG  001135            560*   1920    2037
$ITEMB  001114            550*
$LF     001162            574*   2021    2031    2109    2241
$LPADR  001106            547*
$LPERR  001110            548*
$MAIL = ****** U          723    2062
$MNEW   010330           1895    2035*
$MSWR   010317           1892    2033*
$NULL   001154            568*   2080    2109
$OCNT   011004           2142*   2171*   2184*
$OMODE  011006           2137*   2141*   2146    2149*   2160*   2186*
$PASS   001100            543*
$PWRAD  011354           2315*
$PWRDN  011214            705    2282*   2310
$PWRMG  011350           2313*
$PWRUP  011266           2292    2298*
$QUES   001160            572*   1938    2014    2031    2109    2238    2241
```

DZNCB.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
$RDCHR  007722          1951*  2274
$RDDEC= ****** U        2277
$RDLIN  010042          1979*  2275
$RDOCT  011010          2202*  2276
$RDSZ = 000007          1972*
$R2A  = ****** U        2277
$SAVRE= ****** U        2277
$SAVR6  011364          2291*  2299   2300*  2301*  2319*
$SETUP= 000114           694*   702    703    705    707   1875   2037
$STUP = 177777           694*
$SWR  = 160000           404    405*   572   2316
$TKB    001146           565*  1534   1873   1884   1901   1955   1961
$TKS    001144           564*   753*  1873   1882   1898   1922*  1953   1959
$TN   = 000001           404*
$TPB    001152           567*  1549*  2098*  2109
$TPFLG  001157           571*  2056   2109
$TPS    001150           566*  1547   2096   2109
$TRAP   011150           703    2250*
$TRP  = 000011          2258*  2267*  2268*  2269*  2270*  2271   2272*  2273   2274*  2275*  2276*  2277*
$TRPAD  011172          2255   2265*
$TSTNM  001102           544*
$TTYIN  010276          1981   1982   1994   2012   2026   2030*
$TYPBN= ****** U        2270
$TYPDS= ****** U        2270
$TYPE   010342          2056*  2258   2266
$TYPEC  010512          1924   2077   2084   2091   2096*  2097
$TYPEX  010560          2102   2104   2107*
$TYPOC  010606          2140*  2267
$TYPON  010622          2139   2142*  2269
$TYPOS  010562          2135*  2268
$OFILL  011005          2136*  2140*  2150   2185*
.     = 013627           526*   530*   541*   575    700   1751*  1752*  1866   1873   2030*  2031   2037*  2109
                        2241   2294   2318
```



CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

D06


```
HALT      530   730   742  2058  2293  2317
INC       932   939   965   970   972   977  1036  1044  1094  1170  1177  1188  1200  1205  1212
         1222  1227  1234  1610  1635  1718  1744  1774  1785  1823  1935  2166  2174  2301
INCB     1292  1295  2105
IOT       411
JMP       534   797   836   838   840   841   843   856   859   861   864   866   868   874   890
          895   999   991   894   900   903   911   916   959  1002  1024  1067  1098  1103  1151
         1153  1539
JSR       724   726   745   831   932   839   842   845   846   860   862   976   893   892   893
          901   907   919   923   931   935   938   942   949   952   962   966   971   976  1056
         1089  1115  1116  1119  1122  1123  1127  1128  1140  1143  1147  1162  1169  1173  1176
         1190  1194  1187  1199  1204  1209  1213  1221  1226  1231  1235  1257  1261  1282  1284
         1290  1293  1296  1300  1318  1320  1420  1545  1565  1585  1597  1617  1662  1692  1695
         1729  1909  1912  1924  2077  2094  2091
MOV       697   701   703   704   705   706   709   710   711   712   717   719   720   721   731
          732   733   737   743   746   749   750   751   752   753   793   796   830   835   837
          844   848   869   872   875   979   894   895   898   909   914   929   930   937   944
          946   950   954   955   956   960   961   980   991   993   996   987   988   993  1003
         1008  1010  1011  1015  1025  1026  1048  1054  1055  1062  1071  1099  1093  1095  1109
         1111  1112  1113  1114  1117  1121  1125  1126  1130  1133  1138  1139  1152  1158  1175
         1180  1183  1185  1191  1243  1244  1245  1256  1259  1262  1291  1289  1291  1294  1297
         1313  1329  1330  1331  1335  1339  1343  1346  1347  1352  1354  1365  1366  1367  1371
         1372  1373  1374  1376  1386  1387  1388  1392  1394  1396  1404  1412  1421  1422  1433
         1435  1445  1453  1457  1468  1469  1478  1489  1490  1491  1492  1493  1499  1508  1512
         1520  1534  1544  1546  1549  1576  1590  1589  1596  1612  1633  1640  1654  1655  1656
         1657  1658  1659  1665  1669  1673  1676  1719  1721  1722  1723  1724  1726  1727  1729
         1732  1733  1745  1778  1800  1801  1802  1803  1806  1807  1808  1811  1815  1816  1993
         1911  1922  1951  1952  1979  1981  1992  2023  2024  2025  2026  2060  2061  2065  2090
         2135  2143  2144  2145  2151  2158  2176  2177  2178  2179  2180  2202  2203  2204  2205
         2206  2208  2209  2228  2229  2230  2231  2232  2250  2251  2255  2292  2293  2294  2295
         2286  2287  2288  2289  2290  2291  2292  2298  2299  2303  2304  2305  2306  2307  2308
         2309  2310  2311  2314
MOVB     1273  1274  1299  1309  1310  1528  1556  1560  1581  1595  1725  1731  1747  1810  1894
         1901  1955  1961  1985  1990  1996  2001  2016  2062  2090  2098  2136  2137  2140  2141
         2142  2146  2149  2150  2169  2212  2253
NEG      2147
RESET     744
ROL      1781  1787  2153  2155  2156  2157  2159  2219  2221  2223
ROR      1039  1081  1082  1083  1084  1660  1777  1791
RTI       718  1539  1923  1971  2027  2067  2181  2233  2316
RTS      1355  1377  1397  1407  1415  1426  1446  1459  1470  1490  1501  1521  1529  1550  1566
         1619  1643  1677  1697  1749  1794  1817  1831  2107  2256
SEC      1357  1379  1398
SUB      1000  1016  1019  1037  1042  1064  1499  1513  1593  1794
SWAB     1075  1298  1913  1829
TRAP     2255  2267  2268  2269  2271  2273  2274  2275  2276
TST       738   795   927   994  1021  1027  1034  1045  1072  1086  1107  1141  1149  1163  1135
         1241  1254  1263  1265  1269  1303  1306  1336  1344  1345  1431  1475  1507  1526  1579
         1583  1737  1915  1930  1988  1999  2022  2064  2072  2094  2164  2227  2234  2252
TSTB     1246  1301  1442  1455  1464  1510  1547  1636  1742  1992  1998  1953  1959  2056  2096
.ASCII    572   573  2570  2574  2578  2590  2593  2595
.ASCIZ    574  2031  2032  2033  2035  2710  2716  2723  2732  2738  2744  2749  2754
.BLKB    2030
.BYTE     544   545   550   551   559   560   568   569   570   571  1605  2029  2029  2192  2193
         2184  2185  2361  2362  2363  2364  2366  2369  2370  2373  2376  2379  2383  2387  2391
         2396  2401  2406  2412  2415  2419  2421  2424  2429  2431  2436  2439  2443  2446  2451
```

E06


```
          2454    2459    2462    2467    2471    2476    2480    2485    2489    2494    2499    2504    2509    2515    2520
          2526    2531    2537    2542    2549
.DSABL    1940
.ENABL     394    1973
.END      2759
.ENDC      399     410     502     516     535     538     542     544     572     576     694     701     702     703     705
           707     723    1324    1329    1361    1365    1383    1386    1402    1404    1410    1412    1419    1420    1429
          1431    1449    1452    1462    1464    1473    1475    1483    1489    1504    1507    1524    1526    1532    1534
          1542    1544    1553    1556    1569    1575    1623    1633    1647    1652    1686    1699    1700    1715    1756
          1771    1798    1800    1870    1873    1874    1876    1904    1940    1944    1972    1973    1981    1993    1996
          2014    2031    2037    2039    2042    2062    2110    2113    2199    2191    2197    2241    2242    2245    2251
          2254    2256    2267    2269    2269    2270    2271    2272    2273    2274    2275    2276    2277    2278    2281
          2290    2291    2297    2303    2304    2314    2316    2320
.EQUIV     410     411     419     434     435     464     465     466     467     468     469     470     471     472     473
           492     493     494     495     496     497     498     499     500     501
.EVEN     1607    1753    2037    2554    2597
.IF        395     408     474     502     533     537     541     543     572     575     576     694     696     701     703
           705     707     723    1323    1328    1360    1364    1382    1385    1401    1403    1409    1411    1417    1419
          1428    1430    1448    1451    1461    1463    1472    1474    1492    1488    1503    1506    1523    1525    1531
          1533    1541    1543    1552    1555    1568    1574    1622    1632    1646    1651    1685    1699    1699    1714
          1755    1770    1797    1799    1869    1872    1974    1875    1876    1904    1943    1944    1972    1990    1982
          1986    1997    2030    2031    2037    2038    2041    2062    2109    2112    2197    2190    2193    2203    2241
          2244    2250    2254    2258    2267    2268    2269    2270    2271    2273    2274    2275    2276    2277    2280
          2290    2291    2296    2303    2304    2312    2314    2316    2320
.IFF       408     539     541     543     572     576     701    1324    1329    1361    1365    1393    1396    1402    1404
          1410    1412    1418    1420    1429    1431    1449    1452    1462    1464    1473    1475    1483    1489    1504
          1507    1524    1526    1532    1534    1542    1544    1553    1556    1569    1575    1623    1633    1647    1652
          1686    1699    1700    1715    1756    1771    1798    1800    1870    1973    1976    1944    1946    1951    1972
          1973    1982    2014    2030    2039    2042    2110    2113    2199    2191    2242    2245    2251    2279    2291
          2297    2312
.IFT      1946    1951    2214    2234    2241
.IFTF     1931    1944    1947    2210    2218    2240
.IIF       394     399     404     405     530     575     702    1973    1894    2022    2031    2037    2109    2241    2256
          2267    2268    2269    2271    2273    2274    2275    2276
.IRP       694    2204    2230    2284    2290    2303    2304
.LIST      394     516     530     572     694     707    1972    2259    2266    2267    2268    2269    2270    2271    2272
          2273    2274    2275    2276    2277
.MACRO     535    2258
.MCALL     394     516     707
.NLIST     394     516     530     572     694     707    1972    2259    2266    2267    2268    2269    2270    2271    2272
          2273    2274    2275    2276    2277
.PAGE      535     576     619     654     692     755     917    1322    2320    2555    2589
.REM         1
.REPT      530    1833
.SBTTL     406     524     533     535     576     692     695    1322    1970    2039    2110    2199    2242    2259    2279
          2320    2555    2589    2708
.TITLE     394
.WORD      530     531     532     543     546     547     549     549     552     553     554     555     556     557     558
           561     562     563    2106    2186    2237    2240    2313    2315


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

DZNCB.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
*.DZNCB.SEQ.SOL CRF=DZNCB.P11
RUN-TIME:  32.22 4 SECONDS
RUN-TIME RATIO: 332/80=8.5
CORE USED:  19K  (37 PAGES)
```

G06

Spooler runtime 11 Seconds, 328 KCS, 297 disk reads, 3 disk writes, 70 pages