NCV11

NCV-11 EXER

CZNCDB0

AH-E777B-MC

FICHE 1 OF 1

JAN 1980

digital

MADE IN USA

IDENTIFICATION

Product Code: AC-E776B-MC

Product Name: CZNCDBO NCV-11 EXERCISER

Date: AUGUST 1979

MAintainer: Diagnostic Engineering

0.0 TABLE OF CONTENTS

1.0 ABSTRACT

The 'B' version corrected the 'D' command. It also corrected problems when no VSV01 display was present.
This is a self contained program designed to convert data from the GAMMA camera via the NCV11 interface and display it on a 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. THE PROGRAM IS NOT CHAINABLE/SCRIPTABLE UNDER XXDP/APT.

2.0 REQUIREMENTS

2.1 Equipment

1. PDP-11 FAMILY OR LSI-11 FAMILY Computer
2. I/O Terminal (i.e., LA36)
3. NCV11 Interface
4. Gamma Camera (OR SUBSTITUTE)

2.2 Equipment - Optional

1. H3060 Joystick
2. VSV01 DISPLAY

2.3 PRELIMINARY PROGRAMS

MAINDEC-11-CZNCC 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.

SEQ 0004

4.0 STARTING PROCEDURE

Loading address 200 and starting will initialize the system, IDENTIFY THE PROGRAM and wait a keyboARd command.

5.0 OPERATOR OPTIONS

5.1 OPERATOR KEYBOARD OPTIONS

TYPE 'H'- HELP OPERATOR AND DISPLAY OR TYPE THIS LIST
Type 'N'- Collect new data from camera - clears core matrix FIRST
Type 'C'- Collect data from camera - starts NCV11, previous core matrix data not cleared
TYPE 'X'- SELECT ANOTHER CAMERA CHANNEL
Type 'S'- Stop NCV11 data collection - also terminates free RUN MODE
Type 'Z'- Zoom - Set NCV11 TO GAIN 2
Type 'R'- Regular - SET NCV11 TO GAIN 1 (DEFAULT COND)
Type 'W'- Select VSV01 display using 1st bit map
Type 'M'- Select VSV01 display USING 2nd bit map
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 'X'- SELECT ANOTHER CAMERA CHANNEL
Type 'G'- Initialize everything - start fresh
Type "J"- Joystick calibration NCV11 - use "Cntrl C" to terminate mode
TYPE 'T'- DISPLAY INTENSITY TEST PATTERN ON SELECTED DISPLAY - THIS FEATURE IS FOR DISPLAY VERIFCATION ONLY
TYPE 'O'- OTHER TERMINAL TO CONTROL THE PROGRAM

Type "Cntrl & C" - Abort whatever - back to keyboard monitor

5.2 OPERATOR SWITCH REGISTER OPTION(S)

NONE

SEQ 0005

6.0 RECOMMENDED OPERATOR ACTION

6.1 Camera/A017 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 A017 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 "Z" delay circuit should be adjusted.

6. 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 CAMERA.

6.2 Joystick Setup

1. Select Joystick configration by typing "J".

2. Adjust the Y Pot on the H3060 Joystick so that the HORIZONTAL cross hair provide UNIFORM access within the box drawn on the screen.
ADJUST THE X POT ON THE H3060 SO THAT THE VERTICAL CROSS HAIR WILL ALSO BE UNIFORM EXCEPT THAT DUE TO A HARDWARE OFFSET WILL NOT REACH THE RIGHT EDGE AND EXCEED THE LEFT EDGE BY THE SAME AMOUNT.
Ideally, a physcial center of the Joystick will provide a point approximately centered in the box.

3. Check that when the Joystick interrupt SWITCH is depressed the CROSS HAIRS DO not follow the Joystick.

4. A "Cntrl C" will return user back to the keyboard monitor.

7.0 MISCELLANEOUS

7.1 Device Bus Address Modifications

NCV11 — Modify location 'NCVADR' if base bus address is not 772760

VSV01 — Modify location 'VTVADR' if base bus address (CHARACTER GENERATOR) IS NOT 772600
MODIFY LOCATION 'VTMADR' IF BASE BUS ADDRESS (BIT MAP) IS NOT 772620

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 20(8) provides about 10 SECONDS ON AN LSI-11 CPU.

7.3 Error Reporting

1. INCORRECT KEYBOARD COMMANDS ARE R SPONDED WITH '?'

2. A MESSAGE WILI BE REPORTED IF ANY SELECTED DEVICE (SEE SECTION 7.1) DOES NOT RESPOND.

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 NCV11 interface and display this data on 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 NCV11 is receiving X & Y data from the gamma camera (looking AT 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 NCV11 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 ONE OF 16 intensity levelS. 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
CAMERA	-	SELECTED CAMERA CHANNEL
PB	-	UP/DN INDICATOR OF THE PUSH BUTTON STATE DURING LAST COLLECTION
JB	-	UP/DN INDICATOR OF THE JOY BUTTON STATE DURING LAST COLLECTION
Z Count	-	32 bit counter of EVENT 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 Mode

Selecting the TEST takes advantage of the Joystick mode provided by the NCV11. This mode selects the JOYSTICK inputs and the NCV11 performs like an A/D with the X & Y CONVERTED values available in X-Y holding register. The X and Y data is applied to the display and should conform to the requirements in section 6.2. The VSV01 uses the X & Y cross hairs as the DISPLAY indicator.

9.0 LISTING

```
                                ;DEVELOPED USING SYSMAC.C3
                                .TITLE  CZNCDB  NCV11 EXERCISER
                                ;*COPYRIGHT (C) 1979
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY R.SHOOP
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
172760                                  ABASE=172760            ;DEFAULT NCV11 ADDRESS
                                .SBTTL  BASIC DEFINITIONS

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

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

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

```
(1)           100000                  SW15=   100000
(1)           040000                  SW14=   40000
(1)           020000                  SW13=   20000
(1)           010000                  SW12=   10000
(1)           004000                  SW11=   4000
(1)           002000                  SW10=   2000
(1)           001000                  SW09=   1000
(1)           000400                  SW08=   400
(1)           000200                  SW07=   200
(1)           000100                  SW06=   100
(1)           000040                  SW05=   40
(1)           000020                  SW04=   20
(1)           000010                  SW03=   10
(1)           000004                  SW02=   4
(1)           000002                  SW01=   2
(1)           000001                  SW00=   1
(1)                                   .EQUIV  SW09,SW9
(1)                                   .EQUIV  SW08,SW8
(1)                                   .EQUIV  SW07,SW7
(1)                                   .EQUIV  SW06,SW6
(1)                                   .EQUIV  SW05,SW5
(1)                                   .EQUIV  SW04,SW4
(1)                                   .EQUIV  SW03,SW3
(1)                                   .EQUIV  SW02,SW2
(1)                                   .EQUIV  SW01,SW1
(1)                                   .EQUIV  SW00,SW0
(1)
(1)                                   ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)           100000                  BIT15=  100000
(1)           040000                  BIT14=  40000
(1)           020000                  BIT13=  20000
(1)           010000                  BIT12=  10000
(1)           004000                  BIT11=  4000
(1)           002000                  BIT10=  2000
(1)           001000                  BIT09=  1000
(1)           000400                  BIT08=  400
(1)           000200                  BIT07=  200
(1)           000100                  BIT06=  100
(1)           000040                  BIT05=  40
(1)           000020                  BIT04=  20
(1)           000010                  BIT03=  10
(1)           000004                  BIT02=  4
(1)           000002                  BIT01=  2
(1)           000001                  BIT00=  1
(1)                                   .EQUIV  BIT09,BIT9
(1)                                   .EQUIV  BIT08,BIT8
(1)                                   .EQUIV  BIT07,BIT7
(1)                                   .EQUIV  BIT06,BIT6
(1)                                   .EQUIV  BIT05,BIT5
(1)                                   .EQUIV  BIT04,BIT4
(1)                                   .EQUIV  BIT03,BIT3
(1)                                   .EQUIV  BIT02,BIT2
(1)                                   .EQUIV  BIT01,BIT1
(1)                                   .EQUIV  BIT00,BIT0
(1)
(1)                                   ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
```

```
(1)           000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
(1)           000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)           000014                  TBITVEC=14              ;;'T' BIT
(1)           000014                  TRTVEC= 14              ;;TRACE TRAP
(1)           000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
(1)           000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)           000024                  PWRVEC= 24              ;;POWER FAIL
(1)           000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
(1)           000034                  TRAPVEC=34              ;;''TRAP'' TRAP
(1)           000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
(1)           000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
(1)           000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
 12
 13
 14                                    ;SOME COMMON PROGRAM VALUES AND EQUATES
 15
 16           020000                  MATRIX=20000            ;STARTING ADRS OF MATRIX DATA
 17
 18                                    .SBTTL  TRAP CATCHER
(1)
(1)           000000                          .=0
(1)                                   ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
(1)                                   ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
(1)                                   ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
(1)           000174                          .=174
(1)  000174   000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
(1)  000176   000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
(1)                                   .SBTTL  STARTING ADDRESS(ES)
(1)  000200   000137  001326                  JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
 19           000100                  .=100
 20  000100   000104  000340  000002        104,340,RTI     ;LSI-11 B EVENT
```

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

ERROR POINTER TABLE SEQ 0013

```
  (1)                                   .SBTTL  ERROR POINTER TABLE
  (1)
  (1)                                   ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
  (1)                                   ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
  (1)                                   ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
  (1)                                   ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
  (1)                                   ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
  (1)
  (1)                                   ;*      EM              ;;POINTS TO THE ERROR MESSAGE
  (1)                                   ;*      DH              ;;POINTS TO THE DATA HEADER
  (1)                                   ;*      DT              ;;POINTS TO THE DATA
  (1)                                   ;*      DF              ;;POINTS TO THE DATA FORMAT
  (1)
  (1)
  (1)  001164                           $ERRTB:
   22                                   ;THIS PROGRAM DOES NOT USE THE ABOVE ERROR TABLE
   23
   24
   25
   26
   27
   28
   29
   30                                           ;NCV11 BUS ADRS ASSIGNMENTS - MODS ARE TO BE MADE HERE
   31
   32  001164  172760                   NCADR:  ABASE
   33
   34                                           ;VSV01 BUS ADRS ASSIGNMENTS (CHAR GEN) - MODS ARE TO BE MADE HERE
   35
   36  001166  172600                   VTVADR: 172600
   37
   38                                           ;VSV01 BUS ADRS ASSIGNMENTS (BIT MAP) - MODS ARE TO BE MADE HERE
   39
   40  001170  172620                   VTMADR: 172620  ;FIRST MAP  2ND MAP IS 20 HIGHER
   41
   42                                           ;COUNTER FOR FREE RUN MODE
   43
   44  001172  000024                   TIME:   20.     ;LOOP COUNTER FOR 'F'' MODE
   45
   46  001174  000060                   TKVEC0: TKVEC           ;CONSOLE TERMINAL VECTOR
   47  001176  000062                   TKVEC1: TKVEC+2         ; ADDRESS AN PRIORITY
```

```
                                        ;NCV11 REGISTER ADDRESS POINTERS
001200  172760          NCCSR:  ABASE                 ;NCV11 STATUS CONTROL REGISTER
001202  172762          NCOFF:  ABASE+2               ;NCV11 OFFSET REGISTER
001204  172764          NCWCR:  ABASE+4               ;NCV11 WORD COUNT REGISTER
001206  172766          NCBAR:  ABASE+6               ;NCV11 BUS ADDRESS REGISTER / LOW Z COUNT
001210  172770          NCSFR:  ABASE+10              ;NCV11 SPECIAL FUNCTION / JOYSTICK STATUS REGISTER
001212  172772          NCADM:  ABASE+12              ;NCV11 ADDRESS MAKER REGISTER
001214  172774          NCJOY:  ABASE+14              ;NCV11 JOYSTICK DATA REGISTER
001216  172776          NCBAR1: ABASE+16              ;NCV11 SPARE <SAME AS NCBAR>
                                        ;VSV01 REGISTER ADDRESS POINTERS (CHARACTER GENERATOR)
001220  172600          VTVCRG: 172600                ;CHAR/STATUS
001222  172602          VTVCHP: 172602                ;CROSS HAIR POS
001224  172604          VTVPOS: 172604                ;CHAR POS
                                        ;VSV01 REGISTER ADDRESS POINTERS (BIT MAP)
001226  172620          VTVCSR: 172620                ;COMMAND/STATUS
001230  172622          VTVMAP: 172622                ;MAP ADRS
001232  172624          VTVPX:  172624                ;PIXEL WD
001234  172626          VTVPX1: 172626                ;PIXEL BYTE
001236  172630          VTVINT: 172630                ;INTENSITY LOOK UP

                                        ;COMMON PROGRAM TAGS AND STORAGE LOCATIONS
001240  000000          TEMP0:  0                     ;COMMON UTILITY LOC
001242  000000          TEMP1:  0                     ;COMMON UTILITY LOC
001244  000000          TEMP2:  0                     ;COMMON UTILITY LOC
001246  000000          DUMMY1: 0                     ;OCTAL TEMP LOC.
001250  000000          DUMMY2: 0                     ;OCTAL TEMP LOC.
001252  000000          INTLUT: 0
001254  006400          VTVSAV: 6400                  ;VSV01 SET UP - FULL SCREEN, MONO(BLK/WHT), ENA DISPLAY
001256  000000          MRXADR: 0                     ;CURRENT CORE ADRS OF CELL BEING DISPLAYED
001260  000000          MAPADR: 0                     ;CURRENT ADRS OF BIT MAP LD
001262  000003          PIXCNT: 3                     ;CURRENT PIXEL BYTE COUNT
001264  000000          PIXASM: 0                     ;4 PIXELS ASSEMBLED HERE BEFORE BIT MAP LD
001266  000000          KBUFF:  0                     ;CONTAINS KEYBOARD CHAR TYPED
001270  000000          TTYOUT: 0                     ;OUTPUT CHAR TO PRINTER
001272  000000          THLO:   0                     ;LOW THRESHOLD VALUE APPLIED TO MATRIX CELLS
001274  177777          THHI:   177777                ;HIGH THRESHOLD VALUE APPLIED TO MATRIX CELLS
001276  000000          COMSAV: 0                     ;SAVED NCV11 CSR CONTENTS WHEN DISPLAYING
001300  000000          CAMERA: 0                     ;CAMERA SELECTION IN BITS 8-9
001302  000000          GAIN:   0                     ;GAIN 1=0, GAIN 2=BIT10
001304  010000          TOTSIZ: 4096.                 ;TOTAL # OF CELLS IN MATRIX (ISTOPE A OR B)
001306  000000          CHARCT: 0                     ;COUNTS TOTAL # OF CELLS IN MATRIX (ISOTOPE A OR B)
001310  000000          CPERL:  0                     ;CONTAINS LARGEST CELL OF MATRIX WITHIN THRESHOLDS
001312  000000          ROWCNT: 0                     ;COUNTS 64 CELLS EACH ROW OF CORE MATRIX
001314  020000          TABLEX: MATRIX                ;CONTAINS STARTING ADRS OF MATRIX OF SELECTED ISOTOPE
001316  000000          FREERN: 0                     ;NON-ZERO SAYS COLLECT NEW DATA & DISPLAY IN FREE RUN MO
001320  000000          MSELCT: 0                     ;0 SAYS 1ST BIT MAP, -1 SAYS 2ND BIT MAP(VSV01)
001322  000000          BELLEN: 0                     ;0 SAYS RING BELL ON NO CONVERT, -1 SAYS DON'T
001324  000000          NOVSV:  0                     ;NON-ZERO INDICATES NO VSV01 DISPLAY
```

```
  102  001326                                START:
  (1)                                        .SBTTL  INITIALIZE THE COMMON TAGS
  (1)                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
  (1)  001326  012706  001100                        MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
  (1)  001332  005026                                CLR     (R6)+           ;;CLEAR MEMORY LOCATION
  (1)  001334  022706  001140                        CMP     #SWR,R6 ;;DONE?
  (1)  001340  001374                                BNE     .-6             ;;LOOP BACK IF NO
  (1)  001342  012706  001100                        MOV     #STACK,SP       ;;SETUP THE STACK POINTER
  (1)                                        ;;INITIALIZE A FEW VECTORS
  (1)  001346  012737  007432  000034                MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
  (1)  001354  012737  000340  000036                MOV     #340,@#TRAPVEC+2;LEVEL 7
  (1)  001362  012737  007512  000024                MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
  (1)  001370  012737  000340  000026                MOV     #340,@#PWRVEC+2 ;;LEVEL 7
  (2)                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
  (2)                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
  (2)  001376  013746  000004                        MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
  (2)  001402  012737  001436  000004                MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
  (2)  001410  012737  177570  001140                MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
  (2)  001416  012737  177570  001142                MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
  (2)  001424  022777  177777  177506                CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
  (2)  001432  001012                                BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
  (2)                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
  (2)  001434  000403                                BR      65$             ;;BRANCH IF NO TIMEOUT
  (2)  001436  012716  001444          64$:          MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
  (2)  001442  000002                                RTI
  (2)  001444  012737  000176  001140  65$:          MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
  (2)  001452  012737  000174  001142                MOV     #DISPREG,DISPLAY
  (2)  001460  012637  000004          66$:          MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
  (1)
  103  001464  104401  010120                        TYPE    ,MSG1           ;GO IDENTIFY PROGRAM
  104  001470  004737  004546                        JSR     PC,SELCTA       ;DEFAULT TO VSV01 DISPLAY IF THERE
  105  001474  103004                                BCC     3$              ;BR IF DISPLAY VSV01 IS THERE
  106  001476  005237  001324                        INC     NOVSV           ;SAVE THE FACT NO VSV01 CONNECTED
  107  001502  104401  010242                        TYPE    ,MSG3           ;GO TYPE 'BUS TIMEOUT ER - VSV01 DISPLAY'
  108  001506  013700  001164          3$:           MOV     NCADR,R0        ;GET NCV11 BASE ADRS
  109  001512  012701  001200                        MOV     #NCCSR,R1       ;GET POINTER ADRS
  110  001516  010021                    NCSET:      MOV     R0,(R1)+        ;SET UP NCV11 REG ADRS PTRS
  111  001520  062700  000002                        ADD     #2,R0           ;BUMP REG ADRS
  112  001524  022701  001220                        CMP     #VTVCRG,R1      ;ALL SET UP?
  113  001530  001372                                BNE     NCSET           ;BR IF NOT
  114  001532  012737  001546  000004                MOV     #1$,ERRVEC      ;SET UP TIMEOUT ADRS
  115  001540  005777  177434                        TST     @NCCSR          ;WILL TRAP TO LOC 4 IF NOT THERE
  116  001544  000406                                BR      START1          ;BR IF THERE
  117  001546  022626                    1$:         CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
  118  001550  104401  010400                        TYPE    ,MSG5           ;GO TYPE 'BUS TIMEOUT ER - NCV11'
  119  001554  000000                                HALT                    ;NCV11 NOT SEEN AT ASSIGNED BUS ADRS
  120  001556  000137  001326                        JMP     START
  121  001562  012737  000006  000004  START1:      MOV     #ERRVEC+2,@#ERRVEC      ;RESTORE ERR TRAP LOC TO PT TO 6
  122  001570  000005                                RESET                   ;CLR WORLD
  123  001572  004737  004766                        JSR     PC,NCSTP1       ;GO STOP NCV11 & CLR CORE MATRIX
  124  001576  012777  020000  177376                MOV     #MATRIX,@NCOFF  ;SET OFFSET(ADRS 20000)
  125  001604  005037  001302                        CLR     GAIN            ;REGULAR GAIN
  126  001610  012737  020000  001314                MOV     #MATRIX,TABLEX  ;ISOTOPE A
  127  001616  005037  001272                        CLR     THLO            ;CLEAR LOWER THRESHOLD
  128  001622  012737  177777  001274                MOV     #177777,THHI    ;CEILING THRESHOLD
  129  001630  012777  005220  177336                MOV     #KBINT,@TKVEC0  ;SET KEYBOARD INTR RETURN ADRS
```

```
                                                MOV     #340,@TKVEC1    ;SET UP NEW PRIORTY ON INTR
                                                MOV     #100,@STKS      ;SET INTR ENABLE
                                                TYPE    ,MSG6           ;GO ASK FOR KEYBOARD COMMAND(S)

                                        ;THIS IS A LIST OF KEYBOARD COMMANDS FOR THE NCV11/DISPLAY/JOYSTICK

                                                ;H      HELP FRAME
                                                ;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
                                                ;X      CHANGE CAMERA
                                                ;S      STOP COLLECTION
                                                ;G      INITIALIZE EVERYTHING
                                                ;Z      ZOOM - SET GAIN TO 2
                                                ;R      REGULAR - SET GAIN TO 1
                                                ;A      DISPLAY ISOTOPE A
                                                ;B      DISPLAY ISOTOPE B
                                                ;W      SELECT VSV01 DISPLAY USING 1ST BIT MAP
                                                ;M      SELECT VSV01 DISPLAY USING 2ND BIT MAP
                                                ;F      FREE RUN MODE - COLLECT & DISPLAY NEW DATA CONTINUALLY
                                                ;J      JOYSTICK CALIBRATION
                                                ;T      DISPLAY INTENSITY TEST IMAGE ON SELECTED DISPLAY
                                                ;O      OTHER TERMINAL TO CONTROL PROGRAM
                                                ;CNTRL C  ABORT WHATEVER - BACK TO KEYBOARD MONITOR

                                        ;THIS CODE WILL DISPATCH PROGRAM TO THE PROPER
                                        ;ROUTINE VIA THE DISPATCH TABLE 'RTABLE'
001656  005037  001316          LISEN:  CLR     FREERN          ;KNOCK DOWN FREE RUN MODE IF SET
001662  012706  001100                  MOV     #STACK,SP       ;RESET STACK PTR
001666  005037  001266                  CLR     KBUFF           ;INSURE NO KEYBOARD GARBAGE
001672  005046                          CLR     -(SP)           ;PUSH LEVEL 0 ONTO STACK
001674  012746  001702                  MOV     #LISN,-(SP)     ;TO BE LSI-11 COMPATABLE
001700  000002                          RTI                     ;FAKE RTI TO LOWER PRIORITY
001702  013700  001266          LISN:   MOV     KBUFF,R0        ;LOOK FOR CHAR
001706  001775                          BEQ     LISN            ;WAIT FOR ONE
001710  005037  001266                  CLR     KBUFF           ;
001714  020027  000101                  CMP     R0,#101         ;WAS IT A CHAR?
001720  103445                          BCS     B00B00          ;NOT A GOOD CHAR
001722  042700  177740                  BIC     #177740,R0      ;ELIMINATE LOWER CASE
001726  020027  000033                  CMP     R0,#33          ;IS IT AN ALPHA CHAR?
001732  103040                          BCC     B00B00          ;BR IF NOT
001734  006300                          ASL     R0              ;MAKE UP WORD OFFSET
001736  005760  001746                  TST     RTABLE-2(R0)    ;IS IT A LEGAL COMMAND?
001742  001434                          BEQ     B00B00          ;BR IF NOT
001744  000170  001746                  JMP     @RTABLE-2(R0)   ;GO DO IT
```

```
  179
  180                                           .SBTTL  KEYBOARD DISPATCH TABLE
  181
  182  001750  002054                 RTABLE: ROUTA                   ;'A' DISPLAY ISOTOPE A
  183  001752  002066                         ROUTB                   ;'B' DISPLAY ISOTOPE B
  184  001754  002100                         ROUTC                   ;'C' COLLECT DATA
  185  001756  002110                         ROUTD                   ;'D' DISPLAY DATA
  186  001760  002114                         ROUTE                   ;'E' ERASE SCOPE
  187  001762  002124                         ROUTF                   ;'F' FREE RUN MODE(COLLECT NEW DATA & DISPLAY CONTINOUSL
  188  001764  002220                         ROUTG                   ;'G' INITIALIZE EVERYTHING
  189  001766  002230                         ROUTH                   ;'H' HELP THE OPERATOR
  190  001770  000000                         0                       ;'I' BOOBOO
  191  001772  002302                         ROUTJ                   ;'J' GO TO NCV11 JOYSTICK CALIBRATION
  192  001774  000000                         0                       ;'K' BOOBOO
  193  001776  002320                         ROUTL                   ;'L' GET LOWER THRESHOLD FROM KEYBOARD
  194  002000  002360                         ROUTM                   ;'M' SELECT VSV01 DISPLAY USING 2ND BIT MAP
  195  002002  002420                         ROUTN                   ;'N' GET NEW DATA FROM CAMERA
  196  002004  002434                         ROUTO                   ;'O' GET ADDRESS OF OTHER TERMINAL
  197  002006  000000                         0                       ;'P' BOOBOO
  198  002010  000000                         0                       ;'Q' BOOBOO
  199  002012  002614                         ROUTR                   ;'R' REGULAR GAIN =1
  200  002014  002632                         ROUTS                   ;'S' STOP COLLECTION
  201  002016  002650                         ROUTT                   ;'T' DISPLAY INTENSITY TEST IMAGE ON SELECTED DISPLAY
  202  002020  002664                         ROUTU                   ;'U' GET UPPER THRESHOLD FROM KEYBOARD
  203  002022  000000                         0                       ;'V' BOOBOO
  204  002024  002724                         ROUTW                   ;'W' SELECT VSV01 DISPLAY USING 1ST BIT MAP
  205  002026  002762                         ROUTX                   ;'X' CHANGE CAMERA CHANNEL
  206  002030  000000                         0                       ;'Y' BOOBOO
  207  002032  003102                         ROUTZ                   ;'Z' ZOOM - GAIN   2
```

```
                                        .SBTTL  COMMAND DECODER
                                ;REPORTS ILLEGAL KEYBOARD CHARACTERS

002034  012737  000077  001270  BOOBOO: MOV     #77,TTYOUT      ;SET UP '?'
002042  004737  005300                  JSR     PC,TYPO         ;TYPE IT
002046  004737  005260                  JSR     PC,TYPCR        ;DO A 'CR'
002052  000713                          BR      LISN            ;GO LOOK FOR GOOD CHAR

002054  012737  020000  001314  ROUTA:  MOV     #MATRIX,TABLEX  ;WILL DISPLAY ISOTOPE A
002062  000137  003122                  JMP     CHANGE          ;GO DO IT
002066  012737  040000  001314  ROUTB:  MOV     #MATRIX+20000,TABLEX    ;WILL DISPLAY ISOTOPE B
002074  000137  003122                  JMP     CHANGE          ;GO DO IT
002100  004737  004704          ROUTC:  JSR     PC,NCSTRT       ;GO START NCV11
002104  000137  001702                  JMP     LISN            ;COLLECT DATA UNTIL KEYBRD COMMAND
002110  000137  003122          ROUTD:  JMP     CHANGE          ;GO DISPLAY DATA
002114  004737  005020          ROUTE:  JSR     PC,ERASE        ;GO ERASE SCOPE
002120  000137  001702                  JMP     LISN            ;GO WAIT ON NEXT COMMAND
002124  012737  177777  001316  ROUTF:  MOV     #-1,FREERN      ;SELECT FREE RUN MODE
002132  004737  004766                  JSR     PC,NCSTP1       ;GO STOP NCV11 & CLR CORE MATRIX AREA
002136  004737  004704                  JSR     PC,NCSTRT       ;GO START NCV11
002142  005000                          CLR     R0              ;SET UP TIMER
002144  013701  001172                  MOV     TIME,R1         ;SET UP GROSS TIMER VALUE
002150  005300          1$:     DEC     R0                      ;COUNT
002152  001376                  BNE     1$                      ;WAIT FOR ZERO
002154  042737  000040  001266          BIC     #BIT5,KBUFF     ;LOOK FOR POSSIBLE STOP KEY - RID LOWER CASE
002162  022737  000123  001266          CMP     #123,KBUFF      ;STOP BEEN STRUCK?
002170  001007                          BNE     2$              ;BR IF NOT
002172  052777  000400  177010          BIS     #BIT8,@NCSFR    ;STOP NCV11
002200  005077  176774                  CLR     @NCCSR          ;ZERO NCV CSR
002204  000137  001656                  JMP     LISEN           ;GO AWAIT NEXT COMMAND
002210  005301          2$:     DEC     R1                      ;DO LOOP AGAIN?
002212  001356                  BNE     1$                      ;BR IF SO
002214  000137  003122                  JMP     CHANGE          ;NOW GO DISPLAY DATA JUST COLLECTED
002220  004737  005020          ROUTG:  JSR     PC,ERASE        ;GO ERASE SELECTED DISPLAY
002224  000137  001562                  JMP     START1          ;INITIALIZE AND START FRESH
002230  005737  001324          ROUTH:  TST     NOVSV           ;CHECK IF DISPLAY
002234  001016                          BNE     1$              ;BR IF NO VSV01
002236  004737  005020                  JSR     PC,ERASE        ;CLEAR THE SCREEN
002242  000240                          NOP
002244  000240                          NOP
002246  012777  002000  176750          MOV     #2000,@VTVPOS   ;POSITION THE READ-OUT
002254  004537  005316                  JSR     R5,VTWRIT       ;TELL THE OPERATOR VIA VSV01
002260  010640                          HELP0
002262  004537  005316                  JSR     R5,VTWRIT
002266  011336                          HELP1
002270  000402                          BR      2$
002272  104401  011336          1$:     TYPE,   HELP1           ;NO VSV01 - TELL OPERATOR SHORT LIST
002276  000137  001702          2$:     JMP     LISN
002302  005737  001324          ROUTJ:  TST     NOVSV           ;CHECK IF VSV01 PRESENT
002306  001402                          BEQ     1$              ;BR IF PRESENT
002310  000137  001702                  JMP     LISN            ;FORGET IF NO DISPLAY
002314  000137  004332          1$:     JMP     TJOY            ;CALB. THE JOYSTICK GO TO IT
002320  012746  000340          ROUTL:  MOV     #340,-(SP)      ;PUSH HIGH PSW ON STACK
002324  012746  002332                  MOV     #1$,-(SP)       ;PUSH RETURN ADDRESS
002330  000002                          RTI                     ;FAKE RTI TO RAISE PSW
002332  104401  010161          1$:     TYPE    ,MSG2           ;ASK FOR OCTAL DATA
```

```
                                        RDOCT                   ;GO GET IT
002336  104411                        MOV     (SP)+,THLO      ;SAVE IT
002340  012637  001272                CLR     -(SP)           ;PUSH LOW PSW
002344  005046                        MOV     #2$,-(SP)
002346  012746  002354                RTI
002352  000002                2$:     JMP     CHANGE          ;GO DISPLAY
002354  000137  003122        ROUTM:  TST     NOVSV           ;TEST IF VSV01 DETECTED ?
002360  005737  001324                BNE     1$              ;BR IF NOT
002364  001013                        MOV     #-1,MSELCT      ;SELECT 2ND BIT MAP
002366  012737  177777  001320        JSR     PC,SELCTA       ;RELOAD ADDRESSES
002374  004737  004546                NOP
002400  000240                        MOV     VTVCSR,R0       ;GET 2ND BIT MAP ADRS
002402  013700  001226                BIC     #BIT8,-20(R0)   ;TURN OFF 1ST BIT MAP
002406  042760  000400  177760        JMP     LISN            ;GO AWAIT NEXT COMMAND
002414  000137  001702        1$:     JSR     PC,NCSTP1       ;GO STOP NCV11 & CLR CORE MATRIX
002420  004737  004766        ROUTN:  JSR     PC,NCSTRT       ;START THE NCV11
002424  004737  004704                JMP     LISN            ;COLLECT DATA & AWAIT NEXT KEYBRD COMMAND
002430  000137  001702        ROUTO:  MOV     #340,-(SP)      ;RAISE PS
002434  012746  000340                MOV     #1$,-(SP)
002440  012746  002446                RTI
002444  000002                1$:     TYPE,   MSG4            ,ASK OPR FOR ADDRESS
002446  104401  010324                RDOCT                   ;GET HIS INPUT
002452  104411                        MOV     (SP)+,R0        ;GET ADDRESS
002454  012600                        BNE     2$              ;BR IF NOT ''CR'' OR 0
002456  001002                        JMP     BOOBOO          ;FAT FINGER OPERATOR
002460  000137  002034        2$:     MOV     @#ERRVEC,-(SP)  ;SAVE LOC 4
002464  013746  000004                MOV     #4$,@#ERRVEC    ;SAVE IF WRONG ADDRESS BY OPR.
002470  012737  002576  000004        TST     (R0)            ;REF. THE NEW CONSOLE ADDR.
002476  005710                        MOV     #$TKS,R1        ;GET OLD ADDR. POINTER
002500  012701  001144                MOV     R0,(R1)+        ;LOAD NEW ADDRESS
002504  010021                        TST     (R0)+           ;BUMP ADDRESS
002506  005720                (1)     MOV     R0,(R1)+        ;LOAD NEW ADDRESS
002510  010021                (1)     TST     (R0)+           ;BUMP ADDRESS
002512  005720                (1)     MOV     R0,(R1)+        ;LOAD NEW ADDRESS
002514  010021                (1)     TST     (R0)+           ;BUMP ADDRESS
002516  005720                (1)     MOV     R0,(R1)+        ;LOAD NEW ADDRESS
002520  010021                        MOV     (SP)+,@#ERRVEC  ;RESTORE LOC 4.
002522  012637  000004                TYPE,   MSG8            ;ASK OPR FOR ADDRESS
002526  104401  010555                RDOCT
002532  104411                        MOV     (SP)+,R0        ,GET VALUE
002534  012600                        BNE     3$              ;BR IF VALID
002536  001002                        JMP     BOOBOO          ;BR IF NOT GOOD NUMBER
002540  000137  002034        3$:     BIC     #177003,R0      ,MASK OFF OTHER BITS
002544  042700  177003                MOV     R0,TKVEC0       ;SAVE THE VECTOR
002550  010037  001174                MOV     R0,TKVEC1       ;SAVE THE BR LEVEL
002554  010037  001176                ADD     #2,TKVEC1       ; ADDRESS
002560  062737  000002  001176        CLR     -(SP)
002566  005046                        MOV     #START,-(SP)    ;LOWER PS AND START PROG AGAIN
002570  012746  001326                RTI
002574  000002                4$:     CMP     (SP)+,(SP)+     ;CLEAN STACK
002576  022626                        MOV     (SP)+,@#ERRVEC  ;RESTORE LOC. 4
002600  012637  000004                CLR     -(SP)           ;LOWER PS
002604  005046                        MOV     #BOOBOO,-(SP)   ;RETURN
002606  012746  002034                RTI
002612  000002
```

```
                                                ROUTR:  CLR     GAIN            ;WANT REGULAR GAIN
002614  005037  001302                                  BIC     #BIT0,@NCCSR    ;INSURE REGULAR GAIN
002620  042777  002000  176352                          JMP     LISN            ;LOOK FOR NEXT COMMAND
002626  000137  001702                        ROUTS:    BIS     #BIT8,@NCSFR    ;STOP NCV11
002632  052777  000400  176350                          CLR     @NCCSR          ;ZERO NCV CSR
002640  005077  176334                                  JMP     LISN            ;LOOK FOR NEXT COMMAND
002644  000137  001702                        ROUTT:    JSR     PC,NCSTP        ;GO STOP THE NCV11 IF RUNNING
002650  004737  004744                                  JSR     PC,LDIMGE       ;GO SET UP TEST CORE IMAGE
002654  004737  005152                                  JMP     CHANGE          ;GO DISPLAY IT
002660  000137  003122                        ROUTU:    MOV     #340,-(SP)
002664  012746  000340                                  MOV     #1$,-(SP)
002670  012746  002676                                  RTI
002674  000002                                1$:       TYPE    ,MSG2           ;ASK FOR OCTAL DATA
002676  104401  010161                                  RDOCT                   ;GO GET IT
002702  104411                                          MOV     (SP)+,THHI      ;SAVE IT
002704  012637  001274                                  CLR     -(SP)
002710  005046                                          MOV     #2$,-(SP)
002712  012746  002720                                  RTI
002716  000002                                2$:       JMP     CHANGE          ;GO DISPLAY
002720  000137  003122                        ROUTW:    TST     NOVSV           ;TEST IF VSV01 DETECTED
002724  005737  001324                                  BNE     1$              ;BR IF NOT
002730  001012                                          CLR     MSELCT          ;SELECT 1ST BIT MAP
002732  005037  001320                                  JSR     PC,SELCTA       ;CHANGE ADDRESSES
002736  004737  004546                                  NOP
002742  000240                                          MOV     VTVCSR,R0       ;GET 1ST BIT MAP ADRS
002744  013700  001226                                  BIC     #BIT8,20(R0)    ;TURN OFF 2ND BIT MAP
002750  042760  000400  000020                1$:       JMP     LISN            ;GO AWAIT NEXT COMMAND
002756  000137  001702                        ROUTX:    MOV     #340,-(SP)
002762  012746  000340                                  MOV     #1$,-(SP)
002766  012746  002774                                  RTI
002772  000002                                1$:       TYPE,   MSG7            ;TELL OPERATOR TO SELECT CAMERA
002774  104401  010520                                  RDOCT                   ;WAIT FOR HIS INPUT
003000  104411                                          MOV     (SP)+,10$       ;GET CHAR.
003002  012637  003076                                  NOP
003006  000240                                          NOP
003010  000240                                          NOP
003012  000240                                          BIC     #177774,10$     ;MASK OFF BITS
003014  042737  177774  003076                        INC     10$
003022  005237  003076                                CLR     11$             ;CLEAR ACTUAL VALUE
003026  005037  003100                        2$:       DEC     10$
003032  005337  003076                                BEQ     3$
003036  001403                                        INC     11$             ;UPDATE ACUTAL
003040  005237  003100                                BR      2$
003044  000772                                3$:       MOVB    11$,CAMERA+1    ;UPDATE CAMERA SAVE LOC.
003046  11_737  003100  001301                        BIS     CAMERA,@NCCSR   ;AND SELECT THAT CAMERA
003054  053777  001300  176116                        CLR     -(SP)
003062  005046                                        MOV     #4$,-(SP)
003064  012746  003072                                RTI
003070  000002                                4$:       JMP     CHANGE          ;RETURN
003072  000137  003122                        10$:      0
003076  000000                                11$:      0
003100  000000                                ROUTZ:    MOV     #BIT10,GAIN     ;ENABLE GAIN
003102  012737  002000  001302                        BIS     GAIN,@NCCSR     ;SET IT AT NCV CSR
003110  053777  001302  176062                        JMP     LISN            ;GO AWAIT NEXT COMMAND
003116  000137  001702
```

```
                                        .SBTTL  PROGRAM ROUTINES
                                ;GO CLEAR SCREEN OF SELECTED DISPLAY

003122  004737  005020        CHANGE: JSR     PC,ERASE        ;GO ERASE SCOPE

                                ;STOP NCV11 AND GET SET TO DISPLAY

003126  004737  004744                JSR     PC,NCSTP        ;GO STOP NCV11

003132  005737  001324                TST     NOVSV           ;TEST IF VSV01 DETECTED
003136  001402                        BEQ     BOX             ;BR IF YES
003140  000137  001702                JMP     LISN            ;RETURN IF NOT
                                ;NOW DRAW A BOX AROUND POTENTIAL MATRIX DISPLAY
                                ;AREA ON THE SELECTED DISPLAY
003144  012700  001750        BOX:    MOV     #1750,R0        ;SET UP BIT MAP ADRS - TOP LINE
003150  012701  073567                MOV     #73567,R1       ;SET UP PIXEL DATA IN R1 - INT 7
003154  004737  005060                JSR     PC,DISPY        ;GO LOAD BIT MAP
003160  005200                1$:     INC     R0              ;ADVANCE ADRS
003162  022700  001770                CMP     #1770,R0        ;TOP LINE DONE?
003166  001403                        BEQ     2$              ;BR IF SO
003170  004737  005072                JSR     PC,DCONT        ;LOAD NEXT PIXEL WD
003174  000771                        BR      1$              ;NEXT MAP LOAD
003176  012700  006010        2$:     MOV     #6010,R0        ;SET UP BIT MAP ADRS - BOT LINE
003202  004737  005060                JSR     PC,DISPY        ;GO LOAD BIT MAP
003206  005200                3$:     INC     R0              ;ADVANCE ADRS
003210  022700  006030                CMP     #6030,R0        ;BOT LINE DONE?
003214  001403                        BEQ     4$              ;BR IF SO
003216  004737  005072                JSR     PC,DCONT        ;LOAD NEXT PIXEL WD
003222  000771                        BR      3$              ;NEXT MAP LOAD
003224  012700  001730        4$:     MOV     #1730,R0        ;SET UP BIT MAP ADRS SIDE LINES
003230  062700  000017        5$:     ADD     #17,R0          ;OFFSET NEXT ROW
003234  012701  070000                MOV     #70000,R1       ;SET UP PIXEL 3 DATA IN R1 - INT 7
003240  022700  006047                CMP     #6047,R0        ;AT BOTTOM OF SCREEN?
003244  001411                        BEQ     6$              ;GO START VSV01 DISPLAY
003246  004737  005060                JSR     PC,DISPY        ;GO LOAD BIT MAP
003252  012701  000007                MOV     #7,R1           ;SET UP PIXEL 0 DATA IN R1 - INT 7
003256  062700  000021                ADD     #21,R0          ;OFFSET TO RIGHT LINE
003262  004737  005060                JSR     PC,DISPY        ;GO LOAD BIT MAP
003266  000760                        BR      5$              ;DO NEXT ROW
003270  013777  001254  175730  6$:   MOV     VTVSAV,@VTVCSR  ;START DISPLAY
003276  012737  002010  001260        MOV     #2010,MAPADR    ;OFFSET BIT MAP ADRS
003304  012737  000003  001262        MOV     #3,PIXCNT       ;SET UP PIXEL BYTE COUNT
003312  005037  001264                CLR     PIXASM          ;CLR PIXEL ASSEMBLY WORD

                                ;NOW LETS FIND THE LARGEST CEL
003316  013737  001304  001306  LARGES: MOV   TOTSIZ,CHARCT   ;NUMBER OF ELEMENTS TO CONSIDER
003324  017700  175764                MOV     @TABLEX,R0      ;THIS IS FIRST GUESS
003330  013701  001314                MOV     TABLEX,R1       ;R1 POINTS TO TABLE
003334  020021                1$:     CMP     R0,(R1)+        ;COMPARE GUESS AGAINST NEW
003336  103005                        BHIS    3$              ;GUESS STILL GOOD
003340  024137  001274                CMP     -(R1),THHI      ;GUESS SMALLER BUT CHECK NEW
003344  101001                        BHI     2$              ;HI AGAINST UPPER THESHOLD
003346  011100                        MOV     (R1),R0         ;WITHIN BOUNDS. MAKE THIS NEW HI
003350  005721                2$:     TST     (R1)+           ;INCREASE REGISTER BY 2
003352  005337  001306        3$:     DEC     CHARCT          ;COUNT THIS LAST COMPARISON
003356  001366                        BNE     1$
```

```
                                        ;THE LARGEST CELL WITHIN THE UPPER THRESHOLD IS NOW IN R0
                                        ;NOW ACCOUNT FOR LOWER THRESHOLD - BOW OUT IF TOO SMALL
003360  163700  001272                  SUB     THLO,R0         ;SUBTRACT LOWER THRESHOLD
003364  101002                          BHI     4$              ;BR IF CELL VALUE GREATER THAN LO THRESHOLD
003366  000137  003720                  JMP     PATWRT          ;DON'T DISPLAY-ALL VALUES BELOW LO THRESHOLD
003372  010037  001310          4$:     MOV     R0,CPERL        ;SAVE LARGEST CELL OF MATRIX

                                        ;NOW PICK OUT EACH VALUE IN CORE MATRIX AND SCALE TO THE
                                        ;PROPER INTENSITY LEVEL.  THEN DISPLAY IT ON THE SELECTED DISPLAY

003376  013737  001314  001256  DMATRX: MOV     TABLEX,MRXADR   ;BEGIN AT TOP LEFT ROW
003404  062737  017600  001256          ADD     #8064.,MRXADR   ;OFFSET TO BOTTOM OF CORE MATRIX
003412  012737  000100  001312          MOV     #64.,ROWCNT     ;THERE ARE 64 CELLS PER ROW
003420  017702  175632          DISLOP: MOV     @MRXADR,R2      ;GET A CELL VALUE FROM MATRIX
003424  062737  000002  001256          ADD     #2,MRXADR       ;BUMP MATRIX ADRS
003432  005337  001312                  DEC     ROWCNT          ;COUNT CELL THIS ROW
003436  001006                          BNE     1$              ;BR IF ROW NOT FINISHED
003440  012737  000100  001312          MOV     #64.,ROWCNT     ;RESET NEXT ROW COUNT
003446  162737  000400  001256          SUB     #256.,MRXADR    ;SET UP FOR NEXT ROW IN MATRIX
003454  020237  001274          1$:     CMP     R2,THHI         ;CELL WITHIN HI THRESHOLD?
003460  101003                          BHI     2$              ;BR IF NOT
003462  163702  001272                  SUB     THLO,R2         ;SUB LOW THRESHOLD
003466  101002                          BHI     SCLCEL          ;BR IF CELL ABOVE LOW THRESHOLD
003470  005004                  2$:     CLR     R4              ;SET CELL TO LOWEST INTENSITY
003472  000423                          BR      MAPLD
003474  013701  001310          SCLCEL: MOV     CPERL,R1        ;NOW ESTABLISH THE INTENSITY LEVEL
003500  012703  000005                  MOV     #5,R3           ;SCALE TO 16 LEVELS
003504  005004                          CLR     R4
003506  006304                  2$:     ASL     R4              ;MUL BY 2
003510  020201                          CMP     R2,R1           ;COMPARE THIS CELL TO LARGEST
003512  103404                          BLO     4$              ;BR IF SMALLER
003514  005701                          TST     R1              ;CHECK CASE WHERE 0 DEVISOR
003516  001401                          BEQ     3$              ;BR IF SO
003520  005204                          INC     R4              ;ACCOUNT FOR GOOD SUBTRACTION
003522  160102                  3$:     SUB     R1,R2           ;NO DO THE SUBTRACTION
003524  000241                  4$:     CLC
003526  006001                          ROR     R1              ;MAKE DEVISOR SMALLER
003530  005303                          DEC     R3              ;COUNT POSITION
003532  001365                          BNE     2$              ;AGAIN IF NOT SCALLED TO 16 LEVELS YET
003534  160102                          SUB     R1,R2           ;ACCOUNT FOR REMAINDER
003536  101401                          BLOS    MAPLD           ;BR IF TOO SMALL
003540  005204                          INC     R4              ;ROUND UP
```

```
                                        ;THIS CODE DISPLAYS EACH SCALED CELL ON THE VSV01 (ONE OF 16 LEVELS)

003542  013700  001260          MAPLD:  MOV     MAPADR,R0       ;SET UP BIT MAP ADRS
003546  005704                          TST     R4              ;LOOK FOR NO INTENSITY
003550  001401                          BEQ     1$              ;BR IF NO INTENSITY
003552  005304                          DEC     R4              ;OFFSET TO 0-17
003554  000304                  1$:     SWAB    R4              ;PREPARE FOR PIXEL LOC
003556  006304                          ASL     R4              ;MOVE TO TOP 4 BITS
003560  006304                          ASL     R4              ;
003562  006304                          ASL     R4              ;
003564  006304                          ASL     R4              ;
003566  000241                          CLC                     ;NOW ASSEMBLE THIS PIXEL INTO PIXEL WORD
003570  006037  001264                  ROR     PIXASM          ;NOW MAKE ROOM IN PIXEL WORD
003574  006037  001264                  ROR     PIXASM          ;
003600  006037  001264                  ROR     PIXASM          ;
003604  006037  001264                  ROR     PIXASM          ;
003610  060437  001264                  ADD     R4,PIXASM       ;ADD THIS PIXEL TO OTHERS
003614  005737  001262                  TST     PIXCNT          ;ALL 4 PIXELS DONE FOR THIS WORD?
003620  001004                          BNE     2$              ;BR IF NOT
003622  013701  001264                  MOV     PIXASM,R1       ;LD PIXEL WORD INTO R1
003626  004737  005060                  JSR     PC,DISPY        ;LOAD BIT MAP
003632  005337  001262          2$:     DEC     PIXCNT          ;COUNT PIXEL
003636  100270                          BPL     DISLOP          ;BR IF 4 PIXELS NOT ASSEMBLED YET
003640  005037  001264                  CLR     PIXASM          ;CLR PIXEL ASSEMBLY WORD
003644  012737  000003  001262          MOV     #3,PIXCNT       ;RESET PIXEL COUNT
003652  005237  001260                  INC     MAPADR          ;ADVANCE MAP ADRS
003656  013700  001260                  MOV     MAPADR,R0       ;LOAD INTO R0
003662  022700  005770                  CMP     #5770,R0        ;HAVE ALL 64 ROWS BEEN DONE?
003666  001002                          BNE     3$              ;BR IF NOT
003670  000137  003720                  JMP     PATWRT          ;NOW GO DISPLAY PARAMETERS
003674  042700  177740          3$:     BIC     #177740,R0      ;SAVE ROW POSITION BITS
003700  022700  000030                  CMP     #30,R0          ;NOW LOOK FOR END OF ROW
003704  001003                          BNE     4$              ;BR IF NOT AT END
003706  062737  000020  001260          ADD     #20,MAPADR      ;ADVANCE MAP ADRS TO NEXT ROW
003714  000137  003420          4$:     JMP     DISLOP          ;GO GET NEXT MATRIX DATUM
```

```
                                        ;CODE TO WRITE PARAMETER DATA AT BOTTOM OF SCREEN
                                        
003720  012777  012000  175276  PATWRT: MOV     #12000,@VTVPOS  ;VSV01 - SET CHAR POS,LINE 20, LEFT MARGIN
003726  004537  005316                  JSR     R5,VTWRIT
003732  007664                          WD001                   ;''CR,LOWER THRESHOLD''
003734  004537  005342                  JSR     R5,AWRIT        ;DISPAY OCTAL
003740  001272                          THLO
003742  004537  005316                  JSR     R5,VTWRIT
003746  007707                          WD002                   ;''CR, UPPER THRESHOLD''
003750  004537  005342                  JSR     R5,AWRIT        ;DISPAY OCTAL
003754  001274                          THHI
003756  113737  001301  001246          MOVB    CAMERA+1,DUMMY1 ;GET CAMERA VALUE
003764  062737  000060  001246          ADD     #60,DUMMY1      ;MAKE ASCII
003772  113737  001246  010022          MOVB    DUMMY1,CAMOUT   ;SAVE FOR READOUT
                                        ;TEST THE ''PB'' FLAG
004000  032777  000040  175202          BIT     #BIT5,@NCSFR    ;TEST FOR ''PB'' FLAG
004006  001007                          BNE     1$              ;BR IF SET
004010  112737  000125  010036          MOVB    #'U,PBUP        ;NO- TELL OPER. IT WAS UP
004016  112737  000120  010037          MOVB    #'P,PBUP+1
004024  000406                          BR      2$
004026  112737  000104  010036  1$:     MOVB    #'D,PBUP        ;TFLL OPER. IT WAS DN
004034  112737  000116  010037          MOVB    #'N,PBUP+1
                                        ;TEST FOR JOY-STICK BUTTON FLAG
004042  032777  000100  175140  2$:     BIT     #BIT6,@NCSFR    ;TEST FOR ''JOY-BUTTON'' FLAG
004050  001007                          BNE     3$              ;BR IF SET
004052  112737  000125  010053          MOVB    #'U,JBUP        ;TELL OPER. IT WAS UP
004060  112737  000120  010054          MOVB    #'P,JBUP+1
004066  000406                          BR      4$
004070  112737  000104  010053  3$:     MOVB    #'D,JBUP        ;TELL OPER. IT WAS DN
004076  112737  000116  010054          MOVB    #'N,JBUP+1
004104  004537  005316          4$:     JSR     R5,VTWRIT       ;DISPLAY CAMERA VALUE AND BUTTON STATUS
004110  010007                          WD007
004112  004537  005316                  JSR     R5,VTWRIT
004116  007735                          WD003                   ;''Z COUNT =''
004120  017737  175062  001246          MOV     @NCBAR,DUMMY1
004126  017737  175052  001250          MOV     @NCWCR,DUMMY2
004134  004537  005342                  JSR     R5,AWRIT        ;DISPALY OCTAL
004140  001250                          DUMMY2
004142  004537  005316                  JSR     R5,VTWRIT       ;INSERT ''-''
004146  010634                          DASH
004150  004537  005342                  JSR     R5,AWRIT        ;DISPLAY OCTAL
004154  001246                          DUMMY1
004156  004537  005316                  JSR     R5,VTWRIT
004162  007750                          WD004                   ;''MATRIX COUNT-''
```

```
                                    ;DETERMINE THE # OF COUNTS IN THE MATRIX
                                    
004164  013737  001304  001240          MOV     TOTSIZ,TEMP0
004172  005002                          CLR     R2
004174  005003                          CLR     R3
004176  013701  001314                  MOV     TABLEX,R1
004202  062103                  MXSML:  ADD     (R1)+,R3        ;NOW ADD UP ALL VALUES IN MATRIX
004204  005502                          ADC     R2
004206  005337  001240                  DEC     TEMP0
004212  001373                          BNE     MXSML
004214  010337  001246                  MOV     R3,DUMMY1
004220  010237  001250                  MOV     R2,DUMMY2
004224  004537  005342                  JSR     R5,AWRIT        ;TELL OPER. OCTAL
004230  001250                          DUMMY2
004232  004537  005316                  JSR     R5,VTWRIT       ;INSERT DASH
004236  010634                          DASH
004240  004537  005342                  JSR     R5,AWRIT
004244  001246                          DUMMY1
004246  004537  005316                  JSR     R5,VTWRIT
004252  011716                          BCRLF
                                    
                                    ;TEST FOR GAIN = 2 AND B GAMMA STATES
                                    
004254  005737  001302                  TST     GAIN
004260  001403                          BEQ     1$
004262  004537  005316                  JSR     R5,VTWRIT
004266  007767                          WD005                   ;''ZOOM''
004270  023727  001314  040000  1$:     CMP     TABLEX,#MATRIX+20000 ;ISOTOPE B?
004276  001003                          BNE     2$                      ;NO
004300  004537  005316                  JSR     R5,VTWRIT               ;YES
004304  007776                          WD006                   ;''B-GAMMA''
004306  005737  001316          2$:     TST     FREERN          ;IN FREE RUN MODE?
004312  001402                          BEQ     3$              ;BR IF NOT
004314  000137  002124                  JMP     ROUTF           ;YES, GO GET NEW DATA
004320  013777  001276  174652  3$:     MOV     COMSAV,@NCCSR   ;RESUME THE NCV11
004326  000137  001702                  JMP     LISN            ;DONE WITH MESSAGES
```

```
                                        ;THIS CODE DRAWS CROSS HAIRS(VSV01) WITH THE X & Y JOYSTICK DATA
                                        ;THE BUG WILL FOLLOW THE JOYSTICK WHEN THE INTERRUPT BAR IS NOT DEPRESSED -
                                        ;ALL PARTS WITHIN THE DISPLAYED BOX ON THE SELECTED DISPLAY SHOULD
                                        ;BE ACCESSIBLE IN A UNIFORM MANNER
                                        ;A CNTRL 'C' WILL GET USER BACK TO KEYBOARD MONITOR
004332  004737  005020        TJOY:   JSR     PC,ERASE        ;START FRESH

                                        ;DRAW A BOX UP ON VSV01 SCREEN FOR NCV11 JOYSTICK CALIBRATION
004336  005000                        CLR     R0              ;SET UP BIT MAP ADRS - TOP LINE
004340  012701  073567                MOV     #73567,R1       ;SET UP PIXEL DATA IN R1 - INT 7
004344  004737  005060                JSR     PC,DISPY        ;GO LOAD BIT MAP
004350  005200                1$:     INC     R0              ;ADVANCE BIT MAP ADRS
004352  022700  000040                CMP     #40,R0          ;TOP LINE DONE?
004356  001403                        BEQ     2$              ;BR IF SO
004360  004737  005072                JSR     PC,DCONT        ;LOAD NEXT PIXEL
004364  000771                        BR      1$              ;NEXT MAP LOAD
004366  012700  007740        2$:     MOV     #7740,R0        ;SET UP BIT MAP ADRS - BOT LINE
004372  004737  005060                JSR     PC,DISPY        ;GO LOAD BIT MAP
004376  005200                3$:     INC     R0              ;ADVANCE ADRS
004400  022700  010000                CMP     #10000,R0       ;BOT LINE DONE?
004404  001403                        BEQ     4$              ;BR IF SO
004406  004737  005072                JSR     PC,DCONT        ;LOAD NEXT PIXEL WORD
004412  000771                        BR      3$              ;NEXT MAP LOAD
004414  012700  000040        4$:     MOV     #40,R0          ;SET UP BIT MAP ADRS SIDE LINES
004420  012701  000007        5$:     MOV     #7,R1           ;SET UP PIXEL 0 DATA IN R1 - INT 7
004424  004737  005060                JSR     PC,DISPY        ;GO LOAD BIT MAP
004430  012701  070000                MOV     #70000,R1       ;SET UP PIXEL 3 DATA IN R1 - INT 7
004434  062700  000037                ADD     #37,R0          ;OFFSET TO RIGHT SIDE LINE
004440  004737  005060                JSR     PC,DISPY        ;GO LOAD BIT MAP
004444  005200                        INC     R0              ;GO TO NEXT ROW ON LEFT
004446  022700  007740                CMP     #7740,R0        ;TO BOTTOM YET?
004452  001362                        BNE     5$              ;BR IF NOT
004454  013777  001254  174544        MOV     VTVSAV,@VTVCSR  ;ENABLE BIT MAP

                                        ;NOW COLLECT ANALOG DATA FROM NCV11 JOYSTICK

004462  012777  004000  174520 DISBG: MOV     #BIT11,@NCSFR   ;CLEAR THE DEVICE
004470  052777  000001  174512 1$:    BIS     #BIT0,@NCSFR    ;CONVERT JOYSTICK
004476  105777  174506         2$:    TSTB    @NCSFR          ;DONE?
004502  100375                        BPL     2$              ;BR IF NOT
004504  017737  174504  001240        MOV     @NCJOY,TEMP0    ;STORE CONVERSION VALUES
004512  000240                        NOP
004514  000240                        NOP
004516  000240                        NOP
004520  004737  005472                JSR     PC,XYAVE        ;GO AVG LAST 32. X-Y JOYSTICK VALUES
004524  032777  000040  174456        BIT     #BIT5,@NCSFR    ;LOOK FOR JOY BOTTON DOWN
004532  001356                        BNE     1$              ;DON'T MOVE THE BUG
004534  000240                        NOP
004536  000240                        NOP
004540  004737  005122                JSR     PC,DISPY2       ;GO DISPLAY CROSS HAIRS - VSV01
004544  000751                        BR      1$              ;DO ANOTHER CONVERSION
```

```
                        ;;*****************************************************************
                        ;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
                        ;;*****************************************************************
004546  013700  001166  SELCTA: MOV     VTVADR,R0       ;GET VSV01 BASE ADRS
004552  012701  001220          MOV     #VTVCRG,R1      ;GET PTR ADRS
004556  010021          1$:     MOV     R0,(R1)+        ;SET UP REG ADRS PTRS
004560  062700  000002          ADD     #2,R0           ;BUMP REG ADRS
004564  022701  001226          CMP     #VTVCSR,R1      ;CHAR GEN REGS ALL SET UP?
004570  001372                  BNE     1$              ;BR IF NOT
004572  013700  001170          MOV     VTMADR,R0       ;GET BASE ADRS OF BIT MAP
004576  005737  001320          TST     MSELCT          ;USING SECOND MAP?
004602  001402                  BEQ     2$              ;BR IF NOT
004604  062700  000020          ADD     20,R0           ;POINT TO 2ND BIT MAP ADRS'S
004610  010021          2$:     MOV     R0,(R1)+        ;CONTINUE TO BIT MAP ADRS'S
004612  062700  000002          ADD     #2,R0           ;BUMP REG ADRS
004616  022701  001240          CMP     #VTVINT+2,R1    ;ALL SET UP?
004622  001372                  BNE     2$              ;BR IF NOT
004624  012737  004676  000004  MOV     #5$,@#ERRVEC    ;SET UP BUS TIMEOUT RETURN ADRS IF NO VSV01
004632  005777  174362          TST     @VTVCRG         ;IS CHARACTER GENERATOR THERE?
004636  005777  174364          TST     @VTVCSR         ;IS BIT MAP THERE?
004642  013700  001252          MOV     INTLUT,R0       ;SET UP ADRS & DATA OF INTENSITY LOOK-UP TABLE
004646  010077  174364          3$:     MOV     R0,@VTVINT      ;SET UP TABLE
004652  062700  000401          ADD     #401,R0         ;ADVANCE ADRS & INTENSITY
004656  032700  010000          BIT     #10000,R0       ;TABLE LOADED?
004662  001771                  BEQ     3$              ;BR IF NOT
004664  000241                  CLC                     ;ZERO CARRY SAYS VSV01 THERE
004666  012737  000006  000004  4$:     MOV     #ERRVEC+2,@#ERRVEC      ;RESTORE ER TRAP LOC TO PT TO 6
004674  000207                  RTS     PC              ;EXIT
004676  022626          5$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
004700  000261                  SEC                     ;CARRY ON EXIT SAYS NO VSV01
004702  000771                  BR      4$              ;GO EXIT

                        ;;*****************************************************************
                        ;ROUTINE STARTS NCV11 AT SELECTED GAIN AND CAMERA
                        ;;*****************************************************************
004704  012777  004030  174266  NCSTRT: MOV     #4030,@NCCSR    ;SET UP ZB ENABLE AND 64*64 WORD MATRIX
004712  012777  020000  174262          MOV     #MATRIX,@NCOFF  ;ENSURE OFFSET REG. IS SET
004720  053777  001302  174252          BIS     GAIN,@NCCSR     ;SET UP GAIN
004726  053777  001300  174244          BIS     CAMERA,@NCCSR   ;SET UP CAMERA
004734  052777  000001  174236          BIS     #BIT0,@NCCSR    ;SET GO BIT
004742  000207                  RTS     PC              ;RETURN

                        ;;*****************************************************************
                        ;ROUTINE STOPS NCV11 AND SAVES NCV11 STATUS
                        ;;*****************************************************************
004744  017737  174230  001276  NCSTP:  MOV     @NCCSR,COMSAV   ;SAVE THE INTERFACE ACTION
004752  052777  000400  174230          BIS     #BIT8,@NCSFR    ;DISABLE NPR'S
004760  005077  174214          CLR     @NCCSR          ;ZERO ALL STATUS
004764  000207                  RTS     PC              ;RETURN
```

```
                                        ;;*****************************************************************
                                        ;ROUTINE STOPS NCV11, SAVES STATUS AND CLEARS MATRIX CORE AREA
                                        ;;*****************************************************************
004766  004737  004744          NCSTP1: JSR     PC,NCSTP        ;GO STOP NCV11 AND SAVE STATUS
004772  005077  174206                  CLR     @NCWCR          ;CLEAR HIGH WORD
004776  005077  174204                  CLR     @NCBAR          ;CLEAR LOW WORD
005002  012700  020000                  MOV     #MATRIX,R0      ;GET SET TO ZERO CORE MATRIX AREA
005006  005020                  1$:     CLR     (R0)+           ;ZERO LOC
005010  020027  060000                  CMP     R0,#MATRIX+40000        ;ALL DONE?
005014  001374                          BNE     1$              ;BR IF MORE
005016  000207                          RTS     PC              ;RETURN

                                        ;;*****************************************************************
                                        ;ROUTINE WILL ERASE DISPLAY
                                        ;;*****************************************************************
005020  005737  001324          ERASE:  TST     NOVSV           ;TEST IF VSV01 DETECTED
005024  001014                          BNE     2$              ;BR IF NOT
005026  052777  001000  174172          BIS     #1000,@VTVCSR   ;ERASE DISPLAY (CLR BIT MAP)
005034  105777  174166          1$:     TSTB    @VTVCSR         ;LOOK FOR READY
005040  100375                          BPL     1$              ;WAIT FOR IT
005042  042777  001000  174156          BIC     #1000,@VTVCSR   ;TURN OFF ERASE DISPLAY
005050  012777  002035  174142          MOV     #2035,@VTVCRG   ;CLR CHAR SCREEN & DISABLE CURSOR
005056  000207                  2$:     RTS     PC              ;EXIT

                                        ;;*****************************************************************
                                        ;ROUTINE WILL LOAD BIT MAP (VSV01)
                                        ;R0 CONTAINS BIT MAP ADRS AND R1 THE BIT MAP DATA
                                        ;;*****************************************************************
005060  042777  000400  174140  DISPY:  BIC     #400,@VTVCSR    ;STOP DISPLAY
005066  010077  174136                  MOV     R0,@VTVMAP      ;LOAD BIT MAP ADRS
005072  042777  000400  174126  DCONT:  BIC     #400,@VTVCSR    ;AGAIN IF ENTERING HERE
005100  105777  174122          1$:     TSTB    @VTVCSR         ;READY?
005104  100375                          BPL     1$              ;WAIT THEN
005106  010177  174120                  MOV     R1,@VTVPX       ;LOAD BIT MAP
005112  052777  000400  174106          BIS     #400,@VTVCSR    ;RESUME DISPLAY
005120  000207                          RTS     PC              ;RETURN


                                        ;;*****************************************************************
                                        ;ROUTINE WILL DISPLAY X & Y CROSS HAIRS ON VSV01
                                        ;;*****************************************************************
005122  005777  174072          DISPY2: TST     @VTVCRG         ;READY?
005126  100375                          BPL     DISPY2          ;WAIT IF NOT
005130  105137  001241                  COMB    TEMP0+1         ;X NEEDS TO BE INVERTED
005134  013777  001240  174060          MOV     TEMP0,@VTVCHP   ;LOAD X & Y CROSS HAIRS
005142  052777  016000  174050          BIS     #16000,@VTVCRG  ;ENABLE THE CROSS HAIRS
005150  000207                          RTS     PC              ;RETURN
```

```
                                        ;;*****************************************************************
                                        ;ROUTINE WIIL FILL CORE WITH AN IMAGE THAT WHEN
                                        ;DISPLAYED WILL CONTAIN ALL THE INTENSITY LEVELS -
                                        ;ROWS AT THE BOTTOM OF THE SCREEN WILL APPEAR BRIGHTEST -
                                        ;NOTE THAT THIS IS ONLY A DISPLAY TEST PATTERN FOR
                                        ;POSSIBLE DISPLAY ADJUSTMENTS BY THE USER
                                        ;;*****************************************************************
005152  012700  000100        LDIMGE: MOV     #100,R0         ;COUNT 64 ROWS
005156  013701  001314                MOV     TABLEX,R1       ;GET SELECTED ISOTOPE
005162  012702  000100                MOV     #100,R2         ;COUNT 64 DATA POINTS PER ROW
005166  012703  000100                MOV     #100,R3         ;100 WILL REPRESENT HIGHEST INTENSITY LEVEL(100-0)
005172  010321                1$:     MOV     R3,(R1)+        ;LOAD CORE IMAGE
005174  005302                        DEC     R2              ;DONE ROW?
005176  001375                        BNE     1$              ;BR IF NOT
005200  005300                        DEC     R0              ;DONE ROWS?
005202  001405                        BEQ     2$              ;BR IF SO
005204  162703  000001                SUB     #1,R3           ;LOWER NEXT CELL VALUE
005210  012702  000100                MOV     #100,R2         ;RESET ROW LENGTH COUNTER
005214  000766                        BR      1$              ;LOAD THIS ROW IMAGE
005216  000207                2$:     RTS     PC              ;RETURN FOR DISPLAY

                                        ;;*****************************************************************
                                        ;KEYBOARD INTERRUPT SERVICE ROUTINE
                                        ;;*****************************************************************
005220  017737  173722  001266  KBINT:  MOV     @$TKB,KBUFF     ;READ KEY BOARD
005226  013777  001266  173716        MOV     KBUFF,@$TPB     ;ECHO CHAR.
005234  042737  177600  001266        BIC     #177600,KBUFF   ;RID PARITY
005242  022737  000003  001266        CMP     #3,KBUFF        ;CNTRL 'C'?
005250  001002                        BNE     1$              ;BR IF NOT
005252  000137  001656                JMP     LISEN           ;ABORT WHATEVER & LOOK FOR NEXT COMMAND
005256  000002                1$:     RTI

                                        ;;*****************************************************************
                                        ;ROUTINE TYPES 'CR' AND 'LF' OR CHAR IN TTYOUT
                                        ;;*****************************************************************
005260  012737  000015  001270  TYPCR:  MOV     #15,TTYOUT      ;SET UP FOR A 'CR'
005266  004737  005300                JSR     PC,TYPO
005272  012737  000012  001270        MOV     #12,TTYOUT      ;SET UP FOR LF
005300  105777  173644        TYPO:   TSTB    @$TPS           ;WAIT FOR LAST CHARACTER
005304  100375                        BPL     TYPO
005306  013777  001270  173636        MOV     TTYOUT,@$TPB    ;SEND IT OUT
005314  000207                        RTS     PC
```

```
                                        ;;*****************************************************************
                                        ;THIS SUBROUTINE IS CALLED WITH THE ADDRESS OF A MESSAGE TERM WITH A 0 BYTE
                                        ;;*****************************************************************

005316  012504                  VTWRIT: MOV     (R5)+,R4                ;GET ADDRESS OF MESSAGE IN R4
005320  005777  173674            1$:     TST     @VTVCRG         ;WAIT FOR READY
005324  100375                          BPL     1$
005326  105714                          TSTB    (R4)            ;TERM ?
005330  001403                          BEQ     2$              ;BR IF YES
005332  112477  173662                  MOVB    (R4)+,@VTVCRG   ;LOAD THE CHARACTER
005336  000770                          BR      1$
005340  000205                  2$:     RTS     R5              ;EXIT

                                        ;;*****************************************************************
                                        ;THIS ROUTINE CONVERTS OCTAL INTO ASCII FOR DISPLAY
                                        ;;*****************************************************************
005342  012537  005454          AWRIT:  MOV     (R5)+,10$       ;GET VALUE'S ADDRESS
005346  017737  000102  005454          MOV     @10$,10$        ;GET ACTUAL VALUE
005354  010046                          MOV     R0,-(SP)        ;SAVE R0
005356  012700  005470                  MOV     #NUMEND,R0      ;LOAD LAST ADDRESS OF OCTAL TYPEOUT
005362  012737  000006  005460          MOV     #6,12$          ;LOAD LOOP COUNTER
005370  000406                          BR      2$
005372  006237  005454          1$:     ASR     10$             ;SHIFT DATA
005376  006237  005454                  ASR     10$
005402  006237  005454                  ASR     10$
005406  013737  005454  005456  2$:     MOV     10$,11$         ;COPY THE VALUE
005414  042737  177770  005456          BIC     #177770,11$     ;MASK OFF UNWANTED BITS
005422  062737  000060  005456          ADD     #60,11$         ;MAKE ASCII OCTAL
005430  113740  005456                  MOVB    11$,-(R0)       ;SAVE THE CHAR.
005434  005337  005460                  DEC     12$             ;FINISHED ?
005440  001354                          BNE     1$
005442  012600                          MOV     (SP)+,R0
005444  004537  005316                  JSR     R5,VTWRIT       ;DISPAY THE OCTAL #
005450  005462                          NUMBEG
005452  000205                          RTS     R5              ;EXIT
005454  000000                  10$:    0
005456  000000                  11$:    0
005460  000005                  12$:    5
005462    060     060     060   NUMBEG: .BYTE   60,60,60,60,60,60
005465    060     060     060
005470    000                   NUMEND. .BYTE   0
        005472                          .EVEN
```

```
                        ;;*****************************************************************
                        ;THIS ROUTINE AVERAGES THE LAST 32. X-Y JOYSTICK VALUES
                        ;;*****************************************************************
005472  010046         XYAVE:  MOV     R0,-(SP)        ;SAVE R0
005474  013700 001240          MOV     TEMP0,R0        ;GET X & Y
005500  013702 005720          MOV     XYBUFP,R2       ;GET CURRENT BUFFER POINTER
005504  010022                  MOV     R0,(R2)+        ;SAVE NEW X-Y VALUE
005506  020227 005720          CMP     R2,#XYBUFE      ;END OF BUFFER?
005512  103402                  BLO     1$              ;NO
005514  012702 005620          MOV     #XYBUF,R2       ;YES, GO BACK TO BEGINING OF BUFFER
005520  010237 005720      1$: MOV     R2,XYBUFP       ;SAVE NEW BUFFER POINTER
005524  012702 005620          MOV     #XYBUF,R2       ;CALC AVE X
005530  004737 005562          JSR     PC,10$
005534  110046                  MOVB    R0,-(SP)        ;SAVE IT
005536  012702 005621          MOV     #XYBUF+1,R2     ;CALC AVE Y
005542  004737 005562          JSR     PC,10$          ;GO DO IT
005546  000300                  SWAB    R0              ;GET X-Y IN R0
005550  152600                  BISB    (SP)+,R0        ;GET SAVED X
005552  010037 001240          MOV     R0,TEMP0        ;PUT IN TEMP0
005556  012600                  MOV     (SP)+,R0        ;RESTORE R0
005560  000207                  RTS     PC              ;EXIT WITH AVE X-Y IN TEMP0

005562  005000         10$:    CLR     R0              ;ZERO SUM
005564  005005         11$:    CLR     R5              ;DO A MOVB TO A REG (UNSIGNED)
005566  152205                  BISB    (R2)+,R5
005570  060500                  ADD     R5,R0           ;ADD IT IN
005572  005202                  INC     R2              ;SKIP OTHER VALUE
005574  020227 005720          CMP     R2,#XYBUFE      ;END OF BUFFER?
005600  103771                  BLO     11$             ;NO
005602  006300                  ASL     R0              ;DIVIDE BY 32.
005604  006300                  ASL     R0
005606  006300                  ASL     R0
005610  000300                  SWAB    R0
005612  042700 177400          BIC     #177400,R0      ;CLR HI BYTE
005616  000207                  RTS     PC

005620  000040         XYBUF:
005720                 XYBUFE=.
005720  005620         XYBUFP: XYBUF
```

```
                                        ;;*****************************************************************
                                        .SBTTL  TTY INPUT ROUTINE
                                        ;;*****************************************************************
                                        .ENABL  LSB

                                        ;;*****************************************************************
                                        ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                        ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                        ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                        ;*WHEN OPERATING IN TTY FLAG MODE.
005722  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
005730  001074                        BNE     15$             ;;BRANCH IF NO
005732  105777  173206                TSTB    @$TKS           ;;CHAR THERE?
005736  100071                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
005740  117746  173202                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
005744  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
005750  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
005754  001062                        BNE     15$             ;;NO, RETURN TO USER
005756  123727  001134  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
005764  001456                        BEQ     15$             ;;BRANCH IF YES

005766  104401  006574                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
005772  104401  006601        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
005776  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
006002  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
006004  104401  006612                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
006010  005046                19$:    CLR     -(SP)           ;;CLEAR COUNTER
006012  005046                        CLR     -(SP)           ;;THE NEW SWR
006014  105777  173124          7$:   TSTB    @$TKS           ;;CHAR THERE?
006020  100375                        BPL     7$              ;;IF NOT TRY AGAIN

006022  117746  173120                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
006026  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII



006032  021627  000025          9$:   CMP     (SP),#25        ;;IS IT A CONTROL-U?
006036  001005                        BNE     10$             ;;BRANCH IF NOT
006040  104401  006567                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
006044  062706  000006          20$:  ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
006050  000757                        BR      19$             ;;LET'S TRY IT AGAIN


006052  021627  000015          10$:  CMP     (SP),#15        ;;IS IT A <CR>?
006056  001022                        BNE     16$             ;;BRANCH IF NO
006060  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
006064  001403                        BEQ     11$             ;;BRANCH IF YES
006066  016677  000002  173044        MOV     2(SP),@SWR      ;;SAVE NEW SWR
006074  062706  000006          11$:  ADD     #6,SP           ;;CLEAR UP STACK
006100  104401  001161          14$:  TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
006104  123727  001135  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
006112  001003                        BNE     15$             ;;BRANCH IF NOT
006114  012777  000100  173022        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
006122  000002                15$:    RTI                     ;;RETURN
006124  004737  006774          16$:  JSR     PC,$TYPEC       ;;ECHO CHAR
```

```
(1)  006130  021627  000060                CMP     (SP),#60        ;;CHAR < 0?
(1)  006134  002420                        BLT     18$             ;;BRANCH IF YES
(1)  006136  021627  000067                CMP     (SP),#67        ;;CHAR > 7?
(1)  006142  003015                        BGT     18$             ;;BRANCH IF YES
(1)  006144  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
(1)  006150  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
(1)  006154  001403                        BEQ     17$             ;;BRANCH IF YES
(1)  006156  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
(1)  006160  006316                        ASL     (SP)            ;;   CHAR OVER TO MAKE
(1)  006162  006316                        ASL     (SP)            ;;   ROOM FOR NEW ONE.
(1)  006164  005266  000002          17$:  INC     2(SP)           ;;KEEP COUNT OF CHAR
(1)  006170  056616  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
(1)  006174  000707                        BR      7$              ;;GET THE NEXT ONE
(1)  006176  104401  001160          18$:  TYPE    ,$QUES          ;;TYPE ?<CR><LF>
(1)  006202  000720                        BR      20$             ;;SIMULATE CONTROL-U
(1)                                 .DSABL  LSB
(1)
(1)
(2)                                 ;;**************************************************************************
(1)                                 ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                 ;*CALL:
(1)                                 ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
(1)                                 ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
(1)                                 ;*                              ;;WITH PARITY BIT STRIPPED OFF
(1)                                 ;
(1)
(1)  006204  011646                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
(1)  006206  016666  000004  000002        MOV     4(SP),2(SP)     ;;SAVE THE PS
(1)  006214  105777  172724          1$:   TSTB    @$TKS           ;;WAIT FOR
(1)  006220  100375                        BPL     1$              ;;A CHARACTER
(1)  006222  117766  172720  000004        MOVB    @$TKB,4(SP)     ;;READ THE TTY
(1)  006230  042766  177600  000004        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
(1)  006236  026627  000004  000023        CMP     4(SP),#23       ;;IS IT A CONTROL-S?
(1)  006244  001013                        BNE     3$              ;;BRANCH IF NO
(1)  006246  105777  172672          2$:   TSTB    @$TKS           ;;WAIT FOR A CHARACTER
(1)  006252  100375                        BPL     2$              ;;LOOP UNTIL ITS THERE
(1)  006254  117746  172666                MOVB    @$TKB,-(SP)     ;;GET CHARACTER
(1)  006260  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
(1)  006264  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
(1)  006270  001366                        BNE     2$              ;;IF NOT DISCARD IT
(1)  006272  000750                        BR      1$              ;;YES, RESUME
(1)  006274  026627  000004  000140  3$:   CMP     4(SP),#140      ;;IS IT UPPER CASE?
(1)  006302  002407                        BLT     4$              ;;BRANCH IF YES
(1)  006304  026627  000004  000175        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
(1)  006312  003003                        BGT     4$              ;;BRANCH IF YES
(1)  006314  042766  000040  000004        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
(1)  006322  000002                4$:     RTI                     ;;GO BACK TO USER
(2)                                 ;;**************************************************************************
(1)                                 ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
(1)                                 ;*CALL:
(1)                                 ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
(1)                                 ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
(1)                                 ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
(1)
(1)  006324  010346                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
(1)  006326  005046                        CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
```

```
(1)  006330  012703  006560          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
(1)  006334  022703  006567          2$:     CMP     #$TTYIN+7,R3    ;;BUFFER FULL?
(1)  006340  101456                          BLOS    4$              ;;BR IF YES
(1)  006342  104407                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
(1)  006344  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
(1)  006346  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
(1)  006352  001022                          BNE     5$              ;;BR IF NO
(1)  006354  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
(1)  006356  001007                          BNE     6$              ;;BR IF NO
(1)  006360  112737  000134  006556          MOVB    #'\,9$          ;;TYPE A BACK SLASH
(1)  006366  104401  006556                  TYPE    ,9$
(1)  006372  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
(1)  006376  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
(1)  006400  020327  006560                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
(1)  006404  103434                          BLO     4$              ;;BR IF YES
(1)  006406  111337  006556                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
(1)  006412  104401  006556                  TYPE    ,9$             ;;GO TYPE
(1)  006416  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
(1)  006420  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
(1)  006422  001406                          BEQ     7$              ;;BR IF NO
(1)  006424  112737  000134  006556          MOVB    #'\,9$          ;;TYPE A BACK SLASH
(1)  006432  104401  006556                  TYPE    ,9$
(1)  006436  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
(1)  006440  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
(1)  006444  001003                          BNE     8$              ;;BR IF NO
(1)  006446  104401  006567                  TYPE    ,$CNTLU         ;;TYPE A CONTROL 'U'
(1)  006452  000726                          BR      1$              ;;GO START OVER
(1)  006454  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A '^R'?
(1)  006460  001011                          BNE     3$              ;;BRANCH IF NO
(1)  006462  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
(1)  006464  104401  001161                  TYPE    ,$CRLF          ;;TYPE A 'CR' & 'LF'
(1)  006470  104401  006560                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
(1)  006474  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
(1)  006476  104401  001160          4$:     TYPE    ,$QUES          ;;TYPE A '?'
(1)  006502  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
(1)  006504  111337  006556          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
(1)  006510  104401  006556                  TYPE    ,9$
(1)  006514  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
(1)  006520  001305                          BNE     2$              ;;LOOP IF NOT RETURN
(1)  006522  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
(1)  006526  104401  001162                  TYPE    ,$LF            ;;TYPE A LINE FEED
(1)  006532  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
(1)  006534  012603                          MOV     (SP)+,R3        ;;RESTORE R3
(1)  006536  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
(1)  006540  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
(1)  006546  012766  006560  000004          MOV     #$TTYIN,4(SP)
(1)  006554  000002                          RTI                     ;;RETURN
(1)  006556     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
(1)  006557     000                          .BYTE   0               ;;TERMINATOR
(1)  006560  000007                  $TTYIN: .BLKB   7               ;;RESERVE 7 BYTES FOR TTY INPUT
(1)  006567     136  006525  000012  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
(1)  006574  043536  005015     000  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
(1)  006601     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
(1)  006606  036440  000040
(1)  006612  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
(1)  006620  020075     000
```

```
(1)           006624                    .EVEN
880                                     ;;*****************************************************************
881                                     .SBTTL  TYPE ROUTINE
(1)
(2)                                     ;;*****************************************************************
(1)                                     ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(1)                                     ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(1)                                     ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(1)                                     ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(1)                                     ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(1)                                     ;*
(1)                                     ;*CALL:
(1)                                     ;*1) USING A TRAP INSTRUCTION
(1)                                     ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(1)                                     ;*OR
(1)                                     ;*      TYPE
(1)                                     ;*      MESADR
(1)                                     ;*
(1)
(1)   006624  105737  001157            $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(1)   006630  100002                            BPL     1$              ;;BR IF YES
(1)   006632  000000                            HALT                    ;;HALT HERE IF NO TERMINAL
(1)   006634  000407                            BR      3$              ;;LEAVE
(1)   006636  010046                    1$:     MOV     R0,-(SP)        ;;SAVE R0
(1)   006640  017600  000002                    MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)   006644  112046                    2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)   006646  001005                            BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)   006650  005726                            TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)   006652  012600                    60$:    MOV     (SP)+,R0        ;;RESTORF R0
(1)   006654  062716  000002            3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)   006660  000002                            RTI                     ;;RETURN
(1)   006662  122716  000011            4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)   006666  001430                            BEQ     8$
(1)   006670  122716  000200                    CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(1)   006674  001006                            BNE     5$
(1)   006676  005726                            TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)   006700  104401                            TYPE                    ;;TYPE A CR AND LF
(1)   006702  001161                            $CRLF
(1)   006704  105037  007040                    CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(1)   006710  000755                            BR      2$              ;;GET NEXT CHARACTER
(1)   006712  004737  006774            5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)   006716  123726  001156            6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)   006722  001350                            BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)   006724  013746  001154                    MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                             ;;AND THE NULL CHAR.
(1)   006730  105366  000001            7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)   006734  002770                            BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)   006736  004737  006774                    JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)   006742  105337  007040                    DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)   006746  000770                            BR      7$              ;;LOOP
(1)
(1)                                     ;HORIZONTAL TAB PROCESSOR
(1)
(1)   006750  112716  000040            8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)   006754  004737  006774            9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)   006760  132737  000007  007040            BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
```

```
(1)  006766  001372                          BNE     9$              ;;TAB STOP
(1)  006770  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
(1)  006772  000724                          BR      2$              ;;GET NEXT CHARACTER
(1)  006774  105777  172150        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  007000  100375                          BPL     $TYPEC
(1)  007002  116677  000002  172142        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  007010  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(1)  007016  001003                          BNE     1$              ;;BRANCH IF NO
(1)  007020  105037  007040                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(1)  007024  000406                          BR      $TYPEX          ;;EXIT
(1)  007026  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(1)  007034  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
(1)  007036  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
(1)  007040  000000                $CHARCNT:.WORD   0               ;;CHARACTER COUNT STORAGE
(1)  007042  000207                $TYPEX: RTS     PC
(1)
882                                ;;**************************************************************************
883                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(2)                                ;;**************************************************************************
(1)                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1)                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(1)                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1)                                ;*CALL:
(1)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
(1)                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1)                                ;*      .BYTE   M               ;;M-1 OR 0
(1)                                ;*                                      ;;1=TYPE LEADING ZEROS
(1)                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
(1)                                ;*
(1)                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1)                                ;*$TYPOS OR $TYPOC
(1)                                ;*CALL:
(1)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                ;*      TYPON                   ;;CALL FOR TYPEOUT
(1)                                ;*
(1)                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1)                                ;*CALL:
(1)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                ;*      TYPOC                   ;;CALL FOR TYPEOUT
(1)
(1)  007044  017646  000000        $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  007050  116637  000001  007267        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  007056  112637  007271                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  007062  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  007066  000406                        BR      $TYPON
(1)  007070  112737  000001  007267  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
(1)  007076  112737  000006  007271        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  007104  112737  000005  007266  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(1)  007112  010346                        MOV     R3,-(SP)        ;;SAVE R3
(1)  007114  010446                        MOV     R4,-(SP)        ;;SAVE R4
(1)  007116  010546                        MOV     R5,-(SP)        ;;SAVE R5
(1)  007120  113704  007271                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  007124  005404                        NEG     R4
(1)  007126  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
```

```
   (1)  007132  110437  007270                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
   (1)  007136  113704  007267                MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
   (1)  007142  016605  000012                MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
   (1)  007146  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
   (1)  007150  006105                1$:     ROL     R5              ;;ROTATE MSB INTO "C"
   (1)  007152  000404                        BR      3$              ;;GO DO MSB
   (1)  007154  006105                2$:     ROL     R5              ;;FORM THIS DIGIT
   (1)  007156  006105                        ROL     R5
   (1)  007160  006105                        ROL     R5
   (1)  007162  010503                        MOV     R5,R3
   (1)  007164  006103                3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
   (1)  007166  105337  007270                DECB    $OMODE          ;;TYPE THIS DIGIT?
   (1)  007172  100016                        BPL     7$              ;;BR IF NO
   (1)  007174  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
   (1)  007200  001002                        BNE     4$              ;;TEST FOR 0
   (1)  007202  005704                        TST     R4              ;;SUPPRESS THIS 0?
   (1)  007204  001403                        BEQ     5$              ;;BR IF YES
   (1)  007206  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
   (1)  007210  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
   (1)  007214  052703  000040        5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
   (1)  007220  110337  007264                MOVB    R3,8$           ;;SAVE FOR TYPING
   (1)  007224  104401  007264                TYPE    ,8$             ;;GO TYPE THIS DIGIT
   (1)  007230  105337  007266        7$:     DECB    $OCNT           ;;COUNT BY 1
   (1)  007234  003347                        BGT     2$              ;;BR IF MORE TO DO
   (1)  007236  002402                        BLT     6$              ;;BR IF DONE
   (1)  007240  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
   (1)  007242  000744                        BR      2$              ;;GO DO THE LAST DIGIT
   (1)  007244  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
   (1)  007246  012604                        MOV     (SP)+,R4        ;;RESTORE R4
   (1)  007250  012603                        MOV     (SP)+,R3        ;;RESTORE R3
   (1)  007252  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
   (1)  007260  012616                        MOV     (SP)+,(SP)
   (1)  007262  000002                        RTI                     ;;RETURN
   (1)  007264     000                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
   (1)  007265     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
   (1)  007266     000                $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
   (1)  007267     000                $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
   (1)  007270  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
  884                                 ;;**************************************************************************
  885                                 .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
   (1)
   (2)                                ;;**************************************************************************
   (1)                                ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
   (1)                                ;*CHANGE IT TO BINARY.
   (1)                                ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
   (1)                                ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
   (1)                                ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
   (1)                                ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
   (1)                                ;*CALL:
   (1)                                ;*      RDOCT                   ;;READ AN OCTAL NUMBER
   (1)                                ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
   (1)                                ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
   (1)
   (1)  007272  011646                $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
   (1)  007274  016666  000004  000002        MOV     4(SP),2(SP)     ;;INPUT NUMBER
   (3)  007302  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
```

```
   (3)  0C7304  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
   (3)  0C7306  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
   (1)  007310  104410                1$:     RDLIN                   ;;READ AN ASCIZ LINE
   (1)  007312  012600                        MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
   (1)  007314  010037  007420                MOV     R0,5$           ;;AND SAVE IT
   (1)  007320  005001                        CLR     R1              ;;CLEAR DATA WORD
   (1)  007322  005002                        CLR     R2
   (1)  007324  112046                2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
   (1)  007326  001420                        BEQ     3$              ;;IF ZERO GET OUT
   (1)  007330  122716  000060                CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
   (1)  007334  003026                        BGT     4$              ;;IS AN OCTAL DIGIT
   (1)  007336  122716  000067                CMPB    #'7,(SP)
   (1)  007342  002423                        BLT     4$
   (1)  007344  006301                        ASL     R1              ;;*2
   (1)  007346  006102                        ROL     R2
   (1)  007350  006301                        ASL     R1              ;;*4
   (1)  007352  006102                        ROL     R2
   (1)  007354  006301                        ASL     R1              ;;*8
   (1)  007356  006102                        ROL     R2
   (1)  007360  042716  177770                BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
   (1)  0C7364  062601                        ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
   (1)  007366  000756                        BR      2$              ;;LOOP
   (1)  007370  005726                3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
   (1)  007372  010166  000012                MOV     R1,12(SP)       ;;SAVE THE RESULT
   (1)  007376  010237  007430                MOV     R2,$HIOCT
   (3)  007402  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
   (3)  007404  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
   (3)  007406  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
   (1)  007410  000002                        RTI                     ;;RETURN
   (1)  007412  005726                4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
   (1)  007414  105010                        CLRB    (R0)            ;;SET A TERMINATOR
   (1)  007416  104401                        TYPE                    ;;TYPE UP THRU THE BAD CHAR.
   (1)  007420  000000                5$:     .WORD   0
   (1)  007422  104401  001160                TYPE    ,$QUES          ;;''?'' ''CR'' & 'LF''
   (1)  007426  000730                        BR      1$              ;;TRY AGAIN
   (1)  007430  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.
007432  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
007434  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
007440  005740                          TST     -(R0)           ;;BACKUP BY 2
007442  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
007444  006300                          ASL     R0              ;;POSITION FOR INDEXING
007446  016000  007466                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
007452  000200                          RTS     R0              ;;GO TO ROUTINE


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

007454  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
007456  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
007464  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
007466  007454                  $TRPAD: .WORD   $TRAP2
007470  006624                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
007472  007070                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
007474  007044                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
007476  007104                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)

007500  005772                          $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING

007502  005722                          $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
007504  006204                          $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
007506  006324                          $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
007510  007272                          $RDOCT  ;;CALL=RDOCT    TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
```

```
                                        ;;*****************************************************************
                                        .SBTTL  POWER DOWN AND UP ROUTINES
                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
007512  012737  007656  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
007520  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
007526  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
007530  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
007532  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
007534  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
007536  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
007540  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
007542  017746  171372                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
007546  010637  007662                  MOV     SP,$SAVR6       ;;SAVE SP
007552  012737  007564  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
007560  000000                          HALT
007562  000776                          BR      .-2             ;;HANG UP
                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
007564  012737  007656  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
007572  013706  007662                  MOV     $SAVR6,SP       ;;GET SP
007576  005037  007662                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
007602  005237  007662          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
007606  001375                          BNE     1$              ;;OF  WORD
007610  012677  171324                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
007614  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
007616  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
007620  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
007622  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
007624  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
007626  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
007630  012737  007512  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
007636  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
007644  104401                          TYPE                    ;;REPORT THE POWER FAILURE
007646  010056          $PWRMG: .WORD   PWRMSG                  ;;POWER FAIL MESSAGE POINTER
007650  012716                  MOV     (PC)+,(SP)              ;;RESTART AT START1
007652  001562          $PWRAD: .WORD   START1                  ;;RESTART ADDRESS
007654  000002                  RTI
007656  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
007660  000776                  BR      .-2                     ;; BEFORE THE POWER DOWN WAS COMPLETE
007662  000000          $SAVR6: 0                               ;;PUT THE SP HERE
```

```
  893                                                   .SBTTL  DISPLAY MESSAGES
  894                                                   ;THIS IS THE MESSAGE FOR THE BOTTOM OF THE SCOPE
  895  007664  005015  047514  042527  WD001:  .ASCIZ  <15><12>/LOWER THRESHOLD /
       007672  020122  044124  042522
       007700  044123  046117  020104
       007706     000
  896  007707     040  020040  020040  WD002:  .ASCIZ  /     UPPER THRESHOLD /
       007714  050125  042520  020122
       007722  044124  042522  044123
       007730  046117  020104     000
  897  007735     015  055012  041440  WD003:  .ASCIZ  <15><12>/Z COUNT=/
       007742  052517  052116  000075
  898  007750  046440  052101  044522  WD004:  .ASCIZ  / MATRIX COUNT-/
       007756  020130  047503  047125
       007764  036524     000
  899  007767     040  055040  047517  WD005:  .ASCIZ  /  ZOOM/
       007774  000115
  900  007776  041040  043455  046501  WD006:  .ASCIZ  / B-GAMMA/
       010004  040515     000
  901  010007     015  041412  046501  WD007:  .ASCII  <15><12>/CAMERA # /
       010014  051105  020101  020043
  902  010022     060                  CAMOUT: .BYTE   60
  903  010023     040  020040  020040          .ASCII  /     PB IS /
       010030  041120  044440  020123
  904  010036  050125  020040  020040  PBUP:   .ASCII  /UP     JB IS /
       010044  045040  020102  051511
       010052     040
  905  010053     125  000120          JBUP:   .ASCIZ  /UP/
  906
  907                                                   .SBTTL  ASCII MESSAGES
  908
  909  010056  005015  051012  051505  PWRMSG: .ASCIZ  <15><12><12>/RESTARTED AFTER POWER FAILURE /
       010064  040524  052122  042105
       010072  040440  052106  051105
       010100  050040  053517  051105
       010106  043040  044501  052514
       010114  042522  000040
  910  010120  005015  041412  047132  MSG1:   .ASCIZ  <15><12><12>/CZNCDB   NCV-11 EXERCISER    /
       010126  042103  020102  020040
       010134  041516  026526  030461
       010142  042440  042530  041522
       010150  051511  051105  020040
       010156  020040     000
  911  010161     015  042412  052116  MSG2:   .ASCIZ  <15><12>/ENTER THRESHOLD VALUE IN OCTAL - THEN RETURN -/
       010166  051105  052040  051110
       010174  051505  047510  042114
       010202  053040  046101  042525
       010210  044440  020116  041517
       010216  040524  020114  020055
       010224  044124  047105  051040
       010232  052105  051125  020116
       010240  000075
  912  010242  005015  052502  020123  MSG3:   .ASCIZ  <15><12>/BUS TIMEOUT ERROR - NO VSV01 DISPLAY DETECTED  /
       010250  044524  042515  052517
       010256  020124  051105  047522
       010264  020122  020055  047516
```

```
        010272  053040  053123  030460
        010300  042040  051511  046120
        010306  054501  042040  052105
        010314  041505  042524  020104
        010322  000040
  913   010324  005015  047105  042524  MSG4:   .ASCIZ  <15><12>/ENTER CONSOLE BUS ADDRESS - THEN RETURN  /
        010332  020122  047503  051516
        010340  046117  020105  052502
        010346  020123  042101  051104
        010354  051505  020123  020055
        010362  044124  047105  051040
        010370  052105  051125  020116
        010376  000075
  914   010400  005015  052502  020123  MSG5:   .ASCIZ  <15><12>/BUS TIMEOUT ERROR - NCV11/
        010406  044524  042515  052517
        010414  020124  051105  047522
        010422  020122  020055  041516
        010430  030526  000061
  915   010434  005015  054524  042520  MSG6:   .ASCIZ  <15><12>/TYPE 'H' FOR HELP INFO  ENTER KEYBOARD COMMAND(S)/
        010442  021040  021110  043040
        010450  051117  044040  046105
        010456  020120  047111  047506
        010464  020040  047105  042524
        010472  020122  042513  041131
        010500  040517  042122  041440
        010506  046517  040515  042116
        010514  051450  000051
  916   010520  005015  047105  042524  MSG7:   .ASCIZ  <15><12>/ENTER CAMERA NUMBER 0-3 ? /
        010526  020122  040503  042515
        010534  040522  047040  046525
        010542  042502  020122  026460
        010550  020063  020077     000
  917   010555     015  042412  052116  MSG8:   .ASCIZ  <15><12>/ENTER CONSOLE VECTOR ADDRESS - THEN RETURN -/
        010562  051105  041440  047117
        010570  047523  042514  053040
        010576  041505  047524  020122
        010604  042101  051104  051505
        010612  020123  020055  044124
        010620  047105  051040  052105
        010626  051125  020116  000075
  918   010634  026440  000040           DASH:   .ASCIZ  / - /
  919   010640  005015  042115  030455  HELP0:  .ASCII  <15><12>/MD-11-CZNCD-B NCV11 EXERCISER/<15><12><12>
        010646  026461  055103  041516
        010654  026504  020102  041516
        010662  030526  020061  054105
        010670  051105  044503  042523
        010676  006522  005012
  920   010702  020104  020040  020040          .ASCII  /D      DISPLAY DATA/<15><12>
        010710  042040  051511  046120
        010716  054501  042040  052101
        010724  006501     012
  921   010727     105  020040  020040          .ASCII  /E      ERASE SCOPE/<15><12>
        010734  020040  051105  051501
        010742  020105  041523  050117
        010750  006505     012
  922   010753     114  020040  020040          .ASCII  /L      GET LOWER THRESHOLD/<15><12>
```

```
                010760  020040  042507  020124
                010766  047514  042527  020122
                010774  044124  042522  044123
                011002  046117  006504     012
    923         011007     125  020040  020040          .ASCII  /U      GET UPPER THRESHOLD/<15><12>
                011014  020040  042507  020124
                011022  050125  042520  020122
                011030  044124  042522  044123
                011036  046117  006504     012
    924         011043     101  020040  020040          .ASCII  /A      DISPLAY ISOTOPE A/<15><12>
                011050  020040  044504  050123
                011056  040514  020131  051511
                011064  052117  050117  020105
                011072  006501     012
    925         011075     102  020040  020040          .ASCII  /B      DISPLAY ISOTOPE B/<15><12>
                011102  020040  044504  050123
                011110  040514  020131  051511
                011116  052117  050117  020105
                011124  006502     012
    926         011127     127  020040  020040          .ASCII  /W      SELECT FIRST BIT MAP/<15><12>
                011134  020040  042523  042514
                011142  052103  043040  051111
                011150  052123  041040  052111
                011156  046440  050101  005015
    927         011164  020115  020040  020040          .ASCII  /M      SELECT SECOND BIT MAP/<15><12>
                011172  051440  046105  041505
                011200  020124  042523  047503
                011206  042116  041040  052111
                011214  046440  050101  005015
    928         011222  020106  020040  020040          .ASCII  /F      FREE RUN MODE/<15><12>
                011230  043040  042522  020105
                011236  052522  020116  047515
                011244  042504  005015
    929         011250  020112  020040  020040          .ASCII  /J      JOYSTICK CALIB./<15><12>
                011256  045040  054517  052123
                011264  041511  020113  040503
                011272  044514  027102  005015
    930         011300  020124  020040  020040          .ASCIZ  /T      DISPLAY INTENSITY TEST/
                011306  042040  051511  046120
                011314  054501  044440  052116
                011322  047105  044523  054524
                011330  052040  051505  000124
    951         011336  005015  020116  020040  HELP1:  .ASCII  <15><12>/N      COLLECT NEW DATA/<15><12>
                011344  020040  041440  046117
                011352  042514  052103  047040
                011360  053505  042040  052101
                011366  006501     012
    932         011371     103  020040  020040          .ASCII  /C      COLLECT MORE DATA/<15><12>
                011376  020040  047503  046114
                011404  041505  020124  047515
                011412  042522  042040  052101
                011420  006501     012
    933         011423     130  020040  020040          .ASCII  /X      CHANGE CAMERA CHANNEL/<15><12>
                011430  020040  044103  047101
                011436  042507  041440  046501
                011444  051105  020101  044103
```

```
        011452  047101  042516  006514
        011460     012
  934   011461     123  020040  020040          .ASCII  /S      STOP COLLECTION/<15><12>
        011466  020040  052123  050117
        011474  041440  046117  042514
        011502  052103  047511  006516
        011510     012
  935   011511     107  020040  020040          .ASCII  /G      INITILIZE/<15><12>
        011516  020040  047111  052111
        011524  046111  055111  006505
        011532     012
  936   011533     132  020040  020040          .ASCII  /Z      ZOOM GAIN/<15><12>
        011540  020040  047532  046517
        011546  043440  044501  006516
        011554     012
  937   011555     122  020040  020040          .ASCII  /R      REGULAR GAIN/<15><12>
        011562  020040  042522  052507
        011570  040514  020122  040507
        011576  047111  005015
  938   011602  020117  020040  020040          .ASCII  /O      OTHER CONSOLE TERMINAL/<15><12>
        011610  047440  044124  051105
        011616  041440  047117  047523
        011624  042514  052040  051105
        011632  044515  040516  006514
        011640     012
  939   011641     110  020040  020040          .ASCII  /H      HELP THE OPERATOR AND REPEAT THIS LIST/
        011646  020040  042510  050114
        011654  052040  042510  047440
        011662  042520  040522  047524
        011670  020122  047101  020104
        011676  042522  042520  052101
        011704  052040  044510  020123
        011712  044514  052123
  940   011716     015     012     000  BCRLF:  .BYTE   15,12,0
  941
  942                                  ;LOC. 20000 THRU 57776 ARE BUFFER AREA
  943
  944           000001                         .END
```

```
ABASE = 172760              10#     32      50      51      52      53      54      55      56      57
AWRIT   005342             515     519     546     550     568     572     808#
BCRLF   011716             575     940#
BELLEN  001322              95#
BIT0  = 000001              11#     630     689
BIT00 = 000001              11#
BIT01 = 000002              11#
BIT02 = 000004              11#
BIT03 = 000010              11#
BIT04 = 000020              11#
BIT05 = 000040              11#
BIT06 = 000100              11#
BIT07 = 000200              11#
BIT08 = 000400              11#
BIT09 = 001000              11#
BIT1  = 000002              11#
BIT10 = 002000              11#     320     370
BIT11 = 004000              11#     629
BIT12 = 010000              11#
BIT13 = 020000              11#
BIT14 = 040000              11#
BIT15 = 100000              11#
BIT2  = 000004              11#
BIT3  = 000010              11#
BIT4  = 000020              11#
BIT5  = 000040              11#     233     525     638
BIT6  = 000100              11#     533
BIT7  = 000200              11#
BIT8  = 000400              11#     236     277     322     344     696
BIT9  = 001000              11#
BOOBOO  002034              170     173     176     212#    289     304     315
BOX     003144              384     388#
BPTVEC= 000014              11#
CAMERA  001300              86#     362*    363     521     688
CAMOUT  010022              523*    902#
CHANGE  003122              218     220     223     241     270     327     337     367     377#
CHARCT  001306              89#     419*    428*
CKSWR = 104406              888#
COMSAV  001276              85#     590     695*
CPERL   001310              90#     436*    456
CR    = 000015              11#     881
CRLF  = 000200              11#     881
DASH    010634              549     571     918#
DCONT   005072              394     401     607     614     730#
DDISP = 177570              11#      21     102
DISBG   004462              629#
DISLOP  003420              444#    496     508
DISPLA  001142              21#     102*
DISPRE  000174              18#     102
DISPY   005060              390     397     408     411     494     603     610     618     621     728#
DISPY2  005122              642     741#    742
DMATRX  003376              441#
DSWR  = 177570              11#      21     102
DUMMY1  001246              73#     521*    522*    523     544*    551     566*    573
DUMMY2  001250              74#     545*    547     567*    569
EMTVEC= 000030              11#
```

```
ERASE   005020            224     242     246     377     598     715#
ERRVEC= 000004             11#    102*    114*    121*    290     291*    299*    313*    667*    676*
FREERN  001316             93#    160*    226*    587
GAIN    001302             87#    125*    319*    370*    371     579     687
GNS   = ****** U           18     888
GTSWR = 104405            888#
HELP0   010640            251     919#
HELP1   011336            253     255     931#
HT    = 000011             11#    881
INTLUT  001252             75#    670
IOTVEC= 000020             11#
JBUP    010053            535*    536*    538*    539*    905#
KBINT   005220            129     773#
KBUFF   001266             81#    162*    166     168*    233*    234     773*    774     775*    776
LARGES  003316            419#
LDIMGE  005152            326     755#
LF    = 000012             11#    881
LISEN   001656            160#    238     778
LISN    001702            164     166#    167     215     222     225     256     259     278     281     321     324     345
                          372     385     591
MAPADR  001260             78#    414*    476     499*    500     507*
MAPLD   003542            455     471     476#
MATRIX= 020000             16#     92     124     126     217     219     583     686     706     708
MRXADR  001256             77#    441*    442*    444     445*    449*
MSELCT  001320             94#    273*    340*    660
MSG1    010120            103     910#
MSG2    010161            264     331     911#
MSG3    010242            107     912#
MSG4    010324            285     913#
MSG5    010400            118     914#
MSG6    010434            132     915#
MSG7    010520            349     916#
MSG8    010555            300     917#
MXSML   004202            562#    565
NCADM   001212             55#
NCADR   001164             32#    108
NCBAR   001206             53#    544     705*
NCBAR1  001216             57#
NCCSR   001200             50#    109     115     237*    320*    323*    363*    371*    590*    685*    687*    688*    689*
                          695     697*
NCJOY   001214             56#    633
NCOFF   001202             51#    124*    686*
NCSET   001516            110#    113
NCSFR   001210             54#    236*    322*    525     533     629*    630*    631     638     696*
NCSTP   004744            325     381     695#    703
NCSTP1  004766            123     227     279     703#
NCSTRT  004704            221     228     280     685#
NCWCR   001204             52#    545     704*
NOVSV   001324             96#    106*    244     257     271     338     383     715
NUMBEG  005462            825     830#
NUMEND  005470            811     831#
PATWRT  003720            435     503     512#
PBUP    010036            527*    528*    530*    531*    904#
PIRQ  - 177772             11#
PIRQVE- 000240             11#
PIXASM  001264             80#    416*    486*    487*    488*    489*    490*    493     497*
```

```
PIXCNT  001262              79#     415*     491      495*     498*
PR0    = 000000             11#
PR1    = 000040             11#
PR2    = 000100             11#
PR3    = 000140             11#
PR4    = 000200             11#
PR5    = 000240             11#
PR6    = 000300             11#
PR7    = 000340             11#
PS     = 177776             11#
PSW    = 177776             11#
PWRMSG  010056              891      909#
PWRVEC= 000024               11#     102*     89 *
RDCHR  = 104407             879      888#
RDLIN  = 104410             885      888#
RDOCT  = 104411             265      286      301      332      350      888#
RESVEC= 000010               11#
ROUTA   002054              182      217#
ROUTB   002066              183      219#
ROUTC   002100              184      221#
ROUTD   002110              185      223#
ROUTE   002114              186      224#
ROUTF   002124              187      226#     589
ROUTG   002220              188      242#
ROUTH   002230              189      244#
ROUTJ   002302              191      257#
ROUTL   002320              193      261#
ROUTM   002360              194      271#
ROUTN   002420              195      279#
ROUTO   002434              196      282#
ROUTR   002614              199      319#
ROUTS   002632              200      322#
ROUTT   002650              201      325#
ROUTU   002664              202      328#
ROUTW   002724              204      338#
ROUTX   002762              205      346#
ROUTZ   003102              207      370#
ROWCNT  001312               91#     443*     446*     448*
RTABLE  001750              175      177      182#
SCLCEL  003474              453      456#
SELCTA  004546              104      274      341      653#
STACK -  001100              11#     102      161
START   001326               18      102#     120      310
START1  001562              116      121#     243      891
STKLMT- 177774               11#
SWR     001140               21#     102*     879*     891*
SWREG   000176               18#     102      879
SW0    - 000001              11#
SW00   = 000001              11#
SW01   = 000002              11#
SW02   = 000004              11#
SW03   = 000010              11#
SW04   = 000020              11#
SW05   = 000040              11#
SW06   = 000100              11#
SW07   - 000200              11#
```

```
SW08  = 000400              11#
SW09  = 001000              11#
SW1   = 000002              11#
SW10  = 002000              11#
SW11  = 004000              11#
SW12  = 010000              11#
SW13  = 020000              11#
SW14  = 040000              11#
SW15  = 100000              11#
SW2   = 000004              11#
SW3   = 000010              11#
SW4   = 000020              11#
SW5   = 000040              11#
SW6   = 000100              11#
SW7   = 000200              11#
SW8   = 000400              11#
SW9   = 001000              11#
TABLEX  001314              92#   126*  217*  219*  420   421   441   561   583   756
TBITVE= 000014              11#
TEMP0   001240              70#   558*  564*  633*  743*  744   838   852*
TEMP1   001242              71#
TEMP2   001244              72#
THHI    001274              84#   128*  333*  424   450   520
THLO    001272              83#   127*  266*  433   452   516
TIME    001172              44#   230
TJOY    004332             260    598#
TKVEC = 000060              11#    46    47
TKVEC0  001174              46#   129*  306*
TKVEC1  001176              47#   130*  307*  308*
TOTSIZ  001304              88#   419   558
TPVEC = 000064              11#
TRAPVE  000034              11#   102*
TRTVEC= 000014              11#
TTYOUT  001270              82#   212*  784*  786*  789
TYPCR   005260             214    784#
TYPE  - 104401             103    107   118   132   255   264   285   300   331   349   879   881   883
                           885    888#  891
TYPO    005300             213    785   787#  788
TYPOC   104402             879    888#
TYPON   104404             888#
TYPOS - 104403             888#
VTMADR  001170              40#   659
VTVADR  001166              36#   653
VTVCHP  001222              60#   744*
VTVCRG  001220              59#   112   654   668   721*  741   745*  797   801*
VTVCSR  001226              63#   276   343   413*  625*  657   669   717*  718   720*  728*  730*  731
                           734*
VTVINT  001236              67#   665   671*
VTVMAP  001230              64#   729*
VTVPOS  001224              61#   249*  512*
VTVPX   001232              65#   733*
VTVPX1  001234              66#
VTVSAV  001254              76#   413   625
VTWRIT  005316             250    252   513   517   540   542   548   552   570   574   581   585   796#
                           824
WD001   007664             514    895#
```

```
WD002   007707            518     896#
WD003   007735            543     897#
WD004   007750            553     898#
WD005   007767            582     899#
WD006   007776            586     900#
WD007   010007            541     901#
XYAVE   005472            637     837#
XYBUF   005620            843     845     848     870#    876
XYBUFE= 005720            841     861     875#
XYBUFP  005720            839     844*    876#
$AUTOB  001134             21#    879
$BDADR  001122             21#
$BDDAT  001126             21#
$CHARC  007040            881#*
$CKSWR  005722            879#    888
$CMTAG  001100             21#    102
$CM3  = 000000             21#
$CNTLG  006574            879#
$CNTLU  006567            879#
$CRLF   001161             21#    879     881     885
$ERFLG  001103             21#
$ERMAX  001115             21#
$ERRPC  001116             21#
$ERRTB  001164             21#
$ERTTL  001112             21#
$FILLC  001156             21#    881
$FILLS  001155             21#    881
$GDADR  001120             21#
$GDDAT  001124             21#
$GTSWR  005772            879#    888
$HD   = 000003              9
$HIOCT  007430            885#*
$ICNT   001104             21#
$ILLUP  007656            891#
$INTAG  001135             21#    879
$ITEMB  001114             21#
$LF     001162             21#    879     881     885
$LPADR  001106             21#
$LPERR  001110             21#
$MAIL = ****** U           102     881
$MNEW   006612            879#
$MSWR   006601            879#
$NULL   001154             21#    881
$OCNT   007266            883#*
$OMODE  007270            883#*
$PASS   001100             21#
$PWRAD  007652            891#
$PWRDN  007512            102     891#
$PWRMG  007646            891#
$PWRUP  007564            891#
$QUES   001160             21#    879     881     885
$RDCHR  006204            879#    888
$RDDEC- ****** L           888
$RDLIN  006324            879#    888
$RDOCT  007272            885#    888
$RDSZ - 000007            879#
```

```
$R2A  = ****** U          888
$SAVRE= ****** U          888
$SAVR6  007662            891#*
$SETUP= 000114            101#    102     879
$STUP = 177777            101#
$SWR  = 160000             9#     21      891
$TKB    001146            21#    773     879
$TKS    001144            21#    131*    293     879*
$TN   = 000001             9#
$TPB    001152            21#    774*    789*    881*
$TPFLG  001157            21#    881
$TPS    001150            21#    787     881
$TRAP   007432            102    888#
$TRAP2  007454            888#
$TRP  = 000012            888#
$TRPAD  007466            888#
$TSTNM  001102             21#
$TTYIN  006560            879#
$TYPBN= ****** U          888
$TYPDS= ****** U          888
$TYPE   006624            881#    888
$TYPEC  006774            879     881#
$TYPEX  007042            881#
$TYPOC  007070            883#    888
$TYPON  007104            883#    888
$TYPOS  007044            883#    888
$OFILL  007267            883#*
.       011721             18#     19#     21#     102     832#    875     879#    881     885     891
```

```
COMMEN      11#
ENDCOM      11#
ERROR       11#
ESCAPE      11#
GETPRI      11#
GETSWR      11#
MULT        11#
NEWTST      11#
POP         11#     885     891
PUSH        11#     885     891
REPORT      11#
SCOPE       11#
SETPRI      11#
SETTRA     888#
SETUP       11#     102
SKIP        11#
SLASH       11#
SPACE       11#
STARS       11#      21     648     652     682     684     692     694     700     702     712     714     724     727     738
           740     748     754     770     772     781     783     792     794     805     807     834     836     878     879
           880     881     882     883     884     885     887     888     890     891
SWRSU       11#     102#
TRMTRP     888#
TYPBIN      11#
TYPDEC      11#
TYPNAM      11#
TYPNUM      11#
TYPOCS      11#
TYPOCT      11#     879
TYPTXT      11#
$$CMRE      21#
$$CMTM      21#
$$ESCA      11#
$$NEWT      11#
$$SET      888#
$$SKIP      11#
.EQUAT       6#      11
.HEADE       6#       9
.SETUP       6#     101
.$CATC       6#      18
.$CMTA       6#      21
.$POWE       7#     891
.$RDOC       7#     885
.$READ       7#     879
.$TRAP       7#     888
.$TYPE       7#     881
.$TYPO       7#     883

. ABS.  01172'      000     CON    RO    ABS    GBL    D

 ERRORS DETECTED:  0

 CZNCDB,CZNCDB/CR:-CZNCDB
 RUN-TIME: 13 6 .7 SECONDS
```

RUN-TIME RATIO: 30/20=1.4
CORE USED: 18K (35 PAGES)