NC11-A

DIAGNOSTIC

MD-11-DZNCA-C

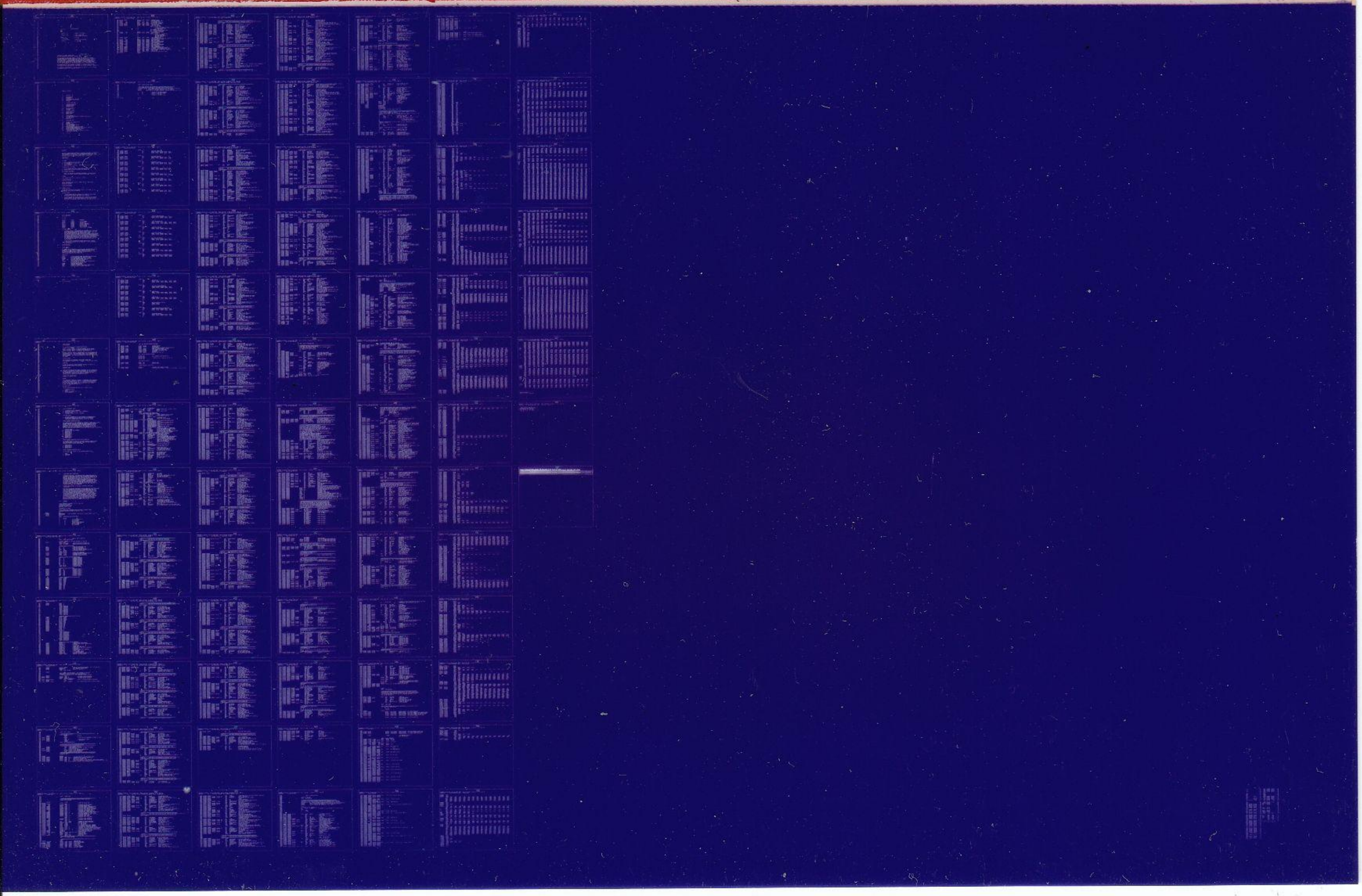
EP-DZNCA-C-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA



.REM %

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZNCA-C-D
PRODUCT NAME:	NC11-A LOGIC TEST (GAMMA-CAMERA INTERFACE)
DATE:	MAY 1976
MAINTAINER:	DIAGNOSTIC ENGINEERING
AUTHOR:	R. MOORE

TABLE OF CONTENTS

1.0 ABSTRACT

THE NC11 (GAMMA-CAMERA INTERFACE) DIAGNOSTIC PROGRAM IS A SERIES OF TESTS DESIGNED TO TEST ALL LOGIC FUNCTIONS AND DATA PATHS. THE GAMMA-CAMERA IS NOT REQUIRED. TIMING MARK & JOYSTICK/ADC TESTING IS OPTIONAL (START 204) WHICH REQUIRES MANUAL INTERVENTION.

2.0 REQUIREMENTS

2.1 EQUIPMENT

1. PDP-11 COMPUTER WITH AT LEAST 16K OF MEMORY
2. I/O TERMINAL (I.E. ASR33 TTY).
3. NC11 INTERFACE
4. H306 JOYSTICK - OPTIONAL - SEE SECTION 9.8 & 9.9 THIS DOCUMENT

2.2 DOCUMENTATION - OPTIONAL

1. NC11-A CAMERA FRONT-END INSTRUCTION MANUAL
2. DRAWING SET NC11-A-0 (CONTAINS TEST PROCEDURES)

2.3 PRELIMINARY OPERATIONS

1. PART 1 OF THE NC11-A ADJUSTMENT PROCEDURE (A-SP-NC11-A-23) SHOULD HAVE BEEN COMPLETED BEFORE RUNNING THIS DIAGNOSTIC.

2.4 STORAGE

THIS PROGRAM USES ALL OF LOWER 16K OF MEMORY.

3.0 LOADING PROCEDURE

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

4.0 STARTING PROCEDURE

4.1 OPERATOR ACTION

INSURE THAT THE NC11-A 'TEST MODE' SWITCH (ON THE BACK OF THE INTERFACE) IS IN THE OFF POSITION.

4.2 PROGRAM STARTS

1. LOADING ADDRESS 200 AND STARTING WILL IDENTIFY THE PROGRAM, INITIALIZE THE SYSTEM, AND BEGIN LOGIC TESTING.

2. LOADING ADDRESS 204 AND STARTING WILL INITIALIZE THE SYSTEM, AND GO DIRECTLY TO THE TIMING MARK & JOYSTICK LOGIC TESTS. NOTE THAT THIS REQUIRES USER INTERACTION (SEE SECTION 9.8 & 9.9).

5.0 SWITCH REGISTER

5.1 OPTIONS

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON TEST
SW13=1	020000	INHIBIT TYPEOUTS
SW11=1	004000	INHIBIT ITERATIONS
SW10=1	002000	BELL ON ERROR
SW09=1	001000	LOOP ON ERROR
SW08=1	0004XX	LOOP ON TEST IN SWR<7-0>

5.2 SOFTWARE CONTROL

1. COMPUTERS WITHOUT A HARDWARE SWITCH REGISTER HAVE A SOFTWARE SWITCH REGISTER LOCATION IN MEMORY CALLED 'SWREG' (LOC 176) WHICH CAN BE CHANGED BY THE CONSOLE FACILITIES.
2. THE SOFTWARE SWITCH REGISTER CAN ALSO BE CHANGED UNDER PROGRAM CONTROL BY TYPING THE 'CNTRL & G'* KEYS FOLLOWED BY ENTERING NEW SWITCH REGISTER OCTAL DATA TERMINATED BY A CARRIAGE RETURN.
3. THE PROGRAM WILL DEFAULT TO THE SOFTWARE SWITCH REGISTER IF THE USER STARTS THE PROGRAM WITH ALL SWITCHES SET. CONTROL OF THE SWITCH REGISTER IS THEN ACCOMPLISHED AS DEFINED IN STEP #2 ABOVE.

* IF THE PROGRAM IS PERFORMING RESET INSTRUCTIONS, SEVERAL 'CONTROL & G' COMMANDS MAY BE NECESSARY TO BE ACKNOWLEDGED BY THE PROGRAM.

6.0 ERROR REPORTING

6.1 ERROR COMMENT

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED IF NEEDED FROM THE COMMENT AT THE ERROR PC OR FROM THE TEST ITSELF. SEE SECTION 9.9 FOR JOYSTICK/ADC ERROR AND DATA REPORTING.

6.2 ERROR DATA

*ERRPC	LISTING ADDRESS WHERE THE ERROR WAS DETECTED
*TSTNUM	TEST NUMBER WHERE THE ERROR OCCURRED
BUSADR	NC11 BUS ADDRESS WHERE 'RCVD' WAS READ
EXPCT	REGISTER DATA THAT WAS EXPECTED
RCVD	ACTUAL REGISTER DATA READ
XYHOLD	DATA WRITTEN TO XYHOLD REG. ON ADDRESS MAKE, OR DATA WRITTEN IN LIST MODE
GDADR	EXPECTED MAKE ADDRESS
BDADR	HARDWARE ADDRESS MADE
ADRS	ADDRESS OF CURRENT NPR FOR ALL RESOLUTIONS
GDDAT	EXPECTED MEMORY CONTENTS AFTER NPR

```
                                        BUDAT           ACTUAL MEMORY CONTENTS AFTER NPR
                                       *ALWAYS REPORTED
```

7.0 MISCELLANEOUS

7.1 NC11 BUS/VECTOR/PRIORITY ADDRESS MODIFICATION

MODIFY LOCATION '$BASE' IF BASE BUS ADDRESS IS NOT 164000.
MODIFY LOCATION '$VECT1' IS BASE VECTOR ADDRESS IS NOT 270.*
MODIFY LOCATION '$VECT1' IF PRIORITY LEVEL IS NOT 7.*

* CAUTION, LOCATION '$VECT1' CONTAINS BOTH THE VECTOR ADDRESS AND THE PRIORITY LEVEL. THE TOP ORDER 3 BITS (15-13) REPRESENT THE PRIORITY LEVEL (1-7) AND THE LOWER ORDER 13 BITS REPRESENT THE VECTOR ADDRESS. WHEN CHANGING EITHER ONE, MAKE SURE YOU REPLACE THE OTHER.

7.2 XXDP/APT NOTES

THIS DIAGNOSTIC IS CHAINABLE (START 200) UNDER XXDP.
THIS DIAGNOSTIC SUPPORTS 'APT', HOWEVER, IT HAS NOT BEEN TESTED.

7.3 POWER FAIL

A POWER FAILURE WILL CAUSE A RESTART MESSAGE ON POWER UP AT WHICH TIME THE PROGRAM IS RESTARTED.

8.0 EXECUTION TIME

EXECUTION TIME RANGES FROM ABOUT 30 SECONDS WITH NO ITERATIONS TO ABOUT 3 MINUTES WITH ITERATIONS ENABLED. IF THE USER SELECTS THE TIMING MARK & JOYSTICK/ADC TESTS AT PROGRAM START 204, THE PROGRAM WILL REMAIN IN THIS SECTION CONTINUALLY OUTPUTTING JOYSTICK DATA TO THE PRINTER DEPENDING ON SW13.

9.0 PROGRAM TEST DESCRIPTIONS

9.1 GENERAL

THIS DIAGNOSTIC CONTAINS A SERIES OF INDEPENDENT TESTS DESIGNED TO TEST LOGIC FUNCTIONS OF THE NC11-A GAMMA-CAMERA INTERFACE. EACH TEST IS LISTED IN THE TABLE OF CONTENTS AT THE BEGINNING OF THE LISTING. THE COMMENT FIELD WITHIN EACH TEST CAN BE HELPFUL IN UNDERSTANDING EACH TEST.

9.2 REGISTER TESTS

THE FOLLOWING REGISTERS ARE READ/WRITE & RESET TESTED:

1. COMMAND AND STATUS
2. OFFSET
3. HIGH & LOW ORDER Z (WC & CA)
4. DATA INCREMENT (I)

9.3 FLAG OR INTERRUPT TESTS

1. Z OVERFLOW FLAG & INTERRUPT
2. WORD COUNT OVERFLOW FLAG (WCO) & INTERRUPT
3. TIME OUT FLAG
4. INCREMENT OVERFLOW FLAG & INTERRUPT

9.4 LIST MODE TESTS

1. ONE WORD TRANSFER OF ALL DATA PATTERNS TO ADDRESS 20000.
2. A 8192 WORD TRANSFER OF COMPLIMENTING DATA PATTERNS
   STARTING AT ADDRESS 20000 WITH WC & CA & WCO CHECKS.

9.5 PROPER ADDRESS GENERATION TESTS

THE ADDRESS MAKER OF THE INTERFACE IS TESTED BY A RESOLUTION (MATRIX)
RUNNING THRU ALL POSSIBLE SOFTWARE ADC VALUES AND ALL ADDRESS
OFFSETS. AFTER A SUCCESSFUL PASS OF ADDRESS GENERATION OF A
RESOLUTION THE TEST IS REPEATED WITH THE NPR BIT IN THE CSR
ENABLED. THIS WILL ALLOW CHECKING OF THE ADDRESS PATH TO THE
BUS ALONG WITH THE NPR INCREMENT LOGIC (NPR'S ARE LIMITED TO 16K OF MEM).

1. JOYSTICK MODE
2. JOYSTICK MODE FOR OVERRIDE
3. RESOLUTION 0
4. RESOLUTION 1
5. RESOLUTION 2
6. RESOLUTION 3
7. RESOLUTION 4
8. RESOLUTION 5
9. RESOLUTION 6
10. RESOLUTION 7

9.6 PROPER ADDRESS GENERATION TESTS WITH B GAMMA SET

THE ADDRESS MAKER IS TESTED IN THE SAME MANNER AS ABOVE EXCEPT
THAT THE NPR PART IS NOT EXERCISED. DUAL ISOTOPE OPERATION
APPLIES TO THE FOLLOWING RESOLUTIONS:

1. RESOLUTION 2
2. RESOLUTION 3
3. RESOLUTION 6
4. RESOLUTION 7

9.7 MEMORY INCREMENT OPERATION TESTS

THE FOLLOWING ARE TESTED IN THE MEMORY INCREMENT TESTS:

1. WORD
2. EVEN BYTE
3. ODD BYTE

```
                                9.8     TIMING MARK TEST (START 204)

                                        TO EXECUTE THIS TEST, TWO SHORTING 'BNC' CONNECTIONS MUST BE
                                        INSERTED IN BOTH THE AZ AND BZ 'BNC' CONNECTORS LOCATED AT THE
                                        BACK OF THE INTERFACE. ALSO THE JOYSTICK MUST BE APPROXIMATELY
                                        CENTERED BEFORE SELECTING THIS TEST. NOTE THAT THE
                                        JOYSTICK POTS MUST BE IN RANGE OF THE NAD01'S OR A POSSIBLE
                                        NO CONVERT SITUATION WILL CAUSE AN ERROR. (REQUIREMENTS IN SP-
                                        NC11-A-23 MUST HAVE BEEN VERIFIED BEFORE SELECTING THIS TEST).
                                        THE TEST IS LOOKING FOR THE TIME MARK FLOP TO SET BIT 15 IN
                                        AN ADDRESS MAKE IN LIST MODE. WHEN THIS TEST HAS COMPLETED THE
                                        JOYSTICK/ADC LOGIC TEST IS ENTERED.

                                9.9     JOYSTICK ADC CONVERT, NO CONVERT & JOYSTICK LOGIC TEST
                                        (START 204).

                                        THIS TEST IS EXECUTED CONTINUALLY AFTER COMPLETING THE TIMING MARK
                                        TEST. THE PURPOSE OF THIS TEST IS TO DETERMINE THAT THE NC11
                                        JOYSTICK LOGIC AND ADC'S ARE FUNCTIONING (REQUIREMENTS IN SP
                                        NC11-A-23 MUST BE VERIFIED BEFORE SELECTING THIS TEST). THERE
                                        WILL BE A CONSTANT PRINTER OUTPUT OF X & Y ADC CONVERTED
                                        VALUES UPON SUCCESSFUL CONVERSIONS, OR A 'NO CONVRT' MESSAGE
                                        IF THE ADC'S FAIL TO CONVERT (OUT OF RANGE), OR A 'BAR DN'
                                        MESSAGE IF THE INTERRUPT BAR IS DEPRESSED. SW13 WILL INHIBIT
                                        ALL TYPEOUTS MENTIONED. THE TEST IS DESIGNED TO INDICATE THAT
                                        ALL X & Y DATA BITS ARE GOOD, THAT THE NO CONVERT BIT WILL
                                        SET WITH THE JOYSTICK OUT OF RANGE, AND THAT THE INTERRUPT
                                        BAR ON THE H306 JOYSTICK IS SENSED BY THE LOGIC.

                                10.0    LISTING
                                        -------
                                        %
                                .TITLE  MAINDEC-11-DZNCA-C  NC11-A LOGIC TEST
                                ;*COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY R. MOORE
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-CO),MAR 21, 1976.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
167400                          $SWR=167400
000001                          $TN=1
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        11            INHIBIT ITERATIONS
                                ;*        10            BELL ON ERROR
                                ;*         9            LOOP ON ERROR
```

```
                                ;*          8          LOOP ON TEST IN SWR<7:0>
                                .SBTTL  BASIC DEFINITIONS

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

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

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

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

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
000020                          SW04=   20
000010                          SW03=   10
```

```
                        000004          SW02=   4
                        000002          SW01=   2
                        000001          SW00=   1
                                        .EQUIV  SW09,SW9
                                        .EQUIV  SW08,SW8
                                        .EQUIV  SW07,SW7
                                        .EQUIV  SW06,SW6
                                        .EQUIV  SW05,SW5
                                        .EQUIV  SW04,SW4
                                        .EQUIV  SW03,SW3
                                        .EQUIV  SW02,SW2
                                        .EQUIV  SW01,SW1
                                        .EQUIV  SW00,SW0

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

                                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                        000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                        000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                        000014          TBITVEC=14              ;;"T" BIT
                        000014          TRTVEC= 14              ;;TRACE TRAP
                        000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                        000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                        000024          PWRVEC= 24              ;;POWER FAIL
                        000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        000034          TRAPVEC=34              ;;"TRAP" TRAP
                        000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        000064          TPVEC=  64              ;;TTY PRINTER VECTOR
                        000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        164000          ABASE=  164000          ;BASE NC11 BUS ADRS EQUATE
```

```
                        160270                  AVECT1= 160270          ;BASE NC11 VECTOR ADRS EQUATE - TOP 3 BITS=PRIORITY
                        000340                  APRIOR= 340             ;NC11 PRIORITY LEVEL EQUATE
                        000001                  ADEVM=  1               ;DEFAULT TO ONE NC11
                                                .SBTTL  TRAP CATCHER


                        000000                          .=0
                                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                        000174                          .=174
000174  000000                                 DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                                 SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  001614                                  JMP     @#START2        ;;JUMP TO STARTING ADDRESS OF PROGRAM
                        000204                          .=204
000204  000137  001604                                  JMP     @#START1        ;WILL SELECT TIMING MARK & JOYSTICK TESTS
```

```
                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
000210                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  012646                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                  .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
000210                                  .=$SVPC                 ;; RESTORE PC
001000                                  .=1000
                                .SBTTL  APT PARAMETER BLOCK

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

001000                          $APTHD:
001000  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001174                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000012                  $TSTM:  .WORD   10.     ;;RUN TIM OF LONGEST TEST
001006  000036                  $PASTM: .WORD   30.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000000                  $UNITM: .WORD   0       ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000031                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=:100
001100                  $CM AG:                         ;;START OF COMMON TAGS
001100  000000                  .WORD   0
001102  000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002             $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000             $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001162  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001164  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001170  077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
001171  015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001172  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;*****************************************************************
                                .EVEN
001174                  $MAIL:                          ;;APT MAILBOX
001174  000000          $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001176  000000          $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001200  000000          $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001202  000000          $PASS:  .WORD   APASS           ;;PASS COUNT
001204  000000          $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001206  000000          $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001210  000000          $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
```

```
597  001212  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
598  001214                          $ETABLE:                ;;APT ENVIRONMENT TABLE
599  001214      000                 $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
600  001215      000                 $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
601  001216  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
602  001220  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
603  001222  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
604                                  ;*                      BITS 15-11=CPU TYPE
605                                  ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
606                                  ;*                              11/70=06,PDQ=07,Q=10
607                                  ;*                      BIT 10=REAL TIME CLOCK
608                                  ;*                      BIT  9=FLOATING POINT PROCESSOR
609                                  ;*                      BIT  8=MEMORY MANAGEMENT
610  001224      000                 $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
611  001225      000                 $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
612                                  ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
613                                  ;*                              900 NSEC CORE=001
614                                  ;*                              300 NSEC BIPOLAR=002
615                                  ;*                              500 NSEC MOS=003
616  001226  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
617                                  ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
618  001230      000                 $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
619  001231      000                 $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
620  001232  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
621  001234      000                 $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
622  001235      000                 $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
623  001236  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
624  001240      000                 $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
625  001241      000                 $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
626  001242  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
627  001244  160270                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
628  001246  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
629  001250  164000                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
630  001252  000001                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
631  001254  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
632  001256                          $ETEND:
633                                  .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


  001256                        $ERRTB:
```

```
                                ;ERROR  1
001256  070035                          EM1             ;NC11 BUS TIMEOUT ER
001260  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001262  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001264  000000                          0

                                ;ERROR  2
001266  070054                          EM2             ;REG READ/WRITE ER
001270  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001272  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001274  000000                          0

                                ;ERROR  3
001276  070076                          EM3             ;BUS RESET CLR ER
001300  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001302  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001304  000000                          0

                                ;ERROR  4
001306  070117                          EM4             ;SF CLR Z ER
001310  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001312  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001314  000000                          0

                                ;ERROR  5
001316  070133                          EM5             ;SF CLR ALL ER
001320  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001322  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001324  000000                          0

                                ;ERROR  6
001326  070151                          EM6             ;Z INC ER ON ADC START
001330  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001332  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001334  000000                          0

                                ;ERROR  7
001336  070177                          EM7             ;Z OVFLO FLAG ER
001340  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001342  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001344  000000                          0

                                ;ERROR  10
001346  070217                          EM10            ;Z OVFLO INTR ER
001350  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001352  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001354  000000                          0
```

```
 697
 698  001356  070237                  :ERROR  11
 699  001360  070714                          EM11            ;INTR AT WRONG BR LEVEL
 700  001362  071124                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
 701  001364  000000                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
 702                                          0
 703                                  :ERROR  12
 704  001366  070266                          EM12            ;LIST MD NPR DATA ER
 705  001370  071027                          DH3             ;ERRPC  TSTNUM  XYHOLD  ADRS    GDDAT   BDDAT
 706  001372  071154                          DT3             ;$ERRPC TSTNUM  XYHOLD  $BDADR  $GDDAT  $BDDAT
 707  001374  000000                          0
 708
 709                                  :ERROR  13
 710  001376  070312                          EM13            ;LIST MD CA INC ER
 711  001400  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
 712  001402  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
 713  001404  000000                          0
 714
 715                                  :ERROR  14
 716  001406  070334                          EM14            ;LIST MD WC INC ER
 717  001410  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
 718  001412  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
 719  001414  000000                          0
 720
 721                                  :ERROR  15
 722  001416  070356                          EM15            ;WCO FLAG OR INTR ER
 723  001420  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
 724  001422  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
 725  001424  000000                          0
 726
 727                                  :ERROR  15
 728  001426  070402                          EM16            ;TIMEOUT FLAG ER
 729  001430  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
 730  001432  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
 731  001434  000000                          0
 732
 733                                  ;ERROR  17
 734  001436  070422                          EM17            ;ADRS MAKE ER
 735  001440  070761                          DH2             ;ERRPC  TSTNUM  XYHOLD  GDADR   BDADR
 736  001442  071140                          DT2             ;$ERRPC TSTNUM  XYHOLD  $GDADR  $BDADR
 737  001444  000000                          0
 738
 739                                  ;ERROR  20
 740  001446  070437                          EM20            ;JOYSTICK MD FAILED TO OVERRIDE LIST MD
 741  001450  070761                          DH2             ;ERRPC  TSTNUM  XYHOLD  GDADR   BDADR
 742  001452  071140                          DT2             ;$ERRPC TSTNUM  XYHOLD  $GDADR  $BDADR
 743  001454  000000                          0
 744
```

```
                                ;ERROR  21
001456  070506                          EM21            ;NPR INC ER
001460  071027                          DH3             ;ERRPC  TSTNUM  XYHOLD  ADRS    GDDAT   BDDAT
001462  071154                          DT3             ;$ERRPC TSTNUM  XYHOLD  $BDADR  $GDDAT  $BDDAT
001464  000000                          0

                                ;ERROR  22
001466  070521                          EM22            ;INC OVFLO FLAG OR INTR ER
001470  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001472  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001474  000000                          0

                                ;ERROR  23
001476  070553                          EM23            ;WORD INC ER
001500  071027                          DH3             ;ERRPC  TSTNUM  XYHOLD  ADRS    GDDAT   BDDAT
001502  071154                          DT3             ;$ERRPC TSTNUM  XYHOLD  $BDADR  $GDDAT  $BDDAT
001504  000000                          0

                                ;ERROR  24
001506  070567                          EM24            ;BYTE INC ER
001510  071027                          DH3             ;ERRPC  TSTNUM  XYHOLD  ADRS    GDDAT   BDDAT
001512  071154                          DT3             ;$ERRPC TSTNUM  XYHOLD  $BDADR  $GDDAT  $BDDAT
001514  000000                          0

                                ;ERROR  25
001516  070603                          EM25            ;ADC OR JOYSTICK LOGIC ER
001520  071105                          DH4             ;ERRPC  TSTNUM
001522  071172                          DT4             ;$ERRPC TSTNUM
001524  000000                          0

                                ;ERROR  26
001526  070634                          EM26            ;TIMING MARK FAILED ON ADRS MAKE
001530  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001532  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001534  000000                          0

                                ;ERROR  27
001536  070674                          EM27            ;NO CONVERT FAILED
001540  070714                          DH1             ;ERRPC  TSTNUM  BUSADR  EXPCT   RCVD
001542  071124                          DT1             ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT
001544  000000                          0
```

```
                                        ;NC11 BUS REGISTER ADDRESS POINTERS
                                        
001546  164000                          NCCSR:  164000          ;COMMAND/STATUS
001550  164002                          NCOFF:  164002          ;OFFSET ADRS
001552                                  NCADR:                  ;READ ADRS MAKE OR WRITE LIST MD NPR DATA
001552  164004                          NCXYH:  164004          ;READ X/Y ADC OR WRITE X/Y ADRS
001554  164006                          NCZHWC: 164006          ;HIGH ORDER Z OR WORD COUNT
001556  164010                          NCZLCA: 164010          ;LOW ORDER Z OR CURRENT ADRS
001560  164012                          NCIREG: 164012          ;INCREMENT
001562  164014                          NCSFR:  164014          ;SPECIAL FUNCTION (SF)
                                        
                                        ;NC11 VECTOR ADDRESS POINTERS
                                        
001564  000270                          NCVCT0: 270             ;CELL OVERFLOW INTR VECTOR PTR
001566  000272                          NCVCT2: 272
001570  000274                          NCVCT4: 274             ;Z OVERFLOW OR WC OVERFLOW VECTOR PTR
001572  000276                          NCVCT6: 276
                                        
                                        ;NC11 DEVICE LEVEL(S)
                                        
001574  000340                          NCBRL:  340             ;DEVICE LEVEL
001576  000300                          NCBRL1: 300             ;DEVICE LEVEL -1
                                        
                                        ;COMMON PROGRAM LOCATION(S)
                                        
001600  000000                          TSTNUM: 0               ;CONTAINS TEST NUMBER ON ERROR
001602  000000                          STST54: 0               ;0 SAYS DO LOGIC TESTS, -1 SAYS DO TIMING MARK & JOYSTICK TESTS
```

```
                                        .SBTTL  PROGRAM START
001604  012737  177777  001602  START1: MOV     #-1,STST54      ;SELECT TIMING MARK & JOYSTICK TESTS
001612  000402                          BR      START           ;START
001614  005037  001602          START2: CLR     STST54          ;SELECT LOGIC TESTS
001620                          START:
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001620  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001624  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001626  022706  001140                  CMP     #SWR,R6 ;;DONE?
001632  001374                          BNE     .-6             ;;LOOP BACK IF NO
001634  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001640  012737  016150  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
001646  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
001654  012737  015606  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
001662  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
001670  012737  017412  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001676  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001704  012737  017206  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001712  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
001720  005037  001160                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
001724  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
001730  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
001736  012737  001736  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
001744  012737  001744  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001752  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001756  012737  002012  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001764  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001772  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002000  022777  177777  177132          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002006  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
002010  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
002012  012716  002020          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002016  000002                          RTI
002020  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002026  012737  000174  001142          MOV     #DISPREG,DISPLAY
002034  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002040  005037  001202                  CLR     $PASS           ;;CLEAR PASS COUNT
002044  132737  000200  001215          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002052  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
002054  012737  001216  001140          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002062                          67$:
002062  013737  001244  001574  SETUP1: MOV     $VECT1,NCBRL    ;GET PRIORITY LEVEL
002070  042737  017777  001574          BIC     #17777,NCBRL    ;RID VECTOR ADRS
002076  000337  001574                  SWAB    NCBRL           ;ADJUST FOR PSW
002102  013737  001574  001576          MOV     NCBRL,NCBRL1    ;PREPARE DEVICE LEVEL -1
002110  162737  000040  001576          SUB     #40,NCBRL1      ;DO IT
002116  012700  001546                  MOV     #NCCSR,R0       ;SET UP REG ADRS POINTERS
002122  013701  001250                  MOV     $BASE,R1        ;GET BASE ADRS
002126  010120          SETUP2: MOV     R1,(R0)+        ;
002130  062701  000002                  ADD     #2,R1           ;
```

```
                                                CMP     #NCSFR+2,R0     ;ALL DONE?
                                                BNE     SETUP2          ;BR IF NOT
                                                MOV     #NCVCT0,R0      ;SET UP NC11 VECTOR ADRS POINTERS
                                                MOV     $VECT1,R1       ;GET BASE VECTOR ADRS
                                                BIC     #160000,R1      ;GET RID OF PRIORITY LEVEL
                                        SETUP3: MOV     R1,(R0)+        ;
                                                ADD     #2,R1           ;
                                                CMP     #NCVCT6+2,R0    ;ALL DONE?
                                                BNE     SETUP3          ;BR IF NOT
                                        .SBTTL  TYPE PROGRAM NAME
                                        ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
                                                INC     #-1             ;;FIRST TIME?
                                                BNE     64$             ;;BRANCH IF NO
                                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                                                TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
                                                BNE     66$             ;;BRANCH IF YES
                                                CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
                                                BEQ     66$             ;;BRANCH IF YES
                                                CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
                                                BNE     67$             ;;BRANCH IF NO
                                                GTSWR                   ;;GET SOFT-SWR SETTINGS
                                                BR      67$
                                        66$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
                                        67$:
                                                BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>#MD11-DZNCA-C  NC11-A LOGIC TEST#<CRLF>
                                        64$:
                                        RESTRT: MOV     #PR7,PSW        ;SET PRIORITY TO HIGHEST LEVEL
                                                MOV     #STACK,SP       ;ALWAYS RESET STACK PTR
                                                RESET                   ;INITIALIZE NC11 BEFORE TESTING
                                                TST     STST54          ;GO TO THE MANUAL INTERVENTION TESTS?
                                                BEQ     1$              ;BR IF NOT
                                                MOVB    #54,$TSTNM      ;SET UP STARTING TEST # 54 FOR THESE TSTS
                                                JMP     @#TEST54        ;GO TO MANUAL TESTS - NC11 JOYSTICK REQUIRED
                                        1$:
```

```
                                        ;;*****************************************************************
                                        ;*TEST 1         TEST THAT ALL NC11 REGS ARE ACCESSIBLE
                                        ;;*****************************************************************
002344  000240                  TST1:   <NOP>
002346  012737  002362  001106          MOV     #10$,$LPADR     ;;SET SCOPE LOOP ADDRESS
002354  012737  000001  001200          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
002362  112737  000001  001102  10$:     MOVB    #1,$TSTNM       ;SET TEST # TO 1
002370  012737  002424  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
002376  005037  001124                  CLR     $GDDAT          ;NO DATA COMPARE
002402  005037  001126                  CLR     $BDDAT          ;NO DATA COMPARE
002406  012737  002442  000004          MOV     #2$,@#ERRVEC    ;SET UP TIMEOUT RETURN ADRS
002414  013700  001546                  MOV     NCCSR,R0        ;SET UP 1ST NC11 BUS ADRS
002420  012701  000007                  MOV     #7,R1           ;SET UP REG COUNT
002424  010037  001122          1$:     MOV     R0,$BDADR       ;SET UP CURRENT NC BUS ADRS
002430  005710                          TST     (R0)            ;SEE IF THERE
002432  005720                          TST     (R0)+           ;BUMP TO NEXT
002434  005301                          DEC     R1              ;COUNT 7 OF THEM
002436  001403                          BEQ     3$              ;BR IF ALL DONE
002440  000771                          BR      1$              ;TRY NEXT
002442  022626                  2$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
002444  104001                          ERROR   1               ;BUS ADRS INDICATED DID NOT RESPOND
002446  012737  000006  000004  3$:     MOV     #ERRVEC+2,@#ERRVEC      ;RESTORE LOC 4

                                        ;;*****************************************************************
                                        ;*TEST 2         TEST THAT COMMAND/STATUS REG IS WRITE/READABLE (COUNT PATTERN)
                                        ;;*****************************************************************
002454  000004                  TST2:   SCOPE
002456  012737  000004  001160          MOV     #4,$TIMES       ;;DO 4 ITERATIONS
002464  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET UP CSR ADRS
002472  012737  002504  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
002500  012700  170000                  MOV     #170000,R0      ;START AT 0 - HI BITS FOR NOISE
002504  010037  001124          1$:     MOV     R0,$GDDAT       ;LD EXPECTED
002510  042737  170200  001124          BIC     #170200,$GDDAT  ;MASK TO WRITEABLE BITS
002516  010077  177024                  MOV     R0,@NCCSR       ;SEND PATTERN OUT
002522  017737  177020  001126          MOV     @NCCSR,$BDDAT   ;READ IT BACK
002530  042737  170200  001126          BIC     #170200,$BDDAT  ;MASK OUT UNKNOWNS
002536  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
002544  001401                          BEQ     2$              ;BR IF SO
002546  104002                          ERROR   2               ;NCCSR WRITE (READ) FAILURE
002550  005200                  2$:     INC     R0              ;ADVANCE PATTERN
002552  001354                          BNE     1$              ;REPEAT NX PATTERN IF NOT ALL TESTED

                                        ;;*****************************************************************
                                        ;*TEST 3         TEST THAT "RESET" WILL CLEAR THE COMMAND/STATUS REG
                                        ;;*****************************************************************
002554  000004                  TST3:   SCOPE
002556  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
002564  012777  000102  176770          MOV     #102,@NCSFR     ;CLR ALL
002572  013737  001546  001122          MOV     NCCSR,$BDADR    ;SETUP CSR ADRS
002600  005037  001124                  CLR     $GDDAT          ;LD EXPECTED
002604  012777  007577  176734          MOV     #7577,@NCCSR    ;LOAD ALL WRITEABLE BITS
002612  000005                          RESET                   ;CLR CSR
002614  017737  176726  001126          MOV     @NCCSR,$BDDAT   ;READ IT
002622  001401                          BEQ     TST4            ;;NEXT TEST IF CLEARED
002624  104003                          ERROR   3               ;RESET FAILED TO CLR CSR REG
```

```
                                        ;;*****************************************************
                                        ;*TEST 4        TEST THAT THE OFFSET REG IS WRITE/READABLE (COUNT PATTERN)
                                        ;;*****************************************************
002626  000004                  TST4:   SCOPE
002630  012737  000004  001160          MOV     #4,$TIMES       ;;DO 4 ITERATIONS
002636  013737  001550  001122          MOV     NCOFF,$BDADR    ;SET UP OFFSET REG ADRS
002644  012737  002654  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
002652  005000                        CLR     R0              ;START WITH ALL BITS ZERO
002654  010037  001124          1$:     MOV     R0,$GDDAT       ;LD EXPECTED
002660  042737  001774  001124          BIC     #1774,$GDDAT    ;MASK TO WRITEABLE BITS
002666  010077  176656                  MOV     R0,@NCOFF               ;SEND PATRN OUT
002672  017737  176652  001126          MOV     @NCOFF,$BDDAT   ;READ IT BACK
002700  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
002706  001401                        BEQ     2$              ;BR IF SO
002710  104002                        ERROR   2               ;OFFSET REG WRITE/READ FAILURE
002712  062700  002001          2$:     ADD     #2001,R0        ;ADVANCE PATTERN
002716  103356                        BCC     1$              ;REPEAT NX PATTERN IF NOT ALL TESTED

                                        ;;*****************************************************
                                        ;*TEST 5        TEST THAT "RESET" WILL CLEAR THE OFFSET REG
                                        ;;*****************************************************
002720  000004                  TST5:   SCOPE
002722  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
002730  013737  001550  001122          MOV     NCOFF,$BDADR    ;SET LD OFFSET REG ADRS
002736  005037  001124                  CLR     $GDDAT          ;LD EXPECTED
002742  012777  174003  176600          MOV     #174003,@NCOFF  ;LOAD ALL WRITEABLE BITS
002750  000005                        RESET                   ;CLR OFFSET REG
002752  017737  176572  001126          MOV     @NCOFF,$BDDAT   ;READ IT
002760  001401                        BEQ     TST6            ;;NEXT TEST IF CLEARED
002762  104003                        ERROR   3               ;RESET FAILED TO CLR HI OFFSET REG

                                        ;;*****************************************************
                                        ;*TEST 6        TEST THAT THE HIGH ORDER Z REG IS WRITE/READABLE (COUNT PATTERN)
                                        ;;*****************************************************
002764  000004                  TST6:   SCOPE
002766  012737  000004  001160          MOV     #4,$TIMES       ;;DO 4 ITERATIONS
002774  013737  001554  001122          MOV     NCZHWC,$BDADR   ;SET UP Z HI REG ADRS
003002  012737  003016  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
003010  012737  177777  001124          MOV     #-1,$GDDAT      ;LD EXPECTED - START WITH ALL ONES
003016  013777  001124  176530  1$:     MOV     $GDDAT,@NCZHWC  ;LD Z REG HI
003024  017737  176524  001126          MOV     @NCZHWC,$BDDAT  ;READ IT BACK
003032  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
003040  001401                        BEQ     2$              ;BR IF SO
003042  104002                        ERROR   2               ;Z HI REG WRITE/READ FAILURE
003044  005337  001124          2$:     DEC     $GDDAT          ;DECREASE COUNT BY ONE
003050  102362                        BVC     1$              ;AGAIN TILL ALL PATTERNS DONE

                                        ;;*****************************************************
                                        ;*TEST 7        TEST THAT THE LOW ORDER Z REG IS WRITE/READABLE (COUNT PATTERN)
                                        ;;*****************************************************
003052  000004                  TST7:   SCOPE
003054  012737  000004  001160          MOV     #4,$TIMES       ;;DO 4 ITERATIONS
003062  013737  001556  001122          MOV     NCZLCA,$BDADR   ;SET UP Z HI REG ADRS
003070  012737  003104  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
003076  012737  177777  001124          MOV     #-1,$GDDAT      ;LD EXPECTED - START WITH ALL ONES
003104  013777  001124  176444  1$:     MOV     $GDDAT,@NCZLCA  ;LD Z REG LOW
```

```
1018  003112  017737  176440  001126          MOV     @NCZLCA,$BDDAT  ;READ IT BACK
1019  003120  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1020  003126  001401                          BEQ     2$              ;BR IF SO
1021  003130  104002                          ERROR   2               ;Z LOW REG WRITE/READ FAILURE
1022  003132  005337  001124          2$:     DEC     $GDDAT          ;DECREASE COUNT BY ONE
1023  003136  102362                          BVC     1$              ;AGAIN TILL ALL PATTERNS DONE
1024
1025                                  ;;*****************************************************************
1026                                  ;*TEST 10       TEST THAT "RESET" WILL CLEAR LOW/HIGH ORDER Z REGS
1027                                  ;;*****************************************************************
1028  003140  000004                  TST10:  SCOPE
1029  003142  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1030  003150  005037  001124                  CLR     $GDDAT          ;LD EXPECTED
1031  003154  012777  177777  176374          MOV     #-1,@NCZLCA     ;SET ALL BITS - Z LOW
1032  003162  012777  177777  176364          MOV     #-1,@NCZHWC     ;SET ALL BITS - Z HIGH
1033  003170  000005                          RESET                   ;DO A BUS RESET
1034  003172  017737  176360  001126          MOV     @NCZLCA,$BDDAT  ;READ Z LOW
1035  003200  001404                          BEQ     1$              ;BR IF CLRED
1036  003202  013737  001556  001122          MOV     NCZLCA,$BDADR   ;SET UP Z LOW REG ADRS
1037  003210  104003                          ERROR   3               ;RESET FAILED TO CLEAR Z LOW REG
1038  003212  017737  176336  001126  1$:     MOV     @NCZHWC,$BDDAT  ;READ Z HIGH
1039  003220  001404                          BEQ     TST11           ;;NEXT TEST IF CLRED
1040  003222  013737  001554  001122          MOV     NCZHWC,$BDADR   ;SET UP Z HIGH REG ADRS
1041  003230  104003                          ERROR   3               ;RESET FAILED TO CLEAR Z HIGH REG
1042
1043                                  ;;*****************************************************************
1044                                  ;*TEST 11       TEST THAT THE I REG IS WRITE/READABLE (COUNT PATTERN)
1045                                  ;;*****************************************************************
1046  003232  000004                  TST11:  SCOPE
1047  003234  012737  000004  001160          MOV     #4,$TIMES       ;;DO 4 ITERATIONS
1048  003242  013737  001560  001122          MOV     NCIREG,$BDADR   ;SET UP I REG ADRS
1049  003250  012737  003264  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
1050  003256  012737  177777  001124          MOV     #-1,$GDDAT      ;LD EXPECTED - START WITH ALL ONES
1051  003264  013777  001124  176266  1$:     MOV     $GDDAT,@NCIREG  ;LD I REG
1052  003272  017737  176262  001126          MOV     @NCIREG,$BDDAT  ;READ IT BACK
1053  003300  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1054  003306  001401                          BEQ     2$              ;BR IF SO
1055  003310  104002                          ERROR   2               ;I REG WRITE/READ FAILURE
1056  003312  005337  001124          2$:     DEC     $GDDAT          ;DECREASE COUNT BY ONE
1057  003316  103762                          BCS     1$              ;AGAIN TILL ALL PATTERNS DONE
1058
1059                                  ;;*****************************************************************
1060                                  ;*TEST 12       TEST THAT "RESET" WILL CLEAR I REG
1061                                  ;;*****************************************************************
1062  003320  000004                  TST12:  SCOPE
1063  003322  012737  000010  001160          MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1064  003330  013737  001560  001122          MOV     NCIREG,$BDADR   ;SET UP I REG ADRS
1065  003336  005037  001124                  CLR     $GDDAT          ;LD EXPECTED
1066  003342  012777  177777  176210          MOV     #-1,@NCIREG     ;SET ALL BITS
1067  003350  000005                          RESET                   ;CLR I REG
1068  003352  017737  176202  001126          MOV     @NCIREG,$BDDAT  ;READ I REG
1069  003360  001401                          BEQ     TST13           ;;NEXT TEST IF CLEARED
1070  003362  104003                          ERROR   3               ;RESET FAILED TO CLR I REG
1071
1072                                  ;;*****************************************************************
1073                                  ;*TEST 13       TEST SPECIAL FUNCTION "CLEAR Z/WC & Z/CA"
```

```
1074                                        ;;*****************************************************************
1075  003364  000004                  TST13:  SCOPE
1076  003366  005037  001124                  CLR     $GDDAT          ;LD EXPECTED
1077  003372  012777  177777  176154          MOV     #-1,@NCZHWC     ;SET ALL BITS - Z/WC
1078  003400  012777  177777  176150          MOV     #-1,@NCZLCA     ;SET ALL BITS - Z/CA
1079  003406  012777  000020  176146          MOV     #20,@NCSFR      ;SEND SPECIAL FUNCT CLR
1080  003414  017737  176134  001126          MOV     @NCZHWC,$BDDAT  ;READ Z/WC
1081  003422  001404                          BEQ     1$              ;BR IF CLRED
1082  003424  013737  001554  001122          MOV     NCZHWC,$BDADR   ;SET UP Z/WC BUS ADRS
1083  003432  104004                          ERROR   4               ;SPECIAL FUNCTION CLR Z/WC FAILED
1084  003434  017737  176116  001126  1$:     MOV     @NCZLCA,$BDDAT  ;READ Z/CA
1085  003442  001404                          BEQ     TST14           ;;NEXT TEST IF CLEARED
1086  003444  013737  001556  001122          MOV     NCZLCA,$BDADR   ;SET UP Z/CA BUS ADRS
1087  003452  104004                          ERROR   4               ;SPECIAL FUNCTION CLR Z/CA FAILED
1088
1089                                        ;;*****************************************************************
1090                                        ;*TEST 14        TEST SPECIAL FUNCTION "CLEAR ALL" (USE I REG)
1091                                        ;;*****************************************************************
1092  003454  000004                  TST14:  SCOPE
1093  003456  013737  001560  001122          MOV     NCIREG,$BDADR   ;SET UP I REG ADRS
1094  003464  005037  001124                  CLR     $GDDAT          ;LD EXPECTED
1095  003470  012777  177777  176062          MOV     #-1,@NCIREG     ;SET ALL BITS
1096  003476  012777  000100  176056          MOV     #100,@NCSFR     ;SEND SPECIAL FUNCTION "CLR ALL"
1097  003504  017737  176050  001126          MOV     @NCIREG,$BDDAT  ;READ I REG
1098  003512  001401                          BEQ     TST15           ;;NEXT TEST IF CLEARED
1099  003514  104005                          ERROR   5               ;SPECIAL FUNCTION FAILED TO "CLEAR ALL" I REG
1100
1101                                        ;;*****************************************************************
1102                                        ;*TEST 15        TEST THAT Z LOW INCREMENTS ON SPECIAL FUNCT CONVERT ADC'S
1103                                        ;;*****************************************************************
1104  003516  000004                  TST15:  SCOPE
1105  003520  012737  000004  001160          MOV     #4,$TIMES       ;;DO 4 ITERATIONS
1106  003526  012737  003546  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
1107  003534  012700  000001                  MOV     #1,R0           ;LD EXPECTED IN R0
1108  003540  012777  000020  176014          MOV     #20,@NCSFR      ;SPECIAL FUNCTION CLR Z
1109  003546  012777  000004  176006  1$:     MOV     #4,@NCSFR       ;START ADC'S
1110  003554  013737  001556  001122          MOV     NCZLCA,$BDADR   ;SET UP Z/CA REG ADRS
1111  003562  010037  001124                  MOV     R0,$GDDAT       ;LD EXPECTED
1112  003566  017737  175764  001126          MOV     @NCZLCA,$BDDAT  ;READ Z REG LO
1113  003574  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1114  003602  001401                          BEQ     2$              ;BR IF SO
1115  003604  104006                          ERROR   6               ;Z LOW FAILED TO INCREMENT ON ADC START
1116  003606  013737  001554  001122  2$:     MOV     NCZHWC,$BDADR   ;SET UP Z/WC REG ADRS
1117  003614  005037  001124                  CLR     $GDDAT          ;LD EXPECTED
1118  003620  017737  175730  001126          MOV     @NCZHWC,$BDDAT  ;SEE THAT HI ORDER Z REG IS 0
1119  003626  001401                          BEQ     3$              ;BR IF SO
1120  003630  104006                          ERROR   6               ;HI ORDER Z SHOULD NOT HAVE INCREMENTED
1121  003632  005200                  3$:     INC     R0              ;SET UP NEXT EXPECTED COUNT
1122  003634  001344                          BNE     1$              ;BR IF LOW Z NOT EQ TO 177777
1123
1124                                        ;;*****************************************************************
1125                                        ;*TEST 16        TEST THAT Z HIGH INCREMENTS ON SPECIAL FUNCT. CONVERT ADC'S
1126                                        ;;*****************************************************************
1127  003636  000004                  TST16:  SCOPE
1128  003640  012737  000004  001160          MOV     #4,$TIMES       ;;DO 4 ITERATIONS
1129  003646  012737  003666  001110          MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
```

```
1130  003654  012700  000001                    MOV     #1,R0           ;LD EXPECTED IN R0
1131  003660  012777  000020  175674            MOV     #20,@NCSFR      ;SPECIAL FUNCTION CLR Z
1132  003666  012777  177777  175662  1$:       MOV     #-1,@NCZLCA     ;ALWAYS SET LO Z FOR CARRY TO HI Z
1133  003674  012777  000004  175660            MOV     #4,@NCSFR       ;START ADC'S
1134  003702  013737  001554  001122            MOV     NCZHWC,$BDADR   ;SET UP Z/WC REG ADRS
1135  003710  010037  001124                    MOV     R0,$GDDAT       ;LD EXPECTED
1136  003714  017737  175634  001126            MOV     @NCZHWC,$BDDAT  ;READ Z REG HI
1137  003722  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
1138  003730  001401                            BEQ     2$              ;BR IF SO
1139  003732  104006                            ERROR   6               ;LO ORDER Z FAILED TO INCREMENT TO HI Z
1140  003734  013737  001556  001122  2$:       MOV     NCZLCA,$BDADR   ;SET UP Z/CA REG ADRS
1141  003742  005037  001124                    CLR     $GDDAT          ;LD EXPECTED
1142  003746  017737  175604  001126            MOV     @NCZLCA,$BDDAT  ;SEE THAT LO ORDER Z REG WENT TO 0
1143  003754  001401                            BEQ     3$              ;BR IF SO
1144  003756  104006                            ERROR   6               ;LO ORDER Z SHOULD HAVE INCREMENTED TO 0
1145  003760  005200                    3$:     INC     R0              ;SET UP NEXT EXPECTED COUNT
1146  003762  001341                            BNE     1$              ;BR IF HI Z NOT EQ TO 177777
1147
1148                                    ;;**************************************************************
1149                                    ;*TEST 17        TEST THAT Z OVFLO FLAG WILL SET THEN CLR WITH SPECIAL FUNCT CLR Z
1150                                    ;;**************************************************************
1151  003764  000004                    TST17:  SCOPE
1152
1153  003766  012777  000020  175566            MOV     #20,@NCSFR      ;SPECIAL FUNCTION CLR Z/FLG
1154  003774  012777  177777  175554            MOV     #-1,@NCZLCA     ;LOAD ALL BITS LO Z
1155  004002  012777  177777  175544            MOV     #-1,@NCZHWC     ;LOAD ALL BITS H Z
1156  004010  012777  000004  175544            MOV     #4,@NCSFR       ;START ADC'S
1157  004016  013737  001546  001122            MOV     NCCSR,$BDADR    ;SET UP CSR REG ADRS
1158  004024  012737  010000  001124            MOV     #BIT12,$GDDAT   ;LD EXPECTED
1159  004032  032777  010000  175506            BIT     #BIT12,@NCCSR   ;LOOK FOR Z OVERFLOW FLAG
1160  004040  001003                            BNE     1$              ;BR IF SET
1161  004042  005037  001126                    CLR     $BDDAT          ;SHOW IT WAS NOT SET
1162
1163  004046  104007                            ERROR   7               ;Z OVERFLOW FLAG FAILED TO SET
1164  004050  005037  001124            1$:     CLR     $GDDAT          ;LD EXPECTED
1165  004054  012777  000020  175500            MOV     #20,@NCSFR      ;SPECIAL FUNCTION CLR Z FLAG
1166  004062  032777  010000  175456            BIT     #BIT12,@NCCSR   ;DID Z OVERFLOW FLAG CLR?
1167  004070  001404                            BEQ     TST20           ;;NEXT TEST IF SO
1168  004072  012737  010000  001126            MOV     #BIT12,$BDDAT   ;SHOW IT IS STILL SET
1169  004100  104007                            ERROR   7               ;SPECIAL FUNCTION CLR Z FLG FAILURE
1170
1171                                    ;;**************************************************************
1172                                    ;*TEST 20        TEST THAT RESET WILL CLEAR Z OVERFLOW FLAG
1173                                    ;;**************************************************************
1174  004102  000004                    TST20:  SCOPE
1175  004104  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
1176  004112  012777  000020  175442            MOV     #20,@NCSFR      ;SPECIAL FUNCTION CLR Z/FLAG
1177  004120  012777  177777  175430            MOV     #-1,@NCZLCA     ;LOAD ALL BITS LO Z
1178  004126  012777  000004  175426            MOV     #4,@NCSFR       ;START ADC'S
1179  004134  013737  001546  001122            MOV     NCCSR,$BDADR    ;SET UP CSR REG ADRS
1180  004142  005037  001124                    CLR     $GDDAT          ;LD EXPECTED
1181  004146  000005                            RESET                   ;I/O CLR
1182  004150  032777  010000  175370            BIT     #BIT12,@NCCSR   ;DID Z OVERFLOW FLAG CLR?
1183  004156  001404                            BEQ     TST21           ;;NEXT TEST IF SO
1184  004160  012737  010000  001126            MOV     #BIT12,$BDDAT   ;SHOW IT IS STILL SET
1185  004166  104003                            ERROR   3               ;RESET FAILED TO CLEAR Z OVERFLOW FLAG
```

```
                                        ;;*****************************************************************
                                        ;*TEST 21       TEST THAT Z OVERFLOW WILL CAUSE AN INTERRUPT AT CHOSEN BR LEVEL -1
                                        ;;*****************************************************************
004170  000004                  TST21:  SCOPE
004172  012737  000340  177776          MOV     #PR7,PSW        ;SET CPU PRIORITY TO HIGHEST
004200  012777  000100  175354          MOV     #100,@NCSFR     ;CLR ALL
004206  012777  004306  175354          MOV     #1$,@NCVCT4     ;SET UP INTR RETURN ADRS
004214  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET UP CSR ADRS
004222  012777  000340  175342          MOV     #PR7,@NCVCT6    ;SET UP INTR PRIORITY
004230  012777  177777  175320          MOV     #-1,@NCZLCA     ;SET ALL BITS Z LO
004236  012777  177777  175310          MOV     #-1,@NCZHWC     ;SET ALL BITS Z HI
004244  012777  004000  175274          MOV     #BIT11,@NCCSR   ;ENABLE Z OVERFLOW INTR
004252  012777  000004  175302          MOV     #4,@NCSFR       ;START ADC'S
004260  013737  001576  177776          MOV     NCBRL1,PSW      ;ALLOW INTR
004266  017737  175254  001126          MOV     @NCCSR,$BDDAT   ;READ CSR
004274  042737  167777  001126          BIC     #167777,$BDDAT  ;SAVE ONLY Z OVERFLOW FLAG
004302  104010                          ERROR   10              ;Z OVERFLOW FAILED TO INTERRUPT
004304  000410                          BR      2$              ;GO RESTORE INTR VECTOR
004306  022626                  1$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
004310  032777  010000  175230          BIT     #BIT12,@NCCSR   ;MAKE SURE Z OVERFLOW FLAG IS SET
004316  001003                          BNE     2$              ;BR IF SO
004320  005037  001126                  CLR     $BDDAT          ;SHOW IT WAS NOT SET
004324  104010                          ERROR   10              ;Z OVERFLOW INTERRUPTED BUT NO FLAG
004326  013777  001572  175234  2$:     MOV     NCVCT6,@NCVCT4  ;RESTORE VECTOR PTR TO HALT
004334  005077  175232                  CLR     @NCVCT6         ;LOAD HALT

                                        ;;*****************************************************************
                                        ;*TEST 22       TEST THAT NC11 DOES NOT INTR AT HIGHER BR LEVELS (USE Z OVFLO)
                                        ;;*****************************************************************
004340  000004                  TST22:  SCOPE
004342  012737  000340  177776          MOV     #PR7,PSW        ;SET CPU PRIORITY TO HIGHEST
004350  012777  004464  175212          MOV     #3$,@NCVCT4     ;SET UP INTR RETURN ADRS
004356  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET UP CSR ADRS
004364  012777  000340  175200          MOV     #PR7,@NCVCT6    ;SET UP INTR PRIORITY
004372  012700  000340                  MOV     #PR7,R0         ;GET TOP BR LEVEL
004376  012777  000100  175156  1$:     MOV     #100,@NCSFR     ;CLR ALL
004404  012777  177777  175144          MOV     #-1,@NCZLCA     ;SET ALL BITS Z LO
004412  012777  177777  175134          MOV     #-1,@NCZHWC     ;SET ALL BITS Z HI
004420  012777  004000  175120          MOV     #BIT11,@NCCSR   ;ENABLE Z OVERFLOW INTR
004426  012737  010000  001124          MOV     #BIT12,$GDDAT   ;LD EXPECTED
004434  012777  000004  175120          MOV     #4,@NCSFR       ;START ADC'S
004442  010037  177776          2$:     MOV     R0,PSW          ;SET UP PRIORITY
004446  021616                          CMP     (SP),(SP)       ;WASTE TIME
004450  162700  000040                  SUB     #40,R0          ;LOWER LEVEL
004454  023700  001576                  CMP     NCBRL1,R0       ;TO DEVICE LEVEL -1?
004460  001406                          BEQ     4$              ;BR IF SO
004462  000767                          BR      2$              ;TRY THIS LEVEL
004464  022626                  3$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI - SHOULDN'T HAVE INTERRUPED
004466  017737  175054  001126          MOV     @NCCSR,$BDDAT   ;READ CSR
004474  104011                          ERROR   11              ;NC11 INTERRUPTED AT DEVICE LEVEL OR HIGHER
004476  013777  001572  175064  4$:     MOV     NCVCT6,@NCVCT4  ;RESTORE VECTOR PTR TO HALT
004504  005077  175062                  CLR     @NCVCT6         ;LOAD HALT

                                        ;;*****************************************************************
                                        ;*TEST 23       TEST FOR PREMATURE Z OVERFLOW INTERRUPT FROM HIGH Z COUNT
```

```
1242                                        ;;*****************************************************************
1243  004510  000004                        TST23:  SCOPE
1244  004512  012737  000004  001160                MOV     #4,$TIMES       ;;DO 4 ITERATIONS
1245  004520  013737  001546  001122                MOV     NCCSR,$BDADR    ;SET UP CSR ADRS
1246  004526  012737  000340  177776                MOV     #PR7,PSW        ;SET CPU PRIORITY TO HIGHEST
1247  004534  012700  000001                       MOV     #1,R0           ;R0 WILL COUNT HI Z
1248  004540  012777  000100  175014                MOV     #100,@NCSFR     ;CLR ALL
1249  004546  012777  004622  175014                MOV     #2$,@NCVCT4     ;SET UP INTR RETURN ADRS
1250  004554  012777  000340  175010                MOV     #PR7,@NCVCT6    ;SET UP INTR PRIORITY
1251  004562  012777  004000  174756                MOV     #BIT11,@NCCSR   ;ENABLE Z OVERFLOW INTR
1252  004570  012777  177777  174760  1$:           MOV     #-1,@NCZLCA     ;ALWAYS SET LOW ORDER Z BITS
1253  004576  012777  000004  174756                MOV     #4,@NCSFR       ;START ADC'S
1254  004604  013737  001576  177776                MOV     NCBRL1,PSW      ;ALLOW A POTENTIAL INTR
1255  004612  021616                               CMP     (SP),(SP)       ;STALL
1256  004614  005200                               INC     R0              ;COUNT ALONG WITH HI Z REG
1257  004616  001364                               BNE     1$              ;BR IF NOT A FULL 16 BIT COUNT YET
1258  004620  000407                               BR      3$              ;GO RESTORE INTR VECTOR
1259  004622  022626                        2$:    CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI - SHOULDN'T HAVE INTR'ED
1260  004624  005037  001124                       CLR     $GDDAT          ;LD EXPECTED
1261  004630  017737  174712  001126                MOV     @NCCSR,$BDDAT   ;READ CSR
1262  004636  104010                               ERROR   10              ;Z OVERFLOW INTERRUPTED ON LESS THAN FULL COUNT
1263  004640  013777  001572  174722  3$:           MOV     NCVCT6,@NCVCT4  ;RESTORE VECTOR PTR TO HALT
1264  004646  005077  174720                       CLR     @NCVCT6         ;LOAD HALT
1265
1266                                        ;;*****************************************************************
1267                                        ;*TEST 24        TEST FOR PREMATURE Z OVERFLOW INTERRUPT FROM LOW Z COUNT
1268                                        ;;*****************************************************************
1269  004652  000004                        TST24:  SCOPE
1270  004654  012737  000004  001160                MOV     #4,$TIMES       ;;DO 4 ITERATIONS
1271  004662  013737  001546  001122                MOV     NCCSR,$BDADR    ;SET UP CSR ADRS
1272  004670  012737  000340  177776                MOV     #PR7,PSW        ;SET CPU PRIORITY TO HIGHEST
1273  004676  012700  000001                       MOV     #1,R0           ;R0 WILL COUNT LO Z
1274  004702  012777  000100  174652                MOV     #100,@NCSFR     ;CLR ALL
1275  004710  012777  004764  174652                MOV     #2$,@NCVCT4     ;SET UP INTR RETURN ADRS
1276  004716  012777  000340  174646                MOV     #PR7,@NCVCT6    ;SET UP INTR PRIORITY
1277  004724  012777  004000  174614                MOV     #BIT11,@NCCSR   ;ENABLE Z OVERFLOW INTR
1278  004732  012777  177777  174614  1$:           MOV     #-1,@NCZHWC     ;ALWAYS SET HIGH ORDER Z BITS
1279  004740  012777  000004  174614                MOV     #4,@NCSFR       ;START ADC'S
1280  004746  013737  001576  177776                MOV     NCBRL1,PSW      ;ALLOW A POTENTIAL INTR
1281  004754  021616                               CMP     (SP),(SP)       ;STALL
1282  004756  005200                               INC     R0              ;COUNT ALONG WITH LO Z REG
1283  004760  001364                               BNE     1$              ;BR IF NOT A FULL 16 BIT COUNT YET
1284  004762  000405                               BR      3$              ;GO RESTORE INTR VECTOR
1285  004764  022626                        2$:    CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI - SHOULDN'T HAVE INTR'ED
1286  004766  017737  174554  001126                MOV     @NCCSR,$BDDAT   ;READ CSR
1287  004774  104010                               ERROR   10              ;Z OVERFLOW INTERRUPTED ON LESS THAN FULL COUNT
1288  004776  013777  001572  174564  3$:           MOV     NCVCT6,@NCVCT4  ;RESTORE VECTOR PTR TO HALT
1289  005004  005077  174562                       CLR     @NCVCT6         ;LOAD HALT
1290
1291                                        ;;*****************************************************************
1292                                        ;*TEST 25        TEST A LIST MODE NPR TO ADRS 20000 ALL DATA PATTERNS
1293                                        ;;*****************************************************************
1294  005010  000004                        TST25:  SCOPE
1295  005012  012737  000004  001160                MOV     #4,$TIMES       ;;DO 4 ITERATIONS
1296  005020  012737  000340  177776                MOV     #PR7,PSW        ;DON'T WANT ANY INTR'S
1297  005026  012737  177777  001124                MOV     #-1,$GDDAT      ;LD EXPECTED
```

```
 1298  005034  012700  020000                    MOV     #20000,R0       ;SET UP WORKING ADRS IN R0
 1299  005040  010037  001122                    MOV     R0,$BDADR       ;SET UP LIST ADRS
 1300  005044  012777  000100  174510  1$:       MOV     #100,@NCSFR     ;CLR ALL
 1301  005052  012777  000405  174466            MOV     #405,@NCCSR     ;SELECT LIST MD + NPR ENABLE
 1302  005060  012777  177777  174466            MOV     #-1,@NCZHWC     ;SET UP WC IN CASE IT GETS AWAY
 1303  005066  010077  174464                    MOV     R0,@NCZLCA      ;SET UP CA
 1304  005072  005010                            CLR     (R0)            ;MAKE WORKING ADRS ZERO
 1305  005074  013737  001124  013226            MOV     $GDDAT,XYHOLD   ;DATA WRITTEN IN LIST MD TO XYHOLD REG
 1306  005102  013777  001124  174442            MOV     $GDDAT,@NCXYH   ;CAUSE A LIST TO MEMORY
 1307  005110  012737  005044  001110            MOV     #1$,$LPERR      ;SET UP SCOPE LOOP ADRS
 1308  005116  011037  001126                    MOV     (R0),$BDDAT     ;READ DATA WRITTEN TO CORE
 1309  005122  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
 1310  005130  001401                            BEQ     2$              ;BR IF SO
 1311  005132  104012                            ERROR   12              ;LIST MODE NPR FAILURE, CHECK DATA
 1312                                                                    ;AND CA PATHS - IF ADRS BITS BAD, A
 1313                                                                    ;PROGRAM LOC COULD BE DESTROYED.
 1314  005134  005337  001124          2$:       DEC     $GDDAT          ;DECREASE EXPECTED NPR DATA PATTERN
 1315  005140  102341                            BVC     1$              ;BR IF NOT ALL COUNTS DONE
 1316
 1317                                    ;;*******************************************************************
 1318                                    ;*TEST 26       TEST LIST MODE FOR PROPER CA & WC INCREMENTING WITH DATA & WCO CHECKS
 1319                                    ;;*******************************************************************
 1320  005142  000004                    TST26:  SCOPE
 1321  005144  012737  000010  001160            MOV     #10,$TIMES      ;;DO 10 ITERATIONS
 1322  005152  012737  000340  177776            MOV     #PR7,PSW        ;DON'T WANT ANY INTR'S
 1323  005160  012700  020000                    MOV     #20000,R0       ;START AT LOC 20000
 1324  005164  012702  125252                    MOV     #125252,R2      ;INITIALIZE DATA PATTERN
 1325  005170  012701  160000                    MOV     #-8192.,R1      ;DO 40000 XFERS (8)
 1326  005174  012777  000100  174360  1$:       MOV     #100,@NCSFR     ;CLR ALL
 1327  005202  010077  174350                    MOV     R0,@NCZLCA      ;SET UP CA AT 20000
 1328  005206  010177  174342                    MOV     R1,@NCZHWC      ;SET UP WC
 1329  005212  012777  000405  174326            MOV     #405,@NCCSR     ;SELECT LIST MD + NPR ENABLE
 1330  005220  012777  000001  174334  2$:       MOV     #1,@NCSFR       ;CLR HOLD REG
 1331  005226  005010                            CLR     (R0)            ;ZERO CA LOC
 1332  005230  010277  174316                    MOV     R2,@NCXYH       ;CAUSE A LIST NPR TO MEM
 1333  005234  010237  001124                    MOV     R2,$GDDAT       ;LD EXPECTED DATA
 1334  005240  010237  013226                    MOV     R2,XYHOLD       ;DATA WRITTEN IN LIST MD TO XYHOLD REG
 1335  005244  010037  001122                    MOV     R0,$BDADR       ;SET UP CURRENT LIST ADRS
 1336  005250  011037  001126                    MOV     (R0),$BDDAT     ;GET DATA WRITTEN TO CORE
 1337  005254  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
 1338  005262  001401                            BEQ     3$              ;BR IF SO
 1339  005264  104012                            ERROR   12              ;LIST MODE NPR DATA FAILURE, CHECK DATA
 1340                                                                    ;AND CA PATHS
 1341  005266  010037  001124          3$:       MOV     R0,$GDDAT       ;LD EXPECTED CA
 1342  005272  062737  000002  001124            ADD     #2,$GDDAT       ;SHOULD BE TWO GREATER THAN BEFORE LAST NPR LIST
 1343  005300  017737  174252  001126            MOV     @NCZLCA,$BDDAT  ;GET CA
 1344  005306  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
 1345  005314  001404                            BEQ     4$              ;BR IF SO
 1346  005316  013737  001556  001122            MOV     NCZLCA,$BDADR   ;SET UP CA REG ADRS
 1347  005324  104013                            ERROR   13              ;LIST MODE CA INCREMENT FAILURE
 1348  005326  010137  001124          4$:       MOV     R1,$GDDAT       ;LD EXPECTED WC
 1349  005332  005237  001124                    INC     $GDDAT          ;SHOULD BE ONE GREATER THAN BEFORE LAST NPR LIST
 1350  005336  017737  174212  001126            MOV     @NCZHWC,$BDDAT  ;GET WC
 1351  005344  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
 1352  005352  001404                            BEQ     5$              ;BR IF SO
 1353  005354  013737  001554  001122            MOV     NCZHWC,$BDADR   ;SET UP WC REG ADRS
```

```
1354  005362  104014                          ERROR   14              ;LIST MODE WC INCREMENT FAILURE
1355  005364  013737  001546  001122  5$:     MOV     NCCSR,$BDADR    ;SET UP CSR REG ADRS
1356  005372  005737  001124                  TST     $GDDAT          ;WCO EXPECTED?
1357  005376  001413                          BEQ     6$              ;BR IF SO
1358  005400  032777  010000  174140          BIT     #BIT12,@NCCSR   ;WCO FLAG SET WHEN NOT EXPECTED?
1359  005406  001422                          BEQ     7$              ;BR IF NOT
1360  005410  005037  001124                  CLR     $GDDAT          ;LD EXPECTED
1361  005414  012737  010000  001126          MOV     #BIT12,$BDDAT   ;SHOW IT WAS SET
1362  005422  104015                          ERROR   15              ;WC OVERFLOW FLAG SET PREMATURELY
1363  005424  000413                          BR      7$              ;CONTINUE NEXT TO NEXT LIST NPR
1364  005426  032777  010000  174112  6$:     BIT     #BIT12,@NCCSR   ;WCO BIT SET WHEN EXPECTED?
1365  005434  001014                          BNE     8$              ;BR IF SO
1366  005436  012737  010000  001124          MOV     #BIT12,$GDDAT   ;LD EXPECTED WCO FLAG
1367  005444  005037  001126                  CLR     $BDDAT          ;SHOW IT WAS NOT SET
1368  005450  104015                          ERROR   15              ;WC OVERFLOW FLAG FAILED TO SET ON WCO
1369  005452  000420                          BR      TST27           ;;NEXT TEST IF NO LOOP ON ER
1370  005454  005102                  7$:     COM     R2              ;COMPLEMENT NPR DATA
1371  005456  005201                          INC     R1              ;ADVANCE WC
1372  005460  062700  000002                  ADD     #2,R0           ;BUMP CA
1373  005464  000655                          BR      2$              ;DO LIST TO NEXT ADRS
1374  005466  000005                  8$:     RESET                   ;CLR WCO
1375  005470  032777  010000  174050          BIT     #BIT12,@NCCSR   ;DID RESET CLR WCO FLAG?
1376  005476  001406                          BEQ     TST27           ;;NEXT TEST IF CLRED
1377  005500  005037  001124                  CLR     $GDDAT          ;LD EXPECTED
1378  005504  012737  010000  001126          MOV     #BIT12,$BDDAT   ;INDICATE IT WAS SET
1379  005512  104015                          ERROR   15              ;WCO FLAG FAILED TO CLR WITH RESET
1380
1381                                  ;;**************************************************************
1382                                  ;*TEST 27        TEST SPECIAL FUNCTION CLEAR WCO FLAG
1383                                  ;;**************************************************************
1384  005514  000004                  TST27:  SCOPE
1385  005516  012737  000340  177776          MOV     #PR7,PSW        ;DON'T WANT WCO INTR
1386  005524  012777  000100  174030          MOV     #100,@NCSFR     ;CLR ALL
1387  005532  012777  020000  174016          MOV     #20000,@NCZLCA  ;SET UP CA AT 20000
1388  005540  012777  177777  174006          MOV     #-1,@NCZHWC     ;SET UP WC FOR ONE XFER
1389  005546  012777  000405  173772          MOV     #405,@NCCSR     ;SELECT LIST MD + NPR ENABLE
1390  005554  005077  173772                  CLR     @NCXYH          ;CAUSE A LIST TO MEM
1391  005560  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET UP CSR ADRS
1392  005566  012777  000020  173766          MOV     #20,@NCSFR      ;USE SPECIAL FUNCTION CLR WCO FLAG
1393  005574  032777  010000  173760          BIT     #BIT12,@NCSFR   ;DID IT CLR?
1394  005602  001401                          BEQ     TST30           ;;NEXT TEST IF SO
1395  005604  104015                          ERROR   15              ;WCO FLAG FAILED TO CLR WITH SPECIAL FUNCTION CLR.
1396
1397                                  ;;**************************************************************
1398                                  ;*TEST 30        TEST THAT WCO WILL CAUSE AN INTERRUPT
1399                                  ;;**************************************************************
1400  005606  000004                  TST30:  SCOPE
1401  005610  012737  000340  177776          MOV     #PR7,PSW        ;SET PRIORITY TO HIGHER
1402  005616  012777  000100  173736          MOV     #100,@NCSFR     ;CLR ALL
1403  005624  012777  005752  173736          MOV     #2$,@NCVCT4     ;SET UP INTR SERVICE ADRS
1404  005632  012777  000340  173732          MOV     #PR7,@NCVCT6    ;SET UP INTR PRIORITY
1405  005640  012777  004405  173700          MOV     #4405,@NCCSR    ;ENABLE WCO, LIST MD + NPR ENABLE
1406  005646  012777  177776  173700          MOV     #-2,@NCZHWC     ;SET UP WC FOR 2 XFERS
1407  005654  012777  020000  173674          MOV     #20000,@NCZLCA  ;SET UP CA AT ADRS 20000
1408  005662  012700  000002                  MOV     #2,R0           ;R0 WILL COUNT NPR'S
1409  005666  005077  173660          1$:     CLR     @NCXYH          ;CAUSE A LIST TO MEM
```

```
1410  005672  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET UP CSR ADRS
1411  005700  013737  001576  177776          MOV     NCBRL1,PSW      ;ALLOW INTR ANYTIME
1412  005706  021616                          CMP     (SP),(SP)       ;STALL
1413  005710  021616                          CMP     (SP),(SP)       ;STALL SOMEMORE
1414  005712  012777  000001  173642          MOV     #1,@NCSFR       ;SPECIAL FUNCTION CLR X Y HOLD REG
1415  005720  005300                          DEC     R0              ;DID WE EXPECT AN INTR?
1416  005722  001361                          BNE     1$              ;BR IF NOT
1417  005724  012737  010000  001124          MOV     #BIT12,$GDDAT   ;LD EXPECTED
1419  005732  017737  173610  001126          MOV     @NCCSR,$BDDAT   ;READ CSR
1419  005740  042737  167777  001126          BIC     #167777,$BDDAT  ;SAVE ONLY WCO FLAG
1420  005746  104015                          ERROR   15              ;WCO FAILED TO INTERRUPT
1421  005750  000433                          BR      4$              ;SET UP EXIT TEST
1422  005752  022626                  2$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
1423  005754  005300                          DEC     R0              ;CHECK THAT THE INTR WAS EXPECTED
1424  005756  001412                          BEQ     3$              ;BR IF SO
1425  005760  005037  001124                  CLR     $GDDAT          ;DID NOT WANT WCO INTR UNTIL 2ND XFER
1426  005764  017737  173556  001126          MOV     @NCCSR,$BDDAT   ;READ CSR
1427  005772  042737  167777  001126          BIC     #167777,$BDDAT  ;SAVE ONLY FLAG
1428  006000  104015                          ERROR   15              ;WCO INTERRUPTED BEFORE LAST XFER
1429  006002  000416                          BR      4$              ;WCO INTERRUPTED BEFORE LAST XFER
1430  006004  012737  010000  001124  3$:     MOV     #BIT12,$GDDAT   ;LD EXPECTED
1431  006012  017737  173530  001126          MOV     @NCCSR,$BDDAT   ;READ CSR
1432  006020  042737  167777  001126          BIC     #167777,$BDDAT  ;SAVE ONLY WCO FLAG
1433  006026  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1434  006034  001401                          BEQ     4$              ;BR IF SO
1435  006036  104015                          ERROR   15              ;WCO FLAG FAILED TO SET ON WCO ON INTR
1436  006040  013777  001572  173522  4$:     MOV     NCVCT6,@NCVCT4  ;RESTORE VECTOR PTR TO HALT
1437  006046  005077  173520                  CLR     @NCVCT6         ;LOAD HALT
1438
1439                                          ;;**************************************************************
1440                                          ;*TEST 31       TEST JOY STICK MODE FOR CORRECT ADDRESS MAKE
1441                                          ;;**************************************************************
1442  006052  000004                  TST31:  SCOPE
1443  006054  012737  006122  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
1444  006062  005037  013232                  CLR     TAM             ;SELECT NO NPR
1445  006066  005037  013212                  CLR     DUAL            ;NO B GAMMA
1446  006072  012777  000100  173462          MOV     #100,@NCSFR     ;CLR ALL
1447  006100  005037  013234                  CLR     OFFSET          ;NO OFFSET IN JOY STICK MD
1448  006104  012737  000010  013236          MOV     #10,SOFCSR      ;SELECT JOY STICK MODE
1449  006112  004737  012714                  JSR     PC,REGLD        ;GO LOAD CSR
1450  006116  004737  012746          1$:     JSR     PC,MAKE         ;GO MAKE UP ADRS
1451  006122  004737  013050          2$:     JSR     PC,MKLOOP       ;GO MAKE UP SAME ADRS (SCOPE LOOP)
1452  006126  000405                          BR      TST32           ;;NEXT TEST - ALL ADRS MADE
1453  006130  023737  001120  001122          CMP     $GDADR,$BDADR   ;ADRS CORRECT?
1454  006136  001767                          BEQ     1$              ;TRY NEXT ADRS IF SO
1455  006140  104017                          ERROR   17              ;JOY STICK ADRS MAKE ER
1456
1457                                          ;;**************************************************************
1458                                          ;*TEST 32       TEST JOYSTICK MODE FOR ABILITY TO OVERRIDE OTHER FUNCTIONS
1459                                          ;;**************************************************************
1460  006142  000004                  TST32:  SCOPE
1461  006144  012737  006204  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
1462  006152  012777  000100  173402          MOV     #100,@NCSFR     ;CLR ALL
1463  006160  012737  177777  013234          MOV     #-1,OFFSET      ;SET ALL POSSIBLE ADRS BITS
1464  006166  012737  003415  013236          MOV     #3415,SOFCSR    ;SELECT RES 7, JOYSTICK MD, NPR EN + LIST MD
1465  006174  004737  012714                  JSR     PC,REGLD        ;GO LOAD CSR + OFFSET REGS
```

```
1466  006200  004737  012746          1$:   JSR   PC,MAKE          ;GO MAKE UP ADRS
1467  006204  004737  013050          2$:   JSR   PC,MKLOOP        ;GO MAKE UP SAME ADRS (SCOPE LOOP)
1468  006210  000405                        BR    TST33            ;;NEXT TEST - ALL ADRS MADE
1469  006212  023737  001120  001122        CMP   $GDADR,$BDADR    ;ADRS CORRECT?
1470  006220  001767                        BEQ   1$               ;TRY NEXT ADRS IF SO
1471  006222  104020                        ERROR 20               ;JOYSTICK MD FAILED TO OVERRIDE LIST MD,
1472                                                               ; OFFSET REG, OR RESOLUTION LOGIC
1473
1474                                  ;;*****************************************************************
1475                                  ;*TEST 33       TEST RESOLUTION 0 TO YIELD 0
1476                                  ;;*****************************************************************
1477  006224  000004                TST33:  SCOPE
1478  006226  005037  013232                CLR   TAM              ;NO NPR ENABLE THIS TEST
1479  006232  012777  000100  173322        MOV   #100,@NCSFR      ;CLR ALL
1480  006240  012737  006266  001110        MOV   #3$,$LPERR       ;SET UP SCOPE LOOP ADRS
1481  006246  005037  013236                CLR   SOFCSR           ;SELECT NO MODE OR RESOLUTION
1482  006252  005037  013234                CLR   OFFSET           ;SELECT 0 OFFSET INITIALLY
1483  006256  004737  012714          1$:   JSR   PC,REGLD         ;GO LOAD CSR & OFFSET REGS
1484  006262  004737  012746          2$:   JSR   PC,MAKE          ;GO MAKE UP ADRS
1485  006266  004737  013050          3$:   JSR   PC,MKLOOP        ;GO MAKE UP SAME ADRS (SCOPE LOOP)
1486  006272  000405                        BR    4$               ;GO UPDATE OFFSET
1487  006274  023737  001120  001122        CMP   $GDADR,$BDADR    ;ADRS CORRECT?
1488  006302  001767                        BEQ   2$               ;TRY NEXT ADRS IF SO
1489  006304  104017                        ERROR 17               ;RESOLUTION 0 SHOULD HAVE MADE ADRS 0
1490  006306  062737  170000  013234  4$:   ADD   #170000,OFFSET   ;SET ALL OFFSET BITS SHOULD HAVE NO EFFECT
1491  006314  103360                        BCC   1$               ;NOW REPEAT WITH OFFSET BITS SET
1492
1493                                  ;;*****************************************************************
1494                                  ;*TEST 34       TEST RESOLUTION 1 - LIST MATRIX
1495                                  ;;*****************************************************************
1496  006316  000004                TST34:  SCOPE
1497  006320  005037  013232                CLR   TAM              ;NO NPR ENABLE THIS TEST
1498  006324  012777  000100  173230        MOV   #100,@NCSFR      ;CLR ALL
1499  006332  012737  006362  001110        MOV   #3$,$LPERR       ;SET UP SCOPE LOOP ADRS
1500  006340  005037  013234                CLR   OFFSET           ;SELECT 0 OFFSET INITIALLY
1501  006344  012737  000400  013236        MOV   #400,SOFCSR      ;SELECT LIST MATRIX
1502  006352  004737  012714          1$:   JSR   PC,REGLD         ;GO LOAD CSR & OFFSET
1503  006356  004737  012746          2$:   JSR   PC,MAKE          ;GO MAKE UP ADRS
1504  006362  004737  013050          3$:   JSR   PC,MKLOOP        ;GO MAKE UP SAME ADRS (SCOPE LOOP)
1505  006366  000405                        BR    4$               ;GO UPDATE OFFSET
1506  006370  023737  001120  001122        CMP   $GDADR,$BDADR    ;ADRS CORRECT?
1507  006376  001767                        BEQ   2$               ;TRY NEXT ADRS IF SO
1508  006400  104017                        ERROR 17               ;LIST MATRIX ADRS MAKE ER
1509  006402  062737  170000  013234  4$:   ADD   #170000,OFFSET   ;SET ALL OFFSET BITS - SHOULD HAVE NO EFFECT
1510  006410  103360                        BCC   1$               ;NOW REPEAT WITH OFFSET BITS SET
1511
1512                                  ;;*****************************************************************
1513                                  ;*TEST 35       TEST RESOLUTION 2 - 64X64X16
1514                                  ;;*****************************************************************
1515  006412  000004                TST35:  SCOPE
1516  006414  012737  000100  001160        MOV   #100,$TIMES      ;;DO 100 ITERATIONS
1517  006422  012737  006466  001110        MOV   #4$,$LPERR       ;SET UP SCOPE LOOP ADRS
1518  006430  005037  013232                CLR   TAM              ;SELECT NO NPR ENABLE INITIALLY
1519  006434  012777  000100  173120  1$:   MOV   #100,@NCSFR      ;CLR ALL
1520  006442  012737  020000  013234        MOV   #20000,OFFSET    ;SET UP BASE
1521  006450  012737  001000  013236        MOV   #1000,SOFCSR     ;SELECT 64X64X16
```

```
1522  006456  004737  012714          2$:     JSR     PC,REGLD        ;GO LOAD CSR + OFFSET
1523  006462  004737  012746          3$:     JSR     PC,MAKE         ;GO DO ADRS MAKE
1524  006466  004737  013050          4$:     JSR     PC,MKLOOP       ;ADRS MAKE SCOPE LOOP
1525  006472  000417                          BR      6$              ;RETURN HERE IF ALL ADRS MADE
1526  006474  005737  013232                  TST     TAM             ;WAS NPR ENABLED?
1527  006500  001006                          BNE     5$              ;BR IF SO
1528  006502  023737  001120  001122          CMP     $GDADR,$BDADR   ;ADRS CORRECT?
1529  006510  001764                          BEQ     3$              ;TRY NEXT ADRS IF SO
1530  006512  104017                          ERROR   17              ;ADRS MAKE ER 64X64X16
1531  006514  000422                          BR      TST36           ;;TO NEXT TEST
1532  006516                          5$:
1533  006516  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1534  006524  001756                          BEQ     3$              ;TRY NEXT ADRS IF SO
1535  006526  104021                          ERROR   21              ;NPR INCREMENT ER AT ADRS INDICATED
1536  006530  000414                          BR      TST36           ;;TO NEXT TEST
1537  006532  062737  020000  013234  6$:     ADD     #20000,OFFSET   ;ADVANCE BASE ADRS
1538  006540  103405                          BCS     7$              ;BR IF 15 BITS TESTED(NON-NPR)
1539  006542  100345                          BPL     2$              ;BR IF 14 BITS NOT TESTED
1540  006544  005737  013232                  TST     TAM             ;ALLOW ONLY 14 BITS IF NPR
1541  006550  001004                          BNE     TST36           ;;NEXT TEST IF NPR TESTED TO 16K
1542  006552  000741                          BR      2$              ;GO TO 15 BITS SINCE NON-NPR
1543  006554  005237  013232          7$:     INC     TAM             ;SELECT NPR
1544  006560  000725                          BR      1$              ;REPEAT TEST(NPR ENABLED)
1545
1546                                  ;;**************************************************************
1547                                  ;*TEST 36        TEST RESOLUTION 3 - 32X32X16
1548                                  ;;**************************************************************
1549  006562  000004                  TST36:  SCOPE
1550  006564  012737  000020  001160          MOV     #20,$TIMES      ;;DO 20 ITERATIONS
1551  006572  012737  006636  001110          MOV     #4$,$LPERR      ;SET UP SCOPE LOOP ADRS
1552  006600  005037  013232                  CLR     TAM             ;SELECT NO NPR ENABLE INITIALLY
1553  006604  012777  000100  172750  1$:     MOV     #100,@NCSFR     ;CLR ALL
1554  006612  012737  020000  013234          MOV     #20000,OFFSET   ;SET UP BASE
1555  006620  012737  001400  013236          MOV     #1400,$OFCSR    ;SELECT 32X32X16
1556  006626  004737  012714          2$:     JSR     PC,REGLD        ;GO LOAD CSR + OFFSET
1557  006632  004737  012746          3$:     JSR     PC,MAKE         ;GO DO ADRS MAKE
1558  006636  004737  013050          4$:     JSR     PC,MKLOOP       ;ADRS MAKE SCOPE LOOP
1559  006642  000417                          BR      6$              ;RETURN HERE IF ALL ADRS MADE
1560  006644  005737  013232                  TST     TAM             ;WAS NPR ENABLED?
1561  006650  001006                          BNE     5$              ;BR IF SO
1562  006652  023737  001120  001122          CMP     $GDADR,$BDADR   ;ADRS CORRECT?
1563  006660  001764                          BEQ     3$              ;TRY NEXT ADRS IF SO
1564  006662  104017                          ERROR   17              ;ADRS MAKE ER 32X32X16
1565  006664  000422                          BR      TST37           ;;TO NEXT TEST
1566  006666                          5$:
1567  006666  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1568  006674  001756                          BEQ     3$              ;TRY NEXT ADRS IF SO
1569  006676  104021                          ERROR   21              ;NPR INCREMENT ER AT ADRS INDICATED
1570  006700  000414                          BR      TST37           ;;TO NEXT TEST
1571  006702  062737  004000  013234  6$:     ADD     #4000,OFFSET    ;ADVANCE BASE ADRS
1572  006710  103405                          BCS     7$              ;BR IF 15 BITS TESTED(NON-NPR)
1573  006712  100345                          BPL     2$              ;BR IF 14 BITS NOT TESTED
1574  006714  005737  013232                  TST     TAM             ;ALLOW ONLY 14 BITS IF NPR
1575  006720  001004                          BNE     TST37           ;;NEXT TEST IF NPR TESTED TO 16K
1576  006722  000741                          BR      2$              ;GO TO 15 BITS SINCE NON-NPR
1577  006724  005237  013232          7$:     INC     TAM             ;SELECT NPR
```

```
1578  006730  000725                          BR      1$              ;REPEAT TEST(NPR ENABLED)
1579
1580                                  ;;*****************************************************************
1581                                  ;*TEST 37       TEST RESOLUTION 4 - 128X128X8
1582                                  ;;*****************************************************************
1583  006732  000004                 TST37:  SCOPE
1584  006734  012737  000200  001160         MOV     #200,$TIMES     ;;DO 200 ITERATIONS
1585  006742  012737  007006  001110         MOV     #4$,$LPERR      ;SET UP SCOPE LOOP ADRS
1586  006750  005037  013232                 CLR     TAM             ;NO NPR ENABLE THIS TEST
1587  006754  012777  000100  172600  1$:    MOV     #100,@NCSFR     ;CLR ALL
1588  006762  012737  040000  013234         MOV     #40000,OFFSET   ;SET UP BASE ADRS
1589  006770  012737  002000  013236         MOV     #2000,$OFCSR    ;SELECT 128X128X8
1590  006776  004737  012714         2$:    JSR     PC,REGLD        ;GO LOAD CSR + OFFSET
1591  007002  004737  012746         3$:    JSR     PC,MAKE         ;GO DO ADRS MAKE
1592  007006  004737  013050         4$:    JSR     PC,MKLOOP       ;ADRS MAKE SCOPE LOOP
1593  007012  000417                         BR      6$              ;RETURN HERE IF ALL ADRS MADE
1594  007014  005737  013232                 TST     TAM             ;WAS NPR ENABLED?
1595  007020  001006                         BNE     5$              ;BR IF SO
1596  007022  023737  001120  001122         CMP     $GDADR,$BDADR   ;ADRS CORRECT?
1597  007030  001764                         BEQ     3$              ;TRY NEXT ADRS IF SO
1598  007032  104017                         ERROR   17              ;ADRS MAKE ER 128X128X8
1599  007034  000422                         BR      TST40           ;;TO NEXT TEST
1600  007036                         5$:
1601  007036  023737  001124  001126         CMP     $GDDAT,$BDDAT   ;CORRECT?
1602  007044  001756                         BEQ     3$              ;TRY NEXT ADRS IF SO
1603  007046  104021                         ERROR   21              ;NPR INCREMENT ER AT ADRS INDICATED
1604  007050  000414                         BR      TST40           ;;TO NEXT TEST
1605  007052  062737  040000  013234  6$:    ADD     #40000,OFFSET   ;ADVANCE BASE ADRS
1606  007060  103405                         BCS     7$              ;BR IF 15 BITS TESTED(NON-NPR)
1607  007062  100345                         BPL     2$              ;BR IF 14 BITS NOT TESTED
1608  007064  005737  013232                 TST     TAM             ;ALLOW ONLY 14 BITS IF NPR
1609  007070  001004                         BNE     TST40           ;;NEXT TEST IF NPR TESTED TO 16K
1610  007072  000741                         BR      2$              ;GO TO 15 BITS SINCE NON-NPR
1611  007074  005237  013232         7$:    INC     TAM             ;SELECT NPR
1612  007100  000725                         BR      1$              ;REPEAT TEST(NPR ENABLED)
1613
1614                                  ;;*****************************************************************
1615                                  ;*TEST 40       TEST RESOLUTION 5 - 128X128X8 (4K BASE)
1616                                  ;;*****************************************************************
1617  007102  000004                 TST40:  SCOPE
1618  007104  012737  000200  001160         MOV     #200,$TIMES     ;;DO 200 ITERATIONS
1619  007112  012737  007156  001110         MOV     #4$,$LPERR      ;SET UP SCOPE LOOP ADRS
1620  007120  005037  013232                 CLR     TAM             ;SELECT NO NPR ENABLE INITIALLY
1621  007124  012777  000100  172430  1$:    MOV     #100,@NCSFR     ;CLR ALL
1622  007132  012737  000000  013234         MOV     #0,OFFSET       ;SET UP BASE
1623  007140  012737  002400  013236         MOV     #2400,$OFCSR    ;SELECT 128X128X8 (4K BASE)
1624  007146  004737  012714         2$:    JSR     PC,REGLD        ;GO LOAD CSR + OFFSET
1625  007152  004737  012746         3$:    JSR     PC,MAKE         ;GO DO ADRS MAKE
1626  007156  004737  013050         4$:    JSR     PC,MKLOOP       ;ADRS MAKE SCOPE LOOP
1627  007162  000417                         BR      6$              ;RETURN HERE IF ALL ADRS MADE
1628  007164  005737  013232                 TST     TAM             ;WAS NPR ENABLED?
1629  007170  001006                         BNE     5$              ;BR IF SO
1630  007172  023737  001120  001122         CMP     $GDADR,$BDADR   ;ADRS CORRECT?
1631  007200  001764                         BEQ     3$              ;TRY NEXT ADRS IF SO
1632  007202  104017                         ERROR   17              ;ADRS MAKE ER 128X128X8 (4K BASE)
1633  007204  000422                         BR      TST41           ;;TO NEXT TEST
```

```
1634  007206                                  5$:
1635  007206  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
1636  007214  001756                            BEQ     3$              ;TRY NEXT ADRS IF SO
1637  007216  104021                            ERROR   21              ;NPR INCREMENT ER AT ADRS INDICATED
1638  007220  000414                            BR      TST41           ;;TO NEXT TEST
1639  007222  062737  100000  013234  6$:       ADD     #100000,OFFSET  ;ADVANCE BASE ADRS
1640  007230  103405                            BCS     7$              ;BR IF 15 BITS TESTED(NON-NPR)
1641  007232  100345                            BPL     2$              ;BR IF 14 BITS NOT TESTED
1642  007234  005737  013232                    TST     TAM             ;ALLOW ONLY 14 BITS IF NPR
1643  007240  001004                            BNE     TST41           ;;NEXT TEST IF NPR TESTED TO 16K
1644  007242  000741                            BR      2$              ;GO TO 15 BITS SINCE NON-NPR
1645  007244  005237  013232          7$:       INC     TAM             ;SELECT NPR
1646  007250  000725                            BR      1$              ;REPEAT TEST(NPR ENABLED)
1647
1648                                  ;;*****************************************************************
1649                                  ;*TEST 41        TEST RESOLUTION 6 - 64X64X8
1650                                  ;;*****************************************************************
1651  007252  000004                  TST41:  SCOPE
1652  007254  012737  000040  001160            MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1653  007262  012737  007326  001110            MOV     #4$,$LPERR      ;SET UP SCOPE LOOP ADRS
1654  007270  005037  013232                    CLR     TAM             ;SELECT NO NPR ENABLE INITIALLY
1655  007274  012777  000100  172260  1$:       MOV     #100,@NCSFR     ;CLR ALL
1656  007302  012737  020000  013234            MOV     #20000,OFFSET   ;SET UP BASE
1657  007310  012737  003000  013236            MOV     #3000,$OFCSR    ;SELECT 64X64X8
1658  007316  004737  012714          2$:       JSR     PC,REGLD        ;GO LOAD CSR + OFFSET
1659  007322  004737  012746          3$:       JSR     PC,MAKE         ;GO DO ADRS MAKE
1660  007326  004737  013050          4$:       JSR     PC,MKLOOP       ;ADRS MAKE SCOPE LOOP
1661  007332  000417                            BR      6$              ;RETURN HERE IF ALL ADRS MADE
1662  007334  005737  013232                    TST     TAM             ;WAS NPR ENABLED?
1663  007340  001006                            BNE     5$              ;BR IF SO
1664  007342  023737  001120  001122            CMP     $GDADR,$BDADR   ;ADRS CORRECT?
1665  007350  001764                            BEQ     3$              ;TRY NEXT ADRS IF SO
1666  007352  104017                            ERROR   17              ;ADRS MAKE ER 64X64X8
1667  007354  000422                            BR      TST42           ;;TO NEXT TEST
1668  007356                                  5$:
1669  007356  023737  001124  001126            CMP     $GDDAT,$BDDAT   ;CORRECT?
1670  007364  001756                            BEQ     3$              ;TRY NEXT ADRS IF SO
1671  007366  104021                            ERROR   21              ;NPR INCREMENT ER AT ADRS INDICATED
1672  007370  000414                            BR      TST42           ;;TO NEXT TEST
1673  007372  062737  010000  013234  6$:       ADD     #10000,OFFSET   ;ADVANCE BASE ADRS
1674  007400  103405                            BCS     7$              ;BR IF 15 BITS TESTED(NON-NPR)
1675  007402  100345                            BPL     2$              ;BR IF 14 BITS NOT TESTED
1676  007404  005737  013232                    TST     TAM             ;ALLOW ONLY 14 BITS IF NPR
1677  007410  001004                            BNE     TST42           ;;NEXT TEST IF NPR TESTED TO 16K
1678  007412  000741                            BR      2$              ;GO TO 15 BITS SINCE NON-NPR
1679  007414  005237  013232          7$:       INC     TAM             ;SELECT NPR
1680  007420  000725                            BR      1$              ;REPEAT TEST(NPR ENABLED)
1681
1682                                  ;;*****************************************************************
1683                                  ;*TEST 42        TEST RESOLUTION 7 - 32X32X8
1684                                  ;;*****************************************************************
1685  007422  000004                  TST42:  SCOPE
1686  007424  012737  000020  001160            MOV     #20,$TIMES      ;;DO 20 ITERATIONS
1687  007432  012737  007476  001110            MOV     #4$,$LPERR      ;SET UP SCOPE LOOP ADRS
1688  007440  005037  013232                    CLR     TAM             ;SELECT NO NPR ENABLE INITIALLY
1689  007444  012777  000100  172110  1$:       MOV     #100,@NCSFR     ;CLR ALL
```

```
1690  007452  012737  020000  013234          MOV     #20000,OFFSET   ;SET UP BASE
1691  007460  012737  003400  013236          MOV     #3400,$OFCSR    ;SELECT 32X32X8
1692  007466  004737  012714          2$:     JSR     PC,REGLD        ;GO LOAD CSR + OFFSET
1693  007472  004737  012746          3$:     JSR     PC,MAKE         ;GO DO ADRS MAKE
1694  007476  004737  013050          4$:     JSR     PC,MKLOOP       ;ADRS MAKE SCOPE LOOP
1695  007502  000417                          BR      6$              ;RETURN HERE IF ALL ADRS MADE
1696  007504  005737  013232                  TST     TAM             ;WAS NPR ENABLED?
1697  007510  001006                          BNE     5$              ;BR IF SO
1698  007512  023737  001120  001122          CMP     $GDADR,$3DADR   ;ADRS CORRECT?
1699  007520  001764                          BEQ     3$              ;TRY NEXT ADRS IF SO
1700  007522  104017                          ERROR   17              ;ADRS MAKE ER 32X32X8
1701  007524  000422                          BR      TST43           ;;TO NEXT TEST
1702  007526                          5$:
1703  007526  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1704  007534  001756                          BEQ     3$              ;TRY NEXT ADRS IF SO
1705  007536  104021                          ERROR   21              ;NPR INCREMENT ER AT ADRS INDICATED
1706  007540  000414                          BR      TST43           ;;TO NEXT TEST
1707  007542  062737  002000  013234  6$:     ADD     #2000,OFFSET    ;ADVANCE BASE ADRS
1708  007550  103405                          BCS     7$              ;BR IF 15 BITS TESTED(NON-NPR)
1709  007552  100345                          BPL     2$              ;BR IF 14 BITS NOT TESTED
1710  007554  005737  013232                  TST     TAM             ;ALLOW ONLY 14 BITS IF NPR
1711  007560  001004                          BNE     TST43           ;;NEXT TEST IF NPR TESTED TO 16K
1712  007562  000741                          BR      2$              ;GO TO 15 BITS SINCE NON-NPR
1713  007564  005237  013232          7$:     INC     TAM             ;SELECT NPR
1714  007570  000725                          BR      1$              ;REPEAT TEST(NPR ENABLED)
1715
1716                                  ;;*****************************************************************
1717                                  ;*TEST 43        TEST RESOLUTION 2 WITH B-GAMMA SET
1718                                  ;;*****************************************************************
1719  007572  000004                  TST43:  SCOPE
1720  007574  012737  000200  001160          MOV     #200,$TIMES     ;;DO 200 ITERATIONS
1721  007602  012737  007654  001110          MOV     #3$,$LPERR      ;SET UP SCOPE LOOP ADRS
1722  007610  005037  013232                  CLR     TAM             ;NO NPR ENABLE THIS TEST
1723  007614  012737  000001  013212          MOV     #1,DUAL         ;DUAL SAYS B-GAMA ADRS MAKE
1724  007622  012777  000100  171732          MOV     #100,@NCSFR     ;CLR ALL
1725  007630  012737  020000  013234          MOV     #20000,OFFSET   ;SET UP BASE
1726  007636  012737  001000  013236          MOV     #1000,$OFCSR    ;SELECT 64X64X16
1727  007644  004737  012714          1$:     JSR     PC,REGLD        ;GO LOAD CSR + OFFSET
1728  007650  004737  012746          2$:     JSR     PC,MAKE         ;GO DO ADRS MAKE
1729  007654  004737  013050          3$:     JSR     PC,MKLOOP       ;ADRS MAKE SCOPE LOOP
1730  007660  000406                          BR      4$              ;RETURN HERE IF ALL ADRS MADE
1731  007662  023737  001120  001122          CMP     $GDADR,$BDADR   ;ADRS CORRECT?
1732  007670  001767                          BEQ     2$              ;TRY NEXT ADRS IF SO
1733  007672  104017                          ERROR   17              ;ADRS MAKE ER 64X64X16 WITH B-GAMMA
1734  007674  000404                          BR      TST44           ;;TO NEXT TEST
1735  007676  062737  020000  013234  4$:     ADD     #20000,OFFSET   ;ADVANCE BASE ADRS
1736  007704  103357                          BCC     1$              ;BR IF NOT THRU 15 BITS
1737
1738                                  ;;*****************************************************************
1739                                  ;*TEST 44        TEST RESOLUTION 3 WITH B-GAMMA SET
1740                                  ;;*****************************************************************
1741  007706  000004                  TST44:  SCOPE
1742  007710  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1743  007716  012737  007770  001110          MOV     #3$,$LPERR      ;SET UP SCOPE LOOP ADRS
1744  007724  005037  013232                  CLR     TAM             ;NO NPR ENABLE THIS TEST
1745  007730  012737  000001  013212          MOV     #1,DUAL         ;DUAL SAYS B-GAMMA ADRS MAKE
```

```
1746  007736  012777  000100  171616          MOV     #100,@NCSFR     ;CLR ALL
1747  007744  012737  020000  013234          MOV     #20000,OFFSET   ;SET UP BASE
1748  007752  012737  001400  013236          MOV     #1400,$OFCSR    ;SELECT 32X32X16
1749  007760  004737  012714          1$:     JSR     PC,REGLD        ;GO LOAD CSR + OFFSET
1750  007764  004737  012746          2$:     JSR     PC,MAKE         ;GO DO ADRS MAKE
1751  007770  004737  013050          3$:     JSR     PC,MKLOOP       ;ADRS MAKE SCOPE LOOP
1752  007774  000406                          BR      4$              ;RETURN HERE IF ALL ADRS MADE
1753  007776  023737  001120  001122          CMP     $GDADR,$BDADR   ;ADRS CORRECT?
1754  010004  001767                          BEQ     2$              ;TRY NEXT ADRS IF SO
1755  010006  104017                          ERROR   17              ;ADRS MAKE ER 32X32X16 WITH B-GAMMA
1756  010010  000404                          BR      TST45           ;;TO NEXT TEST
1757  010012  062737  010000  013234  4$:     ADD     #10000,OFFSET   ;ADVANCE BASE ADRS
1758  010020  103357                          BCC     1$              ;BR IF NOT THRU 15 BITS
1759
1760                                  ;;**************************************************************
1761                                  ;*TEST 45        TEST RESOLUTION 6 WITH B-GAMMA SET
1762                                  ;;**************************************************************
1763  010022  000004                  TST45:  SCOPE
1764  010024  012737  000200  001160          MOV     #200,$TIMES     ;;DO 200 ITERATIONS
1765  010032  012737  010104  001110          MOV     #3$,$LPERR      ;SET UP SCOPE LOOP ADRS
1766  010040  005037  013232                  CLR     TAM             ;NO NPR ENABLE THIS TEST
1767  010044  012737  000001  013212          MOV     #1,DUAL         ;'DUAL' SAYS B-GAMMA ADRS MAKE
1768  010052  012777  000100  171502          MOV     #100,@NCSFR     ;CLR ALL
1769  010060  012737  020000  013234          MOV     #20000,OFFSET   ;SET UP BASE
1770  010066  012737  003000  013236          MOV     #3000,$OFCSR    ;SELECT 64X64X8
1771  010074  004737  012714          1$:     JSR     PC,REGLD        ;GO LOAD CSR + OFFSET
1772  010100  004737  012746          2$:     JSR     PC,MAKE         ;GO DO ADRS MAKE
1773  010104  004737  013050          3$:     JSR     PC,MKLOOP       ;ADRS MAKE SCOPE LOOP
1774  010110  000406                          BR      4$              ;RETURN HERE IF ALL ADRS MADE
1775  010112  023737  001120  001122          CMP     $GDADR,$BDADR   ;ADRS CORRECT?
1776  010120  001767                          BEQ     2$              ;TRY NEXT ADRS IF SO
1777  010122  104017                          ERROR   17              ;ADRS MAKE ER 64X64X8 WITH B-GAMMA
1778  010124  000404                          BR      TST46           ;;TO NEXT TEST
1779  010126  062737  020000  013234  4$:     ADD     #20000,OFFSET   ;ADVANCE BASE ADRS
1780  010134  103357                          BCC     1$              ;BR IF NOT THRU 15 BITS
1781
1782                                  ;;**************************************************************
1783                                  ;*TEST 46        TEST RESOLUTION 7 WITH B-GAMMA SET
1784                                  ;;**************************************************************
1785  010136  000004                  TST46:  SCOPE
1786  010140  012737  000040  001160          MOV     #40,$TIMES      ;;DO 40 ITERATIONS
1787  010146  012737  010220  001110          MOV     #3$,$LPERR      ;SET UP SCOPE LOOP ADRS
1788  010154  005037  013232                  CLR     TAM             ;NO NPR ENABLE THIS TEST
1789  010160  012737  000001  013212          MOV     #1,DUAL         ;'DUAL' SAYS B-GAMMA ADRS MAKE
1790  010166  012777  000100  171366          MOV     #100,@NCSFR     ;CLR ALL
1791  010174  012737  020000  013234          MOV     #20000,OFFSET   ;SET UP BASE
1792  010202  012737  003400  013236          MOV     #3400,$OFCSR    ;SELECT 32X32X8
1793  010210  004737  012714          1$:     JSR     PC,REGLD        ;GO LOAD CSR + OFFSET
1794  010214  004737  012746          2$:     JSR     PC,MAKE         ;GO DO ADRS MAKE
1795  010220  004737  013050          3$:     JSR     PC,MKLOOP       ;ADRS MAKE SCOPE LOOP
1796  010224  000406                          BR      4$              ;RETURN HERE IF ALL ADRS MADE
1797  010226  023737  001120  001122          CMP     $GDADR,$BDADR   ;ADRS CORRECT?
1798  010234  001767                          BEQ     2$              ;TRY NEXT ADRS IF SO
1799  010236  104017                          ERROR   17              ;ADRS MAKE ER 32X32X8 WITH B-GAMMA
1800  010240  000404                          BR      TST47           ;;TO NEXT TEST
1801  010242  062737  004000  013234  4$:     ADD     #4000,OFFSET    ;ADVANCE BASE ADRS
```

```
1802  010250  103357                        BCC     1$              ;BR IF NOT THRU 15 BITS
1803
1804                                        ;;*****************************************************************
1805                                        ;*TEST 47        TEST WORD INCREMENT AND INCREMENT OVERFLOW FLAG
1806                                        ;;*****************************************************************
1807  010252  000004                TST47:  SCOPE
1808  010254  012737  000004  001160        MOV     #4,$TIMES       ;;DO 4 ITERATIONS
1809  010262  012737  010344  001110        MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
1810  010270  012700  000100                MOV     #100,R0         ;SET UP OVERFLOW FLAG CLR COUNTER
1811  010274  012777  000100  171260        MOV     #100,@NCSFR     ;CLR ALL
1812  010302  012777  020000  171240        MOV     #20000,@NCOFF   ;SET UP OFFSET REG TO 20000
1813  010310  012777  001001  171230        MOV     #1001,@NCCSR    ;SELECT RESOLUTION 2 WITH NPR ENABLE
1814  010316  005037  020000                CLR     @#20000         ;START LOC 20000 AT ZERO
1815  010322  005001                        CLR     R1              ;R1 CONTAINS EXPECTED NPR DATA
1816  010324  005002                        CLR     R2              ;R2 WHEN NON-ZERO SAYS INCREMENT OVFLO FLG EXPECTED
1817
1818  010326  005201                1$:     INC     R1              ;INCREMENT EXPECTED
1819  010330  001005                        BNE     2$              ;BR IF NO OVERFLOW
1820  010332  005101                        COM     R1              ;CAN'T EXCEED 177777
1821  010334  012702  000001                MOV     #1,R2           ;OVERFLOW FLAG SHOULD SET WHEN CELL 177777
1822  010340  005300                        DEC     R0              ;COUNT CLR FLAG ATTEMPTS, 32 SOFT, 32 HARD
```

```
1823  010342  001521                          BEQ     TST50           ;;NEXT TEST IF ALL CLR FLAG ATTEMPTS COMPLETED
1824  010344  012777  000001  171210  2$:     MOV     #1,@NCSFR       ;CLR XY HOLD REG
1825  010352  012777  000000  171172          MOV     #0,@NCXYH       ;DO AN WORD INC NPR TO 20000
1826  010360  000240                          NOP                     ;STALL
1827  010362  005702                          TST     R2              ;EXPECT OVERFLOW?
1828  010364  001452                          BEQ     5$              ;BR IF NOT - BUT OK IT DID NOT SET
1829  010366  032777  020000  171152          BIT     #20000,@NCCSR   ;LOOK FOR OVERFLOW FLAG
1830  010374  001012                          BNE     3$              ;BR IS SET
1831  010376  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET UP CSR REG ADRS
1832  010404  012737  070000  001126          MOV     #70000,$BDDAT   ;EXPECTED INCREMENT OVERFLOW REG
1833  010412  005037  001126                  CLR     $BDDAT          ;SHOW IT WAS NOT SET
1834  010416  104022                          ERROR   22              ;INCREMENT OVERFLOW FLAG FAILED TO SET
1835  010420  000472                          BR      TST50           ;;NEXT TEST ON ER
1836  010422  020427  000040          3$:     CMP     R4,#40          ;CLR INCR OVFLO BY SOFTWARE?
1837  010426  100026                          BPL     4$              ;BR IF SO
1838  010430  005037  020002                  CLR     @#20002         ;INSURE LOC 20002=0
1839  010434  012777  000001  171120          MOV     #1,@NCSFR       ;CLR XY HOLD REG
1840  010442  012777  000002  171102          MOV     #2,@NCXYH       ;DO A NPR TO LOC 20002 TO CLR OVFLO FLAG
1841  010450  032777  020000  171070          BIT     #20000,@NCCSR   ;DID FLAG GET CLRED BY HARDWARE?
1842  010456  001433                          BEQ     6$              ;BR IF SO
1843  010460  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET UP NC CSR REG ADRS
1844  010466  005037  001124                  CLR     $GDDAT          ;EXPECTED NO OVERFLOW FLAG
1845  010472  012737  020000  001126          MOV     #20000,$BDDAT   ;SHOW IT DID NOT CLR ON ADRS MAKE
1846  010500  104022                          ERROR   22              ;INCREMENT OVERFLOW FLAG FAILED TO CLR ON
1847                                                                  ;HARDWARE MAKE ADRS
1848  010502  000441                          BR      TST50           ;;NEXT TEST ON ER
1849  010504  012777  000002  171050  4$:     MOV     #2,@NCSFR       ;DO A SPECIAL FUNCTION CLR ON INCR OVFLO FLG
1850  010512  032777  020000  171026  5$:     BIT     #20000,@NCCSR   ;DID FLAG SET IN ER OR FAIL TO CLR?
1851  010520  001412                          BEQ     6$              ;BR IF OK (CLRED) - CK DATA INC
1852  010522  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET UP REG CSR ADRS
1853  010530  005037  001124                  CLR     $GDDAT          ;EXPECTED NO FLAG
1854  010534  012737  020000  001126          MOV     #20000,$BDDAT   ;SHOW IT WAS SET
1855  010542  104022                          ERROR   22              ;INCREMENT OVERFLOW FLAG SET PREMATURELY
1856                                                                  ;OR FAILED TO CLR BY SPECIAL FUNCTION CLR
1857  010544  000420                          BR      TST50           ;;NEXT TEST ON ER
1858  010546  012737  000000  013226  6$:     MOV     #0,XYHOLD       ;SET UP XY HOLD LOC FOR TYPE
1859  010554  012737  020000  001122          MOV     #20000,$BDADR   ;SET UP ADRS OF NPR
1860  010562  010137  001124                  MOV     R1,$GDDAT       ;LD EXPECTED
1861  010566  013737  020000  001126          MOV     @#20000,$BDDAT  ;GET LOCATION AFTER INCREMENT
1862  010574  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1863  010602  001651                          BEQ     1$              ;BR IF SO - DO ANOTHER ADRS MAKE
1864  010604  104023                          ERROR   23              ;WORD INCREMENT FAILURE
1865
1866                                  ;;*****************************************************************
1867                                  ;*TEST 50        TEST EVEN BYTE INCREMENT AND INCREMENT OVERFLOW FLAG
1868                                  ;;*****************************************************************
1869  010606  000004                  TST50:  SCOPE
1870  010610  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1871  010616  012737  010700  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
1872  010624  012700  000100                  MOV     #100,R0         ;SET UP OVERFLOW FLAG CLR COUNTER
1873  010630  012777  000100  170724          MOV     #100,@NCSFR     ;CLR ALL
1874  010636  012777  036000  170704          MOV     #36000,@NCOFF   ;OFFSET BITS 10-13 SHOULD HAVE NO AFFECT AT RES 4
1875  010644  012777  002001  170674          MOV     #2001,@NCCSR    ;SELECT RESOLUTION 4 WITH NPR ENABLE
1876  010652  005037  020000                  CLR     @#20000         ;START LOC 20000 AT ZERO
1877  010656  005001                          CLR     R1              ;R1 CONTAINS EXPECTED NPR DATA
1878  010660  005002                          CLR     R2              ;R2 WHEN NON ZERO SAYS INCREMENT OVFL FLG EXPECTED
```

```
1879
1880  010662  105201                    1$:   INCB    R1              ;INCREMENT EXPECTED
1881  010664  001005                          BNE     2$              ;BR IF NO OVERFLOW
1882  010666  105101                          COMB    R1              ;CAN'T EXCEED 377
1883  010670  012702  000001                MOV     #1,R2           ;OVERFLOW FLAG SHULD SET WHEN CELL 377
1884  010674  005300                          DEC     R0              ;COUNT CLR FLAG ATTEMPTS, 32 SOFT, 32 HARD
1885  010676  001516                          BEQ     TST51           ;;NEXT TEST IF ALL CLR FLAG ATTEMPTS COMPLETED
1886  010700  012777  000001  170654  2$:    MOV     #1,@NCSFR       ;CLR XY HOLD REG
1887  010706  012777  040000  170636        MOV     #40000,@NCXYH   ;DO AN WORD INC NPR TO 20000
1888  010714  000240                          NOP                     ;STALL
1889  010716  005702                          TST     R2              ;EXPECT OVERFLOW?
1890  010720  001447                          BEQ     5$              ;BR IF NOT - BUT CHECK IF DID NOT SET
1891  010722  032777  020000  170616        BIT     #20000,@NCCSR   ;LOOK FOR OVERFLOW FLAG
1892  010730  001007                          BNE     3$              ;BR IF SET
1893  010732  013737  001546  001122        MOV     NCCSR,$BDADR    ;SET UP CSR REG ADRS
1894  010740  005037  001126                CLR     $BDDAT          ;SHOW IT WAS NOT SET
1895  010744  104022                          ERROR   22              ;INCREMENT OVERFLOW FLAG FAILED TO SET
1896  010746  000472                          BR      TST51           ;;NEXT TEST ON ER
1897  010750  020027  000040          3$:    CMP     R0,#40          ;CLR INCR OVFLO BY SOFTWARE?
1898  010754  100026                          BPL     4$              ;BR IF SO
1899  010756  005037  020002                CLR     @#20002         ;INSURE LOC 20002=0
1900  010762  012777  000001  170572        MOV     #1,@NCSFR       ;CLR XY HOLD REG
1901  010770  012777  040002  170554        MOV     #40002,@NCXYH   ;DO A NPR TO LOC 20002 TO CLR OVFLO FLAG
1902  010776  032777  020000  170542        BIT     #20000,@NCCSR   ;DID FLAG GET CLRED BY HARDWARE?
1903  011004  001433                          BEQ     6$              ;BR IF SO
1904  011006  013737  001546  001122        MOV     NCCSR,$BDADR    ;SET NRP NC CSR REG - ADRS
1905  011014  005037  001124                CLR     $GDDAT          ;EXPECTED NO OVERFLOW FLAG
1906  011020  012737  020000  001126        MOV     #20000,$BDDAT   ;SHOW IF DID NOT CLR ON ADRS MAKE
1907  011026  104022                          ERROR   22              ;INCREMENT OVERFLOW FLAG FAILED TO CLR ON
1908                                                                  ;HARDWARE MAKE ADRS
1909  011030  000441                          BR      TST51           ;;NEXT TEST ON ER
1910  011032  012777  000002  170522  4$:    MOV     #2,@NCSFR       ;DO A SPECIAL FUNCTION CLR ON INCR OVFLO FLAG
1911  011040  032777  020000  170500  5$:    BIT     #20000,@NCCSR   ;DID FLAG SET IN ER OR FAIL TO CLR?
1912  011046  001412                          BEQ     6$              ;BR IF OK (CLRED) - CK DATA INC
1913  011050  013737  001546  001122        MOV     NCCSR,$BDADR    ;SET UP REG CSR ADRS
1914  011056  005037  001124                CLR     $GDDAT          ;EXPECTED NO FLAG
1915  011062  012737  020000  001126        MOV     #20000,$BDDAT   ;SHOW IT WAS SET
1916  011070  104022                          ERROR   22              ;INCREMENT OVERFLOW FLAG SET PREMATURELY
1917                                                                  ;OR FAILED TO CLR BY SPECIAL FUNCTION ON
1918  011072  000420                          BR      TST51           ;;NEXT TEST ON ER
1919  011074  012737  040000  013226  6$:    MOV     #40000,XYHOLD   ;SET UP XY HOLD LOC FOR TYPE
1920  011102  012737  020000  001120        MOV     #20000,$GDADR   ;SET UP ADRS OF NPR
1921  011110  010137  001124                MOV     R1,$GDDAT       ;LD EXPECTED
1922  011114  013737  020000  001126        MOV     @#20000,$BDDAT  ;GET LOCATION AFTER INCREMENT
1923  011122  023737  001124  001126        CMP     $GDDAT,$BDDAT   ;CORRECT?
1924  011130  001654                          BEQ     1$              ;BR IF SO - DO ANOTHER ADRS MAKE
1925  011132  104024                          ERROR   24              ;EVEN BYTE INCREMENT FAILURE
1926
1927                                  ;;**************************************************************
1928                                  ;*TEST 51        TEST ODD BYTE INCREMENT AND INCREMENT OVERFLOW FLAG
1929                                  ;;**************************************************************
1930  011134  000004                  TST51:  SCOPE
1931  011136  012737  000100  001160          MOV     #100,$TIMES     ;;DO 100 ITERATIONS
1932  011144  012737  011234  001110          MOV     #2$,$LPERR      ;SET UP SCOPE LOOP ADRS
1933  011152  012700  000100                  MOV     #100,R0         ;SET UP OVERFLOW FLAG CLR COUNTER
1934  011156  012777  000100  170376          MOV     #100,@NCSFR     ;CLR ALL
```

```
1935  011164  012777  030000  170356          MOV     #30000,@NCOFF   ;OFFSET BITS 10-13 SHOULD HAVE NO AFFECT AT RES 4
1936  011172  012777  002001  170346          MOV     #2001,@NCCSR    ;SELECT RESOLUTION 2 WITH NPR ENABLE
1937  011200  005037  020000                  CLR     @#20000         ;START LOC 20000 AT ZERO
1938  011204  005037  013230                  CLR     GOOD            ;GOOD CONTAINS EXPECTED NPR DATA
1939  011210  005002                          CLR     R2              ;R2 WHEN NON-ZERO SAYS INCREMENT OVFLO FLAG EXPECTED
1940
1941  011212  105237  013231          1$:     INCB    GOOD+1          ;INCREMENT EXPECTED
1942  011216  001006                          BNE     2$              ;BR IF NO OVERFLOW
1943  011220  105137  013231                  COMB    GOOD+1          ;CAN'T EXCEED 177400
1944  011224  012702  000001                  MOV     #1,R2           ;OVERFLOW FLAG SHOULD SET WHEN CELL 177400
1945  011230  005300                          DEC     R0              ;COUNT CLR FLAG ATTEMPTS, 32 SOFT, 32 HARD
1946  011232  001522                          BEQ     TST52           ;;NEXT TEST IF ALL CLR FLAG ATTEMPTS COMPLETED
1947  011234  012777  000001  170320  2$:     MOV     #1,@NCSFR       ;CLR XY HOLD REG
1948  011242  012777  040001  170302          MOV     #40001,@NCXYH   ;DO AN WORD IN NPR TO 20001
1949  011250  000240                          NOP                     ;STALL
1950  011252  005702                          TST     R2              ;EXPECT OVERFLOW?
1951  011254  001452                          BEQ     5$              ;BR IF NOT - BUT CHECK IT DID NOT SET
1952  011256  032777  020000  170262          BIT     #20000,@NCCSR   ;LOOK FOR OVERFLOW FLAG
1953  011264  001012                          BNE     3$              ;BR IF SET
1954  011266  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET UP CSR REG ADRS
1955  011274  012737  020000  001124          MOV     #20000,$GDDAT   ;EXPECT INCREMENT OVERFLOW FLAG
1956  011302  005037  001126                  CLR     $BDDAT          ;SHOW IT WAS NOT SET
1957  011306  104022                          ERROR   22              ;INCREMENT OVERFLOW FLAG FAILED TO SET
1958  011310  000473                          BR      TST52           ;;NEXT TEST ON ER
1959  011312  020027  000040          3$:     CMP     R0,#40          ;CLR OVERFLOW BY SOFTWARE?
1960  011316  100026                          BPL     4$              ;BR IF SO
1961  011320  005037  020002                  CLR     @#20002         ;INSURE LOC=0
1962  011324  012777  000001  170230          MOV     #1,@NCSFR       ;CLR XY HOLD REG
1963  011332  012777  040002  170212          MOV     #40002,@NCXYH   ;DO A NPR TO LOC 20002 TO CLR OVFLO FLG
1964  011340  032777  020000  170200          BIT     #20000,@NCCSR   ;DID FLAG GET CLEARED BY HARDWARE?
1965  011346  001433                          BEQ     6$              ;BR IF SO
1966  011350  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET NRP NC CSR REG ADRS
1967  011356  005037  001124                  CLR     $GDDAT          ;EXPECTED NO OVERFLOW FLAG
1968  011362  012737  020000  001126          MOV     #20000,$BDDAT   ;SHOW IT DID NOT CLR ON ADRS MAKE
1969  011370  104022                          ERROR   22              ;INCREMENT OVERFLOW FLAG FAILED TO CLR ON
1970                                                                  ;HARDWARE MAKE ADRS
1971  011372  000442                          BR      TST52           ;;NEXT TEST ON ER
1972  011374  012777  000002  170160  4$:     MOV     #2,@NCSFR       ;DO A SPECIAL FUNCTION CLR ON INCR OVFLO FLAG
1973  011402  032777  020000  170136  5$:     BIT     #20000,@NCCSR   ;DID FLAG SET IN ER OR FAIL TO CLR?
1974  011410  001412                          BEQ     6$              ;BR IF OK (CLRED) - CK DATA INC
1975  011412  013737  001546  001122          MOV     NCCSR,$BDADR    ;SET UP REG CSR ADRS
1976  011420  005037  001124                  CLR     $GDDAT          ;EXPECTED NO FLAG
1977  011424  012737  020000  001126          MOV     #20000,$BDDAT   ;SHOW IT WAS SET
1978  011432  104022                          ERROR   22              ;INCREMENT OVERFLOW FLAG SET PREMATURELY
1979                                                                  ;OR FAILED TO CLR BY SPECIAL FUNCTION CLR
1980  011434  000421                          BR      TST52           ;;NEXT TEST ON ER
1981  011436  012737  040001  013226  6$:     MOV     #40001,XYHOLD   ;SET UP XY HOLD LOC FOR TYPE
1982  011444  012737  020001  001122          MOV     #20001,$BDADR   ;SET UP ADRS OF NPR
1983  011452  013737  013230  001124          MOV     GOOD,$GDDAT     ;LD EXPECTED
1984  011460  013737  020000  001126          MOV     @#20000,$BDDAT  ;GET LOCATION AFTER INCREMENT
1985  011466  023737  001124  001126          CMP     $GDDAT,$BDDAT   ;CORRECT?
1986  011474  001646                          BEQ     1$              ;BR IF SO - DO ANOTHER ADRS MAKE
1987  011476  104024                          ERROR   24              ;ODD BYTE INCREMENT FAILURE
1988
1989                          ;;*****************************************************************
1990                          ;*TEST 52        TEST THAT INCREMENT OVERFLOW FLAG WILL INTERRUPT
```

```
1991                                    ;;*****************************************************************
1992  011500  000004                    TST2:   SCOPE
1993  011502  012737  000004  001160            MOV     #4,$TIMES       ;;DO 4 ITERATIONS
1994  011510  012737  000340  177776            MOV     #PR7,PSW        ;SET PRIORITY TO HIGHEST
1995  011516  012777  011644  170040            MOV     #2$,@NCVCT0     ;SET UP INTR RETURN ADRS
1996  011524  012777  000340  170034            MOV     #340,@NCVCT2    ;PRIORITY TO 7 ON INTR
1997  011532  005000                            CLR     R0              ;R0 DETERMINES WHEN INTR EXPECTED
1998  011534  012777  000103  170020            MOV     #103,@NCSFR     ;CLR ALL
1999  011542  012777  020000  170000            MOV     #20000,@NCOFF   ;SET UP OFFSET TO 20000
2000  011550  012777  001101  167770            MOV     #1101,@NCCSR    ;SELECT RESOLUTION ?  NPR + OVFLO ENABLE
2001  011556  005037  020000                    CLR     @#20000         ;START CELL AT 0
2002
2003  011562  012777  000001  167772  1$:       MOV     #1,@NCSFR       ;CLR XY HOLD REG
2004  011570  012777  000000  167754            MOV     #0,@NCXYH       ;DO AN ADRS MAKE TO LOC 20000
2005  011576  012737  000300  177776            MOV     #PR6,PSW        ;ALLOW AN INTR ON INCREMENT OVFLO
2006  011604  013737  001546  001122            MOV     NCCSR,$BDADR    ;SET UP CSR REG ADRS
2007  011612  005200                            INC     R0              ;COUNT ALONG WITH LOC 20000
2008  011614  001362                            BNE     1$              ;OVERFLOW OCCURS INCREMENTING AT 177777
2009  011616  012737  020000  001124            MOV     #20000,$GDDAT   ;SHOULD NEVER GET HERE - EXPECT OVFLO FLG
2010  011624  017737  167716  001126            MOV     @NCCSR,$BDDAT   ;READ CSR
2011  011632  042737  157777  001126            BIC     #157777,$BDDAT  ;SAVE INCREMENT OVERFLOW FLAG
2012  011640  104022                            ERROR   22              ;INCREMENT OVERFLOW FLAG FAILED TO INTERRUPT
2013  011642  000430                            BR      4$              ;RESTORE VECTOR LOCS
2014  011644  022626                    2$:     CMP     (SP)+,(SP)+     ;FIX STACK SINCE NO RTI
2015  011646  020027  177777                    CMP     R0,#-1          ;EXPECT INTR ON COUNT OF 177777
2016  011652  001412                            BEQ     3$              ;BR IF EXPECTED
2017  011654  005037  001124                    CLR     $GDDAT          ;LD EXPECTED
2018  011660  017737  167662  001126            MOV     @NCCSR,$BDDAT   ;READ CSR
2019  011666  042737  157777  001126            BIC     #157777,$BDDAT  ;SAVE ONLY INCREMENT OVFLO FLAG
2020  011674  104022                            ERROR   22              ;PREMATURE INCREMENT OVERFLOW INTR
2021  011676  000412                            BR      4$              ;RESTORE VECTOR LOCS
2022  011700  012737  020000  001124  3$:       MOV     #20000,$GDDAT   ;LD EXPECTED
2023  011706  032777  020000  167632            BIT     #20000,@NCCSR   ;LOOK FOR INCREMENT OVERFLOW FLAG
2024  011714  001003                            BNE     4$              ;BR IF SET
2025  011716  005037  001126                    CLR     $BDDAT          ;SHOW IT WAS NOT SET
2026  011722  104022                            ERROR   22              ;INCREMENT OVFLO FLAG NOT SET ON INTR
2027  011724  013777  001566  167632  4$:       MOV     NCVCT2,@NCVCT0  ;RESTORE VECTOR PTR
2028  011732  005077  167630                    CLR     @NCVCT2         ;RESTORE HALT
2029
2030                                    ;;*****************************************************************
2031                                    ;*TEST 53        TEST THAT TIMEOUT WILL SET THEN CLR WITH SPECIAL FUNCTION CLR
2032                                    ;;*****************************************************************
2033  011736  000004                    TST53:  SCOPE
2034  011740  012777  000100  167614            MOV     #100,@NCSFR     ;CLR ALL
2035  011746  012777  140003  167574            MOV     #140003,@NCOFF  ;SET UPPER ADRS BITS 14,15,16,17 (740000)
2036  011754  012777  002001  167564            MOV     #2001,@NCCSR    ;SELECT RES4 WITH NPR ENABLE
2037  011762  012777  040000  167562            MOV     #40000,@NCXYH   ;TRY A BYTE INC TO LOC (760000) RESERVED TIMEOUT ADRS
2038  011770  013737  001546  001122            MOV     NCCSR,$BDADR    ;SET UP CSR ADRS
2039  011776  012737  100000  001124            MOV     #100000,$GDDAT  ;EXPECT THE TIMEOUT BIT
2040  012004  005777  167536                    TST     @NCCSR          ;SEE IF SET
2041  012010  100403                            BMI     1$              ;BR IF SO
2042  012012  005037  001126                    CLR     $BDDAT          ;SHOW IT DID NOT SET
2043  012016  104016                            ERROR   16              ;TIMEOUT FAILED TO SET AT ADRS 760000
2044  012020  012777  000100  167534  1$:       MOV     #100,@NCSFR     ;NOW USE SPECIAL FUNCTION CLR ON TIMEOUT FLAG
2045  012026  005777  167514                    TST     @NCCSR          ;LOOK FOR IT CLEARED
2046  012032  100006                            BPL     2$              ;BR IF SO
```

```
2047  012034  005037  001124                  CLR     $GDDAT          ;EXPECTED ZERO
2048  012040  012737  100000  001126          MOV     #BIT15,$BDDAT   ;SHOW IT WAS STILL SET
2049  012046  104016                          ERROR   16              ;TIMEOUT FLAG FAILED TO CLR WITH SPECIAL FUNCTION
2050  012050  000137  012560          2$:     JMP     $EOP            ;GO REPORT 'ENDPASS'
2051
2052  012054                          TST54:
2053                                  ;;*****************************************************************
2054                                  ;*TEST 54       TEST THAT TIMING MARK SETS IN LIST MODE - START 204
2055                                  ;;*****************************************************************
2056  012054  000240                  TST54:  <NOP>
2057  012056  012737  012064  001106          MOV     #10$,$LPADR     ;;SET SCOPE LOOP ADDRESS
2058  012064  012737  012064  001110  10$:    MOV     #10$,$LPERR     ;SET UP SCOPE LOOP ADRS
2059  012072  012700  002000                  MOV     #2000,R0        ;DO CHECK 2000 TIMES (8)
2060  012076  013737  001552  001122  1$:     MOV     NCADR,$BDADR    ;SET UP ADRS REG ADRS
2061  012104  012777  000102  167450          MOV     #102,@NCSFR     ;CLEAR INTERFACE
2062  012112  012777  000400  167426          MOV     #400,@NCCSR     ;SELECT LIST MD
2063  012120  012777  000000  167424          MOV     #0,@NCXYH       ;CLR ADRS REG
2064  012126  012777  000102  167426          MOV     #102,@NCSFR     ;CLR ALL
2065  012134  012777  000040  167420          MOV     #40,@NCSFR      ;SET TIME MARK
2066  012142  012777  000422  167376          MOV     #422,@NCCSR     ;LIST MD, EXT INPUT, ENABLE ADC
2067  012150  052777  000004  167404  2$:     BIS     #4,@NCSFR       ;START ADC'S
2068  012156  004737  012702                  JSR     PC,DELAY        ;GO STALL FOR A/D DONE
2069  012162  017737  167364  001126          MOV     @NCADR,$BDDAT   ;GET ADRS MADE
2070  012170  001002                          BNE     3$              ;BR IF ADRS MADE
2071  012172  104025                          ERROR   25              ;CHECK THE FOLLOWING, THAT THE JOYSTICK
2072                                                                  ;IS APPROX. CENTERED, THAT BOTH ADC'S CONVERTED
2073  012174  000435                          BR      TST55           ;;NEXT TEST ON ER
2074  012176  042737  077777  001126  3$:     BIC     #77777,$BDDAT   ;SAVE ONLY TIME MARK ADRS BIT
2075  012204  100405                          BMI     4$              ;BR IF SET
2076  012206  012737  100000  001124          MOV     #100000,$GDDAT  ;SHOW IT WAS EXPECTED
2077  012214  104026                          ERROR   26              ;ADRS MAKER FAILED TO RECOGNIZE
2078                                                                  ;THE TIMING MARK FLOP
2079  012216  000424                          BR      TST55           ;;NEXT TEST ON ER
2080  012220  052777  000004  167334  4$:     BIS     #4,@NCSFR       ;START ADC'S
2081  012226  004737  012702                  JSR     PC,DELAY        ;GO STALL FOR A/D DONE
2082  012232  017737  167314  001126          MOV     @NCADR,$BDDAT   ;GET ADRS MADE
2083  012240  001002                          BNE     5$              ;BR IF ADRS MADE
2084  012242  104025                          ERROR   25              ;AGAIN, CHECK THAT THE JOYSTICK
2085                                                                  ;IS APPROX CENTERED, THAT BOTH ADC'S CONVERTED
2086  012244  000411                          BR      TST55           ;;NEXT TEST ON ER
2087  012246  042737  077777  001126  5$:     BIC     #77777,$BDDAT   ;SAVE TIME MARK ADRS BIT ONLY
2088  012254  100003                          BPL     6$              ;OK - TIME MARK CLRED ON NEXT ADC DONE
2089  012256  005037  001124                  CLR     $GDDAT          ;EXPECTED NO TIME MARK
2090  012262  104026                          ERROR   26              ;ADC DONE FAILED TO CLR TIME MARK
2091  012264  005300                  6$:     DEC     R0              ;DO 2000 TIMES
2092  012266  001303                          BNE     1$              ;AGAIN TILL DONE
2093
2094                                  ;;*****************************************************************
2095                                  ;*TEST 55       TEST JOYSTICK ADC CONVERT, NO CONVERT & JOYSTICK LOGIC
2096                                  ;;*****************************************************************
2097  012270  000004                  TST55:  SCOPE
2098  012272  012700  000001          1$:     MOV     #1,R0           ;R0 COUNTS NO. OF ITEMS PER LINE
2099  012276  012777  000102  167256          MOV     #102,@NCSFR     ;CLR INTERFACE
2100  012304  012777  000032  167234          MOV     #32,@NCCSR      ;SELECT EXTERNAL, JOYSTICK, ENAB ADC
2101  012312  005300                  2$:     DEC     R0              ;LOOK FOR LINEFEED & CARRIAGE RETURN
2102  012314  001010                          BNE     3$              ;BR IF NOT NEEDED
```

```
2103  012316  012700  000006                    MOV     #6,R0           ;RESET TYPE COUNTER
2104  012322  032777  020000  166610  10$:      BIT     #BIT13,@SWR     ;ARE WE TYPING?
2105  012330  001002                            BNE     3$              ;BR IF NOT
2106  012332  104400  070000                    TYPE    .MSG2           ;GO TYPE 'LF+CR'
2107  012336  104406                    3$:     CKSWR                   ;GO LOOK FOR 'CONTROL-G'
2108  012340  012777  000004  167214            MOV     #4,@NCSFR       ;DO A CONVERSION
2109  012346  004737  012702                    JSR     PC,DELAY        ;GO STALL FOR A/D DONE
2110  012352  105777  167170                    TSTB    @NCCSR          ;SHOULD NOT BE BUSY
2111  012356  100016                            BPL     4$              ;BR IF DONE
2112  012360  032777  040000  167160            BIT     #40000,@NCCSR   ;NO CONVERT SHOULD BE SET IF STILL BUSY
2113  012366  001032                            BNE     5$              ;BR IF "NO CONVERT" FLAG IS SET
2114  012370  013737  001546  001122            MOV     NCCSR,$BDADR    ;SET UP REG CSR ADRS
2115  012376  012737  040000  001124            MOV     #40000,$GDDAT   ;EXPECTED NO CONVERT WHEN BUSY STILL SET
2116  012404  005037  001126                    CLR     $BDDAT          ;SHOW IT WAS NOT SET
2117  012410  104027                            ERROR   27              ;NO CONVERT FLAG NOT SET WHEN BUSY SET
2118  012412  000727                            BR      1$              ;GO CONVERT AGAIN
2119
2120  012414  017737  167132  013222  4$:       MOV     @NCXYH,XYAD     ;READ CONVERTED VALUES
2121  012422  012777  000010  167132            MOV     #10,@NCSFR      ;CLR JOYSTICK DEPRESS BUTTON
2122  012430  005777  167112                    TST     @NCCSR          ;IS BUTTON DEPRESSED?
2123  012434  100016                            BPL     6$              ;BR IF NOT
2124  012436  032777  020000  166474            BIT     #BIT13,@SWR     ;TYPE "BAR DN"?
2125  012444  001326                            BNE     10$             ;BR IF NO TYPE
2126  012446  104400  070003                    TYPE    ,MSG3           ;TYPE "BAR DN"
2127  012452  000717                            BR      2$              ;GO CONVERT AGAIN
2128
2129  012454  032777  020000  166456  5$:       BIT     #BIT13,@SWR     ;TYPE "NO CNVRT" MESSAGE?
2130  012462  001317                            BNE     10$             ;BR IF NOT
2131  012464  104400  070013                    TYPE    ,MSG4           ;TYPE "NO CNVRT"
2132  012470  000710                            BR      2$              ;GO CONVERT AGAIN
2133
2134  012472  005037  013216          6$:       CLR     XADC            ;CLR X LOC
2135  012476  005037  013220                    CLR     YADC            ;CLR Y LOC
2136  012502  113737  013222  013216            MOVB    XYAD,XADC       ;GET X CONVERSION
2137  012510  113737  013223  013220            MOVB    XYAD+1,YADC     ;GET Y CONVERSION
2138  012516  032777  020000  166414            BIT     #BIT13,@SWR     ;TYPE X+Y DATA?
2139  012524  001276                            BNE     10$             ;BR IF NOT
2140  012526  104400  070025                    TYPE    ,MSG5           ;TYPE X=
2141  012532  013746  013216                    MOV     XADC,-(SP)      ;;SAVE XADC FOR TYPEOUT
2142                                                                    ;;TYPE X VALUE
2143  012536  104402                            TYPOS                   ;;GO TYPE--OCTAL ASCII
2144  012540     003                            .BYTE   3               ;;TYPE 3 DIGIT(S)
2145  012541     001                            .BYTE   1               ;;TYPE LEADING ZEROS
2146  012542  104400  070031                    TYPE    ,MSG6           ;TYPE Y=
2147  012546  013746  013220                    MOV     YADC,-(SP)      ;;SAVE YADC FOR TYPEOUT
2148                                                                    ;;TYPE Y VALUE
2149  012552  104402                            TYPOS                   ;;GO TYPE--OCTAL ASCII
2150  012554     003                            .BYTE   3               ;;TYPE 3 DIGIT(S)
2151  012555     001                            .BYTE   1               ;;TYPE LEADING ZEROS
2152  012556  000655                            BR      2$              ;GO CONVERT AGAIN
```

```
                                .SBTTL  END OF PASS ROUTINE

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

012560                          $EOP:
012560  000004                          SCOPE
012562  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
012566  005037  001160                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
012572  005237  001202                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
012576  042737  100000  001202          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
012604  005327                          DEC     (PC)+           ;;LOOP?
012606  000001                  $EOPCT: .WORD   1
012610  003022                          BGT     $DOAGN          ;;YES
012612  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
012614  000001                  $ENDCT: .WORD   1
012616  012606                          $EOPCT
012620  104400  012665                  TYPE    ,$ENDMG         ;;TYPE "END PASS #"
012624  013746  001202                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
012630  104404                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
012632  104400  012662                  TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
012636  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
012642  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
012644  000005                          RESET                   ;;CLEAR THE WORLD
012646  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
012650  000240                          NOP                     ;;SAVE ROOM
012652  000240                          NOP                     ;;FOR
012654  000240                          NOP                     ;;ACT11
012656                          $DOAGN:
012656  000137                          JMP     @(PC)+          ;;RETURN
012660  002310                  $RTNAD: .WORD   RESTRT
012662     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
012665     015  042412  042116  $ENDMG: .ASCIZ  <15><12>/END PASS #/
012672  050040  051501  020123
012700  000043
```

```
2191                                    ;;*****************************************************************
2192                                    ;THIS IS A COMMON DELAY ROUTINE OF A COUNT OF 10
2193                                    ;;*****************************************************************
2194  012702  012705  000010           DELAY:  MOV     #10,R5          ;SET UP COUNT IN R5
2195  012706  005305                   1$:     DEC     R5              ;COUNT AWAY
2196  012710  001376                           BNE     1$              ;TILL DONE
2197  012712  000207                           RTS     PC              ;RETURN WHEN DONE
2198
2199                                    ;;*****************************************************************
2200                                    ;THIS SUBROUTINE LOADS INTERFACE REGISTERS FROM SOFTWARE REGS
2201                                    ;IT ALSO INITS THE SOFTWARE ADC'S
2202                                    ;;*****************************************************************
2203  012714  013777  013234  166626   REGLD:  MOV OFFSET,@NCOFF       ;LOAD OFFSET REGISTER
2204  012722  053737  013232  013236           BIS TAM,SOFCSR          ;SET STAGE II ENABLE IF WARRANTED
2205  012730  013777  013236  166610           MOV SOFCSR,@NCCSR       ;SET UP COMMAND REGISTER
2206  012736  012737  000200  013214           MOV     #200,XYPTRN     ;START ADC'S WITH FLOATING ONE
2207  012744  000207                           RTS PC                  ;RETURN TO ADRS MAKE TEST
2208
2209                                    ;;*****************************************************************
2210                                    ;THIS IS A ROUTINE TO SIMULATE 2 ADC'S
2211                                    ;ON EACH CALL THE ONE OR ZERO PATTERN IS SHIFTED
2212                                    ;RIGHT AND LOADED INTO THE SOFTWARE XADC & YADC -
2213                                    ;FIRST A ONE IS FLOATED RIGHT AND THEN A ZERO - WHEN THE
2214                                    ;ZERO HAS PASSED THRU BIT 0 OF THE ADC'S THE RETURN IS TO CALL
2215                                    ;+4 - ALL OTHER RETURNS ARE TO CALL+6.
2216                                    ;THE XYHOLD LOC CONTAINS THE X/Y ADC VALUES AS WRITTEN
2217                                    ;TO THE XYHOLD NC11 REG.
2218                                    ;THE ADDRESS IS THEN MADE IN SOFTWARE AND THEN IN HARDWARE
2219                                    ;SOFTWARE ADDRESS IS RETURNED IN $GDADR
2220                                    ;HARDWARE ADDRESS RETURNED IN $BDADR
2221                                    ;WHEN NPR (TAM=1) IS ENABLED THE CONTENTS OF THE NPR'ED LOC
2222                                    ;ARE SAVED IN 'LOCSAV' AND RESTORED AFTER THE NPR.
2223                                    ;THE MEMORY LOC AFTER THE NPR IS PUT IN $BDDAT
2224                                    ;THE EXPECTED NPR DATA IS PUT IN $GDDAT
2225                                    ;;*****************************************************************
2226  012746  062716  000004           MAKE:   ADD     #4,(SP)         ;RETURN AFTER THE SCOPE LOOP JSR
2227  012752  006237  013214                   ASR     XYPTRN          ;SHIFT ONE/ZERO RIGHT
2228  012756  001003                           BNE     1$              ;BR IF NOT DONE WITH ZERO YET
2229  012760  012737  177677  013214           MOV     #177677,XYPTRN  ;NOW SET UP FLOATING ZERO PATTERN
2230  012766  022737  177777  013214   1$:     CMP     #-1,XYPTRN      ;ALL DONE FLOATING ZERO?
2231  012774  001505                           BEQ     RNGDN           ;BR IF SO
2232  012776  013737  013214  013216           MOV     XYPTRN,XADC     ;SET UP X VALUE
2233  013004  042737  177600  013216           BIC     #177600,XADC    ;LOW ORDER 7 BITS ONLY
2234  013012  013737  013216  013220           MOV     XADC,YADC       ;GOES TO Y VALUE ALSO
2235  013020  113737  013216  013226   SAMDN:  MOVB XADC,XYHOLD        ;FORM THE XYHOLD WORD
2236  013026  005737  013212                   TST DUAL                ;B GAMMA SET?
2237  013032  001403                           BEQ NODUAL              ;BR IF NOT
2238  013034  052737  000200  013226           BIS #200,XYHOLD         ;SET B GAMMA IN XYHOLD WORD
2239
2240  013042  113737  013220  013227   NODUAL: MOVB YADC,XYHOLD+1      ;SET UP YADC IN XYHOLD WORD
2241  013050  062716  000002           MKLOOP: ADD     #2,(SP)         ;RETURN TO CALL +6 TO CK DATA
2242                                            ;OR +2 IF ENTERED HERE FROM SCOPE LOOP
2243  013054  012777  000001  166500           MOV #1,@NCSFR           ;CLEAR XYHOLD
2244  013062  004737  013242                   JSR PC,SAMAK            ;DO A SOFTWARE ADDRESS MAKE
2245  013066  013737  013230  001120           MOV     GOOD,$GDADR     ;GET MAKE ADRS
2246  013074  005737  013232                   TST     TAM             ;ARE WE DOING NPR'S
```

```
2247  013100  001423                          BEQ     2$              ;BR IF NOT
2248  013102  042737  000001  013230          BIC     #1,GOOD         ;MAKE INTO WORD ADRS
2249  013110  017737  000114  013240          MOV     @GOOD,LOCSAV    ;SAVE LOCATION OF NPR
2250  013116  005077  000106                  CLR     @GOOD           ;ZERO LOCATION
2251  013122  032737  000001  001120          BIT     #1,$GDADR       ;LOOK FOR BYTE INCREMENT
2252  013130  001404                          BEQ     1$              ;BR IF LOW BYTE
2253  013132  012737  000400  001124          MOV     #400,$GDDAT     ;HI BYTE INCREMENT
2254  013140  000403                          BR      2$              ;SKIP
2255  013142  012737  000001  001124  1$:     MOV     #1,$GDDAT       ;LOW BYTE INCREMENT
2256  013150  013777  013226  166374  2$:     MOV     XYHOLD,@NCXYH   ;DO A HARDWARE ADRS MAKE
2257  013156  021616                          CMP     (SP),(SP)       ;STALL
2258  013160  017737  166366  001122          MOV     @NCADR,$BDADR   ;READ THE HARDWARE ADRS MADE
2259  013166  005737  013232                  TST     TAM             ;DOING NPR'S?
2260  013172  001406                          BEQ     RNGDN           ;BR IF NOT
2261  013174  017737  000030  001126          MOV     @GOOD,$BDDAT    ;READ THE CONTENTS OF THE NPR'ED ADRS
2262  013202  013777  013240  000020          MOV     LOCSAV,@GOOD    ;RESTORE THE NPR LOC WITH PREVIOUS CONTENTS
2263  013210  000207                  RNGDN:  RTS     PC              ;NOW RETURN TO CK MAKE ADRS OR NPR DATA
2264  013212  000000                  DUAL:   0                       ;NON-ZERO INDICATES B GAMMA BEING TESTED
2265  013214  000200                  XYPTRN: 200                     ;CONTAINS PATTERN FOR X/Y SOFT ADC'S
2266  013216  000000                  XADC:   0                       ;SOFTWARE X ADC
2267  013220  000000                  YADC:   0                       ;SOFTWARE Y ADC
2268  013222  000000                  XYAD:   0                       ;HARDWARE XY DATA READ
2269  013224  000000                  XTEMP:  0                       ;USED IN BUILDING SOFTWARE ADDRESS MAKE
2270  013226  000000                  XYHOLD: 0                       ;SOFTWARE XY ADC VALUE WRITTEN TO XYHOLD REG
2271  013230  000000                  GOOD:   0                       ;MEMORY ADRS OF ADRS MAKE NPR
2272  013232  000000                  TAM:    0                       ;NON-ZERO INDICATES NPR ENABLED TO ADRS MADE
2273  013234  000000                  OFFSET: 0                       ;OFFSET ADRS BITS LOADED TO OFFSET REG
2274  013236  000000                  SOFCSR: 0                       ;CSR BITS WRITTEN TO NC CSR
2275  013240  000000                  LOCSAV: 0                       ;ALL SINGLE LOCS ARE SAVED & RESTORED FROM THIS LOC (NPR
2276
2277                                  ;;*****************************************************************
2278                                  ;THIS CODE DETERMINES WHAT RESOLUTION IS SELECTED AND
2279                                  ;THEN DIRECTS THE PROGRAM TO THE SOFTWARE ADDRESS MAKER
2280                                  ;ROUTINE FOR THAT RESOLUTION - EACH ROUTINE ALIGNS THE
2281                                  ;X & Y ADC VALUES TO THERE PROPER ADDRESS BIT ASSIGNMENTS
2282                                  ;AND THEN ADDS IN THE OFFSET ADDRESS BITS ACCORDING TO THE
2283                                  ;LOGIC OF THAT RESOLUTION - THE ADDRESS MADE IS RETURNED IN 'GOOD'
2284                                  ;;*****************************************************************
2285  013242  005037  013230          SAMAK:  CLR GOOD                ;CLEAR FINAL ADDRESS
2286  013246  032737  000010  013236          BIT #10,SOFCSR          ;JOYSTICK MODE?
2287  013254  001042                          BNE JSM                 ;YES
2288  013256  113737  013216  013230  SDCM:   MOVB XADC,GOOD          ;SET UP FOR FORMATION
2289  013264  013700  013236                  MOV SOFCSR,R0           ;STORE SOFCSR IN R0 FOR SPEED
2290  013270  032700  003400                  BIT #3400,R0            ;CHECK FOR RES0
2291  013274  001444                          BEQ RES00
2292  013276  032700  003000                  BIT #3000,R0            ;CHECK FOR RES1
2293  013302  001444                          BEQ RES01
2294  013304  032700  002400                  BIT #2400,R0            ;CHECK FOR RES2
2295  013310  001445                          BEQ RES02
2296  013312  032700  002000                  BIT #2000,R0            ;CHECK FOR RES3
2297  013316  001524                          BEQ RES03
2298  013320  032700  001400                  BIT #1400,R0            ;CHECK FOR RES4
2299  013324  001410                          BEQ RES04X
2300  013326  032700  001000                  BIT #1000,R0            ;CHECK FOR RES5
2301  013332  001407                          BEQ RES05X
2302  013334  032700  000400                  BIT #400,R0             ;CKECK FOR RES6
```

```
2303  013340  001406                          BEQ RES06X
2304  013342  000137  014260                  JMP RES07                ;MUST BE RES7
2305  013346  000137  013716        RES04X:   JMP RES04                ;JUMP TO SOFTWARE ADRS MAKER RES4
2306  013352  000137  014014        RES05X:   JMP RES05                ;JUMP TO SOFTWARE ADRS MAKER RES5
2307  013356  000137  014136        RES06X:   JMP RES06                ;JUMP TO SOFTWARE ADRS MAKER RES6
2308
2309                                ;;*****************************************************************
2310                                ;MAKE ADDRESS A LA JOY STICK MODE
2311                                ;BITS 1-7 OF X AND Y IN THEIR RESPECTIVE BYTES
2312                                ;;*****************************************************************
2313  013362  113737  013216  013230  JSM:     MOVB XADC,GOOD           ;GET X VALUE
2314  013370  113737  013220  013231          MOVB YADC,GOOD+1         ;GET Y VALUE
2315  013376  042737  100200  013230          BIC #100200,GOOD         ;CLEAR BITS 15 AND 7
2316  013404  000207                          RTS PC
2317
2318                                ;;*****************************************************************
2319                                ;MAKE ADDRESS A LA RESOLUTION 0
2320                                ;CLEAR ADDRESS REGISTER
2321                                ;;*****************************************************************
2322  013406  005037  013230        RES00:    CLR GOOD                 ;NULL - ALWAYS EXPECT ZERO
2323  013412  000207                          RTS PC
2324
2325                                ;;*****************************************************************
2326                                ;MAKE ADDRESS A LA RESOLUTION 01
2327                                ;BYTE 0 CONTAINS 8 BITS X, BYTE 1 CONTAINS 8 BITS Y
2328                                ;;*****************************************************************
2329  013414  113737  013220  013231  RES01:   MOVB YADC,GOOD+1         ;LOAD X&Y DIRECTLY WITH NO OFFSET
2330  013422  000207                          RTS PC
2331
2332                                ;;*****************************************************************
2333                                ;MAKE ADDRESS A LA RESOLUTION 2
2334                                ;64X64X16
2335                                ;0   0   0   Y  Y  Y  Y  Y  Y  X  X  X  X  X  X  0
2336                                ;15  14  13  6  5  4  3  2  1  6  5  4  3  2  1
2337                                ;;*****************************************************************
2338  013424  004737  013560        RES02:    JSR PC,M64X16            ;DO ROTATION AND STRIP
2339  013430  013737  013230  013224          MOV GOOD,XTEMP           ;SAVE X
2340  013436  113737  013220  013230          MOVB YADC,GOOD           ;GET Y
2341  013444  004737  013560                  JSR PC,M64X16            ;GET RID OF UNUSED BITS
2342  013450  000337  013230                  SWAB GOOD                ;POSITION Y
2343  013454  006237  013230                  ASR GOOD                 ;TO EXACT POSITION
2344  013460  006237  013230                  ASR GOOD                 ;
2345  013464  063737  013224  013230          ADD XTEMP,GOOD           ;GET SAVED X
2346  013472  013737  013234  013224          MOV OFFSET,XTEMP         ;ACCOUNT FOR OFFSET
2347  013500  042737  017777  013224          BIC #17777,XTEMP         ;ONLY BITS 15 14 13
2348  013506  005737  013212                  TST DUAL                 ;G GAM SET?
2349  013512  001416                          BEQ ND2                  ;BR IF NOT
2350  013514  032737  020000  013224          BIT     #BIT13,XTEMP     ;OFFSET 13 SET?
2351  013522  001407                          BEQ     1$               ;BR IF NOT
2352  013524  052737  040000  013224          BIS     #BIT14,XTEMP     ;OFFSET 13 & B GAM SETS A14
2353  013532  042737  020000  013224          BIC     #BIT13,XTEMP     ;OFFSET 13 & B GAM CLRS A13
2354  013540  000403                          BR      ND2              ;GO ADD OFFSET IN
2355  013542  052737  020000  013224  1$:     BIS     #BIT13,XTEMP     ;OFFSET 13 CLRED & B GAM SETS A13
2356  013550  063737  013224  013230  ND2:    ADD XTEMP,GOOD           ;ADD IN OFFSET
2357  013556  000207                          RTS PC                   ;
2358  013560  042737  177601  013230  M64X16: BIC #177601,GOOD         ;CLEAN GARBAGE
```

```
2359  013566  000207                        RTS PC                  ;
2360
2361                                  ;;*****************************************************************
2362                                  ;MAKE ADDRESS A LA RESOLUTION 3
2363                                  ;32X32X16
2364                                  ;0   0   0   0   0  Y  Y  Y  Y  Y  X  X  X  X  X  0
2365                                  ;15  14  13  12  11 6  5  4  3  2  6  5  4  3  2
2366                                  ;;*****************************************************************
2367  013570  004737  013702         RES03:  JSR PC,M32X16            ;SHIFT AND STRIP
2368  013574  013737  013230  013224         MOV GOOD,XTEMP           ;SAVE X
2369  013602  113737  013220  013230         MOVB YADC,GOOD           ;GET Y
2370  013610  004737  013702                JSR PC,M32X16            ;GO ALIGN + DELETE UNUSED BITS
2371  013614  000337  013230                SWAB GOOD                ;POSITION Y
2372  013620  006237  013230                ASR GOOD                 ;TO EXACT POSITION
2373  013624  006237  013230                ASR GOOD                 ;
2374  013630  006237  013230                ASR GOOD                 ;
2375  013634  063737  013224  013230         ADD XTEMP,GOOD           ;ADD X VALUE
2376  013642  013737  013234  013224         MOV OFFSET,XTEMP         ;GET OFFSET
2377  013650  042737  003777  013224         BIC #3777,XTEMP          ;USE BITS 15 14 13 12 11 OF OFFSET
2378  013656  005737  013212                TST DUAL                 ;B GAMA SET?
2379  013662  001403                        BEQ ND3                  ;BR IF NOT
2380  013664  052737  004000  013224         BIS #4000,XTEMP          ;B GAMA SETS BIT 11
2381  013672  063737  013224  013230  ND3:   ADD XTEMP,GOOD           ;NOW ADD IN OFFSET
2382  013700  000207                        RTS PC                   ;
2383  013702  006237  013230         M32X16: ASR GOOD                ;ALIGN
2384  013706  042737  177701  013230         BIC #177701,GOOD         ;RID UNUSED BITS
2385  013714  000207                        RTS PC                   ;
2386
2387                                  ;;*****************************************************************
2388                                  ;MAKE ADDRESS A LA RESOLUTION 4
2389                                  ;128X128X8 BASE 12K
2390                                  ;0   0    Y   Y  Y  Y  Y  Y  Y  X  X  X  X  X  X  X
2391                                  ;15  14   6   5  4  3  2  1  0  6  5  4  3  2  1  0
2392                                  ;;*****************************************************************
2393  013716  004737  014004         RES04:  JSR PC,M128X8            ;GO DELETE UNUSED BITS
2394  013722  013737  013230  013224         MOV GOOD,XTEMP           ;SAVE X
2395  013730  113737  013220  013230         MOVB YADC,GOOD           ;GET Y
2396  013736  004737  014004                JSR PC,M128X8            ;GO DELETE UNUSED
2397  013742  000337  013230                SWAB GOOD                ;POSITION Y
2398  013746  006237  013230                ASR GOOD                 ;TO EXACT POSITION
2399  013752  063737  013224  013230         ADD XTEMP,GOOD           ;ADD IN X VALUE
2400  013760  013737  013234  013224         MOV OFFSET,XTEMP         ;GET OFFSET
2401  013766  042737  037777  013224         BIC #37777,XTEMP         ;OFFSET BIT 15,14 ONLY
2402  013774  063737  013224  013230         ADD XTEMP,GOOD           ;ADD IN OFFSET
2403  014002  000207                        RTS     PC               ;
2404  014004  042737  177600  013230  M128X8: BIC #177600,GOOD        ;RID UNUSED BITS
2405  014012  000207                        RTS PC                   ;
2406
2407                                  ;;*****************************************************************
2408                                  ;MAKE ADDRESS A LA RESOLUTION 5
2409                                  ;128X128X8 BASE 4K
2410                                  ;0   Y    Y  Y  Y  Y  Y  Y  Y  X  X  X  X  X  X  X
2411                                  ;15  6   (6) 5  4  3  2  1  0  6  5  4  3  2  1  0
2412                                  ;;*****************************************************************
2413  014014  004737  014004         RES05:  JSR PC,M128X8            ;GO DELETE UNUSED BITS
2414  014020  013737  013230  013224         MOV GOOD,XTEMP           ;SAVE X
```

```
2415  014026  113737  013220  013230          MOVB YADC,GOOD          ;GET Y
2416  014034  004737  014004                  JSR PC,M128X8           ;GO DELETE UNUSED BITS
2417  014040  000337  013230                  SWAB GOOD               ;POSITION Y
2418  014044  006337  013230                  ASL GOOD                ;TAKE A LOOK AT Y6
2419  014050  006237  013230                  ASR GOOD                ;NOW POSITION
2420  014054  006237  013230                  ASR GOOD                ;
2421  014060  005737  013230                  TST GOOD                ;WAS Y6 SET?
2422  014064  100004                          BPL SET13               ;BR IF NOT
2423  014066  042737  120000  013230          BIC #120000,GOOD        ;YES, ONLY LEAVE 14 SET (Y6=1)
2424  014074  000403                          BR DN05                 ;SKIP NEXT INSTR
2425  014076  052737  020000  013230  SET13:  BIS #20000,GOOD        ;Y6=0 SETS 13
2426  014104  063737  013224  013230  DN05:   ADD XTEMP,GOOD         ;ADD IN X VALUE
2427  014112  013737  013234  013224          MOV OFFSET,XTEMP        ;GET OFFSET
2428  014120  042737  077777  013224          BIC #77777,XTEMP        ;OFFSET BIT 15 ONLY
2429  014126  063737  013224  013230          ADD XTEMP,GOOD          ;ADD IN OFFSET
2430  014134  000207                          RTS PC                  ;
2431
2432                                  ;;*****************************************************************
2433                                  ;MAKE ADDRESS A LA RESOLUTION 6
2434                                  ;64X64X8
2435                                  ;0   0   0   0  Y  Y  Y  Y  Y  Y  X  X  X  X  X  X
2436                                  ;15  14  13  12 6  5  4  3  2  1  6  5  4  3  2  1
2437                                  ;;*****************************************************************
2438  014136  004737  014244          RES06:  JSR PC,M64X8            ;GO ALIGN & STRIP UNUSED BITS
2439  014142  013737  013230  013224          MOV GOOD,XTEMP          ;SAVE X
2440  014150  113737  013220  013230          MOVB YADC,GOOD          ;GET Y
2441  014156  004737  014244                  JSR PC,M64X8            ;GO ALIGN & STRIP UNUSED BITS
2442  014162  000337  013230                  SWAB GOOD               ;POSITION Y
2443  014166  006237  013230                  ASR GOOD                ;TO EXACT POSITION
2444  014172  006237  013230                  ASR GOOD                ;
2445  014176  063737  013224  013230          ADD XTEMP,GOOD          ;ADD IN X VALUE
2446  014204  013737  013234  013224          MOV OFFSET,XTEMP        ;GET OFFSET
2447  014212  042737  007777  013224          BIC #7777,XTEMP         ;ONLY BITS 15,14,13,12
2448  014220  005737  013212                  TST DUAL                ;B GAM SET?
2449  014224  001403                          BEQ ND5                 ;BR IF NOT
2450  014226  052737  010000  013224          BIS #10000,XTEMP        ;YES, B GAM SETS 12
2451  014234  063737  013224  013230  ND6:    ADD XTEMP,GOOD          ;ADD IN OFFSET
2452  014242  000207                          RTS PC                  ;
2453  014244  006037  013230          M64X8:  ROR GOOD                ;ALIGN
2454  014250  042737  177700  013230          BIC #177700,GOOD        ;RID UNUSED BITS
2455  014256  000207                          RTS PC                  ;
2456
2457                                  ;;*****************************************************************
2458                                  ;MAKE ADDRESS A LA RESOLUTION 7
2459                                  ;32X32X8
2460                                  ;0   0   0   0   0   0   Y  Y  Y  Y  Y  X  X  X  X  X
2461                                  ;15  14  13  12  11  10  6  5  4  3  2  6  5  4  3  2
2462                                  ;;*****************************************************************
2463  014260  004737  014372          RES07:  JSR PC,M32X8            ;GO ALIGN & STRIP UNUSED BITS
2464  014264  013737  013230  013224          MOV GOOD,XTEMP          ;SAVE X
2465  014272  113737  013220  013230          MOVB YADC,GOOD          ;GET Y
2466  014300  004737  014372                  JSR PC,M32X8            ;GO ALIGN & SAVE UNUSED BITS
2467  014304  000337  013230                  SWAB GOOD               ;POSITION Y
2468  014310  006237  013230                  ASR GOOD                ;TO EXACT POSITION
2469  014314  006237  013230                  ASR GOOD                ;
2470  014320  006237  013230                  ASR GOOD                ;
```

```
 2471  014324  063737  013224  013230          ADD XTEMP,GOOD          ;ADD IN X
 2472  014332  013737  013234  013224          MOV OFFSET,XTEMP        ;GET OFFSET
 2473  014340  042737  001777  013224          BIC #1777,XTEMP         ;ONLY BITS 15,14,13,12,11,10
 2474  014346  005737  013212                  TST DUAL                ;B GAM SET?
 2475  014352  001403                          BEQ ND7                 ;BR IF NOT
 2476  014354  052737  002000  013224          BIS #2000,XTEMP         ;YES, B GAM SETS 10
 2477  014362  063737  013224  013230  ND7:    ADD XTEMP,GOOD          ;ADD IN OFFSET
 2478  014370  000207                          RTS PC                  ;
 2479  014372  004737  014244          M32X8:  JSR PC,M64X8            ;GO ALIGN & STRIP UNUSED BITS
 2480  014376  006237  013230                  ASR GOOD                ;ALIGN
 2481  014402  000207                          RTS PC                  ;
```

```
                                        .SBTTL  SYSMAC ROUTINES

                                .NLIST  MC,MD,CND
                                .SBTTL  TYPE ROUTINE

                                ;;**********************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         #FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*

014404  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
014410  100002                          BPL     1$              ;;BR IF YES
014412  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
014414  000430                          BR      3$              ;;LEAVE
014416  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
014420  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
014424  122737  000001  001214          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
014432  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
014434  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
014442  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
014444  010037  014454                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
014450  004737  014674                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
014454  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
014456  132737  000040  001215  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
014464  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
014466  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
014470  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
014472  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
014474  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
014476  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
014502  000002                          RTI                     ;;RETURN
014504  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
014510  001430                          BEQ     8$
014512  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
014516  001006                          BNE     5$
014520  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
014522  104400                          TYPE                    ;;TYPE A CR AND LF
014524  001171                          $CRLF
014526  105037  014662                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
014532  000755                          BR      2$              ;;GET NEXT CHARACTER
014534  004737  014616          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
014540  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
014544  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
014546  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
014552  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
```

```
2538  014556  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2539  014560  004737  014616                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
2540  014564  105337  014662                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2541  014570  000770                          BR      7$              ;;LOOP
2542
2543                                  ;HORIZONTAL TAB PROCESSOR
2544
2545  014572  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2546  014576  004737  014616          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2547  014602  132737  000007  014662          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2548  014610  001372                          BNE     9$              ;;TAB STOP
2549  014612  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
2550  014614  000724                          BR      2$              ;;GET NEXT CHARACTER
2551  014616  105777  164326          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2552  014622  100375                          BPL     $TYPEC
2553  014624  116677  000002  164320          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2554  014632  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2555  014640  001003                          BNE     1$              ;;BRANCH IF NO
2556  014642  105037  014662                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2557  014646  000406                          BR      $TYPEX          ;;EXIT
2558  014650  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2559  014656  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
2560  014660  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
2561  014662  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2562  014664  000207          $TYPEX: RTS     PC
2563
2564                                  .SBTTL  APT COMMUNICATIONS ROUTINE
2565
2566                                  ;;**************************************************************
2567  014666  112737  000001  015132  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
2568  014674  112737  000001  015130  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
2569  014702  000403                          BR      $ATYC
2570  014704  112737  000001  015132  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
2571  014712                          $ATYC:
2572  014712  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2573  014714  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2574  014716  105737  015130                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
2575  014722  001450                          BEQ     5$              ;;IF NOT:  BR
2576  014724  122737  000001  001214          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
2577  014732  001031                          BNE     3$              ;;IF NOT:  BR
2578  014734  132737  000100  001215          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
2579  014742  001425                          BEQ     3$              ;;IF NOT:  BR
2580  014744  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
2581  014750  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
2582  014756  005737  001174          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
2583  014762  001375                          BNE     1$              ;;IF NOT:  WAIT
2584  014764  010037  001210                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
2585  014770  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
2586  014772  001376                          BNE     2$
2587  014774  163700  001210                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
2588  015000  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
2589  015002  010037  001212                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
2590  015006  012737  000004  001174          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
2591  015014  000413                          BR      5$
2592  015016  017637  000004  015042  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
2593  015024  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
```

```
                                                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
                                                JSR     PC,$TYPE        ;;CALL TYPE MACRO
                                        4$:     .WORD   0
                                        5$:
                                        10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
                                                BEQ     12$             ;;IF NOT:  BR
                                                TST     $ENV            ;;RUNNING UNDER APT?
                                                BEQ     12$             ;;IF NOT:  BR
                                        11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
                                                BNE     11$             ;;IF NOT:  WAIT
                                                MOV     @4(SP),$FATAL   ;;GET ERROR #
                                                ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
                                                INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
                                        12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
                                                CLRB    $LFLG           ;;CLEAR LOG FLAG
                                                CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
                                                MOV     (SP)+,R1        ;;POP STACK INTO R1
                                                MOV     (SP)+,R0        ;;POP STACK INTO R0
                                                RTS     PC              ;;RETURN
                                        $MFLG:  .BYTE   0               ;;MESSG. FLAG
                                        $LFLG:  .BYTE   0               ;;LOG FLAG
                                        $FFLG:  .BYTE   0               ;;FATAL FLAG
                                                .EVEN
                                        APTSIZE=200
                                        APTENV=001
                                        APTSPOOL=100
                                        APTCSUP=040
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

015134  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
015140  116637  000001  015357          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
015146  112637  015361                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
015152  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
```

```
2650  015156  000406                          BR      $TYPON
2651  015160  112737  000001  015357  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
2652  015166  112737  000006  015361          MOVB    #6,$OMODE+1     ;;SE  FOR SIX(6) DIGITS
2653  015174  112737  000005  015356  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
2654  015202  010346                          MOV     R3,-(SP)        ;;SAVE R3
2655  015204  010446                          MOV     R4,-(SP)        ;;SAVE R4
2656  015206  010546                          MOV     R5,-(SP)        ;;SAVE R5
2657  015210  113704  015361                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
2658  015214  005404                          NEG     R4
2659  015216  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
2660  015222  110437  015360                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
2661  015226  113704  015357                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2662  015232  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
2663  015236  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
2664  015240  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2665  015242  000404                          BR      3$              ;;GO DO MSB
2666  015244  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
2667  015246  006105                          ROL     R5
2668  015250  006105                          ROL     R5
2669  015252  010503                          MOV     R5,R3
2670  015254  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2671  015256  105337  015360                  DECB    $OMODE          ;;TYPE THIS DIGIT?
2672  015262  100016                          BPL     7$              ;;BR IF NO
2673  015264  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
2674  015270  001002                          BNE     4$              ;;TEST FOR 0
2675  015272  005704                          TST     R4              ;;SUPPRESS THIS 0?
2676  015274  001403                          BEQ     5$              ;;BR IF YES
2677  015276  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2678  015300  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2679  015304  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2680  015310  110337  015354                  MOVB    R3,8$           ;;SAVE FOR TYPING
2681  015314  104400  015354                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
2682  015320  105337  015356          7$:     DECB    $OCNT           ;;COUNT BY 1
2683  015324  003347                          BGT     2$              ;;BR IF MORE TO DO
2684  015326  002402                          BLT     6$              ;;BR IF DONE
2685  015330  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2686  015332  000744                          BR      2$              ;;GO DO THE LAST DIGIT
2687  015334  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
2688  015336  012604                          MOV     (SP)+,R4        ;;RESTORE R4
2689  015340  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2690  015342  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2691  015350  012616                          MOV     (SP)+,(SP)
2692  015352  000002                          RTI                     ;;RETURN
2693  015354     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2694  015355     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2695  015356     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2696  015357     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2697  015360  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2698                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
2699
2700                                  ;;**************************************************************
2701                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
2702                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
2703                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
2704                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
2705                                  ;*REPLACED WITH SPACES.
```

```
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                        ;*      TYPDS                   ;;GO TO THE ROUTINE
                                        $TYPDS:
015362  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
015364  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
015366  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
015370  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
015372  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
015374  012746  020200                         MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
015400  016605  000020                         MOV     20(SP),R5       ;;GET THE INPUT NUMBER
015404  100004                                  BPL     1$              ;;BR IF INPUT IS POS.
015406  005405                                  NEG     R5              ;;MAKE THE BINARY NUMBER POS.
015410  112766  000055  000001                 MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
015416  005000                          1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
015420  012703  015576                         MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
015424  112723  000040                         MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
015430  005002                          2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
015432  016001  015566                         MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
015436  160105                          3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
015440  002402                                  BLT     4$              ;;BR IF DONE
015442  005202                                  INC     R2              ;;INCREASE THE BCD DIGIT BY 1
015444  000774                                  BR      3$
015446  060105                          4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
015450  005702                                  TST     R2              ;;CHECK IF BCD DIGIT=0
015452  001002                                  BNE     5$              ;;FALL THROUGH IF 0
015454  105716                                  TSTB    (SP)            ;;STILL DOING LEADING 0'S?
015456  100407                                  BMI     7$              ;;BR IF YES
015460  106316                          5$:     ASLB    (SP)            ;;MSD?
015462  103003                                  BCC     6$              ;;BR IF NO
015464  116663  000001  177777                 MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
015472  052702  000060                  6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
015476  052702  000040                  7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
015502  110223                                  MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
015504  005720                                  TST     (R0)+           ;;JUST INCREMENTING
015506  020027  000010                         CMP     R0,#10          ;;CHECK THE TABLE INDEX
015512  002746                                  BLT     2$              ;;GO DO THE NEXT DIGIT
015514  003002                                  BGT     8$              ;;GO TO EXIT
015516  010502                                  MOV     R5,R2           ;;GET THE LSD
015520  000764                                  BR      6$              ;;GO CHANGE TO ASCII
015522  105726                          8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
015524  100003                                  BPL     9$              ;;BR IF NO
015526  116663  177777  177776                 MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
015534  105013                          9$:     CLRB    (R3)            ;;SET THE TERMINATOR
015536  012605                                  MOV     (SP)+,R5        ;;POP STACK INTO R5
015540  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
015542  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
015544  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
015546  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
015550  104400  015576                         TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
015554  016666  000002  000004                 MOV     2(SP),4(SP)     ;;ADJUST THE STACK
015562  012616                                  MOV     (SP)+,(SP)
015564  000002                                  RTI                     ;;RETURN TO USER
015566  023420                          $DTBL:  10000.
015570  001750                                  1000.
```

```
                                        100.
                                        10.
                        $DBLK:  .BLKW   4
                        .SBTTL  ERROR HANDLER ROUTINE

                        ;;**********************************************************************
                        ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                        ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                        ;*AND GO TO SWRCK ON ERROR
                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                        ;*SW15=1         HALT ON ERROR
                        ;*SW13=1         INHIBIT ERROR TYPEOUTS
                        ;*SW10=1         BELL ON ERROR
                        ;*SW09=1         LOOP ON ERROR
                        ;*CALL
                        ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

015606                  $ERROR:
015606  104406                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
015610  104406                  CKSWR
015612  105237  001103  7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
015616  001775                  BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
015620  013777  001102  163314  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
015626  032777  002000  163304  BIT     #BIT10,@SWR     ;;BELL ON ERROR?
015634  001402                  BEQ     1$              ;;NO - SKIP
015636  104400  001164          TYPE    ,$BELL          ;;RING BELL
015642  005237  001112  1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
015646  011637  001116          MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
015652  162737  000002  001116  SUB     #2,$ERRPC
015660  117737  163232  001114  MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
015666  032777  020000  163244  BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
015674  001004                  BNE     20$             ;;SKIP TYPEOUTS
015676  004737  015776          JSR     PC,SWRCK        ;;GO TO USER ERROR ROUTINE
015702  104400  001171          TYPE    ,$CRLF
015706                  20$:
015706  122737  000001  001214  CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
015714  001007                  BNE     2$              ;;NO,SKIP APT ERROR REPORT
015716  113737  001114  015730  MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
015724  004737  014704          JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
015730     000          21$:    .BYTE   0
015731     000                  .BYTE   0
015732  000777          22$:    BR      22$             ;;APT ERROR LOOP
015734  005777  163200  2$:     TST     @SWR            ;;HALT ON ERROR
015740  100002                  BPL     3$              ;;SKIP IF CONTINUE
015742  000000                  HALT                    ;;HALT ON ERROR!
015744  104406                  CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
015746  032777  001000  163164  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
015754  001402                  BEQ     4$              ;;BR IF NO
015756  013716  001110          MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
015762  005737  001162  4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
015766  001402                  BEQ     5$              ;;BR IF NONE
015770  013716  001162          MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
015774                  5$:
015774  000002                  RTI                     ;;RETURN
                        ;;**********************************************************************
                        ;GO TYPE ERROR
```

```
                                        ;GO UPDATE SOFTWARE SWR IF 'CNTRL/G'
                                        ;;*****************************************************************
016776  113737  001102  001600  $WRCK:  MOVB    $TSTNM,TSTNUM   ;SET UP TEST # ON ER
016004  004737  016014                  JSR     PC,$ERRTYP      ;GO TYPE ERROR
016010  104406                          CKSWR                   ;LOOK AT KEYBOARD
016012  000207                          RTS     PC              ;RETURN TO ERROR HANDLER
                                        .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE


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

016014                                  $ERRTYP:
016014  104400  001171                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
016020  010046                                  MOV     R0,-(SP)        ;;SAVE R0
016022  005000                                  CLR     R0              ;;PICKUP THE ITEM INDEX
016024  153700  001114                          BISB    @#$ITEMB,R0
016030  001004                                  BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                                        ;;TYPE THE PC OF THE ERROR
016032  013746  001116                          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;ERROR ADDRESS
016036  104401                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
016040  000426                                  BR      6$              ;;GET OUT
016042  005300                          1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
016044  006300                                  ASL     R0              ;;      WORK FOR THE ERROR TABLE
016046  006300                                  ASL     R0
016050  006300                                  ASL     R0
016052  062700  001256                          ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
016056  012037  016066                          MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
016062  001404                                  BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
016064  104400                                  TYPE                    ;;TYPE THE "ERROR MESSAGE"
016066  000000                          2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
016070  104400  001171                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
016074  012037  016104                  3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
016100  001404                                  BEQ     5$              ;;SKIP TYPEOUT IF 0
016102  104400                                  TYPE                    ;;TYPE THE "DATA HEADER"
016104  000000                          4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
016106  104400  001171                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
016112  011000                          5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
016114  001004                                  BNE     7$              ;;GO TYPE THE DATA
016116  012600                          6$:     MOV     (SP)+,R0        ;;RESTORE R0
016120  104400  001171                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
016124  000207                                  RTS     PC              ;;RETURN
016126                                  7$:
016126  013046                                  MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
016130  104401                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
016132  005710                                  TST     (R0)            ;;IS THERE ANOTHER NUMBER?
016134  001770                                  BEQ     6$              ;;BR IF NO
016136  104400  016144                          TYPE    ,8$             ;;TYPE TWO(2) SPACES
016142  000771                                  BR      7$              ;;LOOP
016144  020040      000           8$:     .ASCIZ  /  /                ;;TWO(2) SPACES
016150                                  .EVEN
                                        .SBTTL  SCOPE HANDLER ROUTINE


                                        ;;*****************************************************************
```

```
                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                        ;*ANB LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW14=1         LOOP ON TEST
                                        ;*SW11=1         INHIBIT ITERATIONS
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT

016150                                  $SCOPE:
016150  104406                                  CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
016152  104406                                  CKSWR
016154  032777  040000  162756  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
016162  001114                          BNE     $OVER           ;;YES IF SW14=1
                                        ;*****START OF CODE FOR THE XOR TESTER*****
016164  000416                          $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
016166  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
016172  012737  016212  000004                  MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
016200  005737  177060                          TST     @#177060        ;;TIME OUT ON XOR?
016204  012637  000004                          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
016210  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
016212  022626                          5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
016214  012637  000004                          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
016220  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
016222                                  6$:;*****END OF CODE FOR THE XOR TESTER*****
016222  032777  000400  162710                  BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
016230  001404                          BEQ     2$              ;;BR IF NO
016232  127737  162702  001102                  CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
016240  001465                          BEQ     $OVER           ;;BR IF YES
016242  105737  001103                  2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
016246  001421                          BEQ     3$              ;;BR IF NO
016250  123737  001115  001103                  CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
016256  101015                          BHI     3$              ;;BR IF NO
016260  032777  001000  162652                  BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
016266  001404                          BEQ     4$              ;;BR IF NO
016270  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
016276  000446                          BR      $OVER
016300  105037  001103                  4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
016304  005037  001160                          CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
016310  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
016312  032777  004000  162620  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
016320  001011                          BNE     1$              ;;BR IF YES
016322  005737  001202                          TST     $PASS           ;;IF FIRST PASS OF PROGRAM
016326  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
016330  005237  001104                          INC     $ICNT           ;;INCREMENT ITERATION COUNT
016334  023737  001160  001104                  CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
016342  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
016344  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
016352  013737  016430  001160                  MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
016360  105237  001102                  $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
016364  113737  001102  001200                  MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
016372  011637  001106                          MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
016376  011637  001110                          MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
```

```
2930  016402  005037  001162                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
2931  016406  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
2932  016414  013777  001102  162520  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
2933  016422  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
2934  016426  000002                          RTI                     ;;FIXES PS
2935  016430  003720                  $MXCNT: 2000.                   ;;MAX. NUMBER OF ITERATIONS
2936                                   .SBTTL  TTY INPUT ROUTINE
2937
2938                                   ;;**************************************************************************
2939                                   .ENABL  LSB
2940
2941                                   ;;**************************************************************************
2942                                   ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
2943                                   ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
2944                                   ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
2945                                   ;*WHEN OPERATING IN TTY FLAG MODE.
2946  016432  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
2947  016440  001074                          BNE     15$             ;;BRANCH IF NO
2948  016442  105777  162476                  TSTB    @$TKS           ;;CHAR THERE?
2949  016446  100071                          BPL     15$             ;;IF NO, DON'T WAIT AROUND
2950  016450  117746  162472                  MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
2951  016454  042716  177600                  BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
2952  016460  022726  000007                  CMP     #7,(SP)+        ;;IS IT A CONTROL G?
2953  016464  001062                          BNE     15$             ;;NO, RETURN TO USER
2954  016466  123727  001134  000001          CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
2955  016474  001456                          BEQ     15$             ;;BRANCH IF YES
2956
2957  016476  104400  017157                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
2958  016502  104400  017164  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
2959  016506  013746  000176                  MOV     $SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
2960  016512  104401                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2961  016514  104400  017175                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
2962  016520  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
2963  016522  005046                          CLR     -(SP)           ;;THE NEW SWR
2964  016524  105777  162414          7$:     TSTB    @$TKS           ;;CHAR THERE?
2965  016530  100375                          BPL     7$              ;;IF NOT TRY AGAIN
2966
2967  016532  117746  162410                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
2968  016536  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
2969
2970
2971
2972  016542  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
2973  016546  001005                          BNE     10$             ;;BRANCH IF NOT
2974  016550  104400  017152                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
2975  016554  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
2976  016560  000757                          BR      19$             ;;LET'S TRY IT AGAIN
2977
2978
2979  016562  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
2980  016566  001022                          BNE     16$             ;;BRANCH IF NO
2981  016570  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
2982  016574  001403                          BEQ     11$             ;;BRANCH IF YES
2983  016576  016677  000002  162334          MOV     2(SP),@SWR      ;;SAVE NEW SWR
2984  016604  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
2985  016610  104400  001171          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
```

```
2986  016614  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
2987  016622  001003                          BNE     15$             ;;BRANCH IF NOT
2988  016624  012777  000100  162312          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
2989  016632  000002                  15$:    RTI                     ;;RETURN
2990  016634  004737  014616          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
2991  016640  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
2992  016644  002420                          BLT     18$             ;;BRANCH IF YES
2993  016646  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
2994  016652  003015                          BGT     18$             ;;BRANCH IF YES
2995  016654  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
2996  016660  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
2997  016664  001403                          BEQ     17$             ;;BRANCH IF YES
2998  016666  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
2999  016670  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
3000  016672  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
3001  016674  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
3002  016700  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
3003  016704  000707                          BR      7$              ;;GET THE NEXT ONE
3004  016706  104400  001170          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
3005  016712  000720                          BR      20$             ;;SIMULATE CONTROL-U
3006                                .DSABL  LSB
3007
3008
3009                                ;;*****************************************************************
3010                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
3011                                ;*CALL:
3012                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
3013                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
3014                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
3015                                ;
3016
3017  016714  011646                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
3018  016716  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
3019  016724  105777  162214          1$:     TSTB    @$TKS           ;;WAIT FOR
3020  016730  100375                          BPL     1$              ;;A CHARACTER
3021  016732  117766  162210  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
3022  016740  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
3023  016746  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
3024  016754  001013                          BNE     3$              ;;BRANCH IF NO
3025  016756  105777  162162          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
3026  016762  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
3027  016764  117746  162156                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
3028  016770  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
3029  016774  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
3030  017000  001366                          BNE     2$              ;;IF NOT DISCARD IT
3031  017002  000750                          BR      1$              ;;YES, RESUME
3032  017004  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
3033  017012  002407                          BLT     4$              ;;BRANCH IF YES
3034  017014  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
3035  017022  003003                          BGT     4$              ;;BRANCH IF YES
3036  017024  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
3037  017032  000002                  4$:     RTI                     ;;GO BACK TO USER
3038                                ;;*****************************************************************
3039                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
3040                                ;*CALL:
3041                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
```

```
                                        ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                        ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
                                $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
017034  010346                  1$:     MOV     @#$TTYIN,R3             ;;GET ADDRESS
017036  012703  017142          2$:     CMP     @#$TTYIN+8.,R3          ;;BUFFER FULL?
017042  022703  017152                  BLOS    4$                      ;;BR IF YES
017046  101405                          RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
017050  104407                          MOVB    (SP)+,(R3)              ;;GET CHARACTER
017052  112613                  10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
017054  122713  000177                  BNE     3$                      ;;SKIP IF NOT
017060  001003                  4$:     TYPE    ,$QUES                  ;;TYPE A '?'
017062  104400  001170                  BR      1$                      ;;CLEAR THE BUFFER AND LOOP
017066  000763                  3$:     MOVB    (R3),9$                 ;;ECHO THE CHARACTER
017070  111337  017140                  TYPE    ,9$
017074  104400  017140                  CMPB    #15,(R3)+               ;;CHECK FOR RETURN
017100  122723  000015                  BNE     2$                      ;;LOOP IF NOT RETURN
017104  001356                          CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
017106  105063  177777                  TYPE    ,$LF                    ;;TYPE A LINE FEED
017112  104400  001172                  MOV     (SP)+,R3                ;;RESTORE R3
017116  012603                          MOV     (SP),-(SP)              ;;ADJUST THE STACK AND PUT ADDRESS OF THE
017120  011646                          MOV     4(SP),2(SP)             ;;      FIRST ASCII CHARACTER ON IT
017122  016666  000004  000002          MOV     @#$TTYIN,4(SP)
017130  012766  017142  000004          RTI                             ;;RETURN
017136  000002                  9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
017140     000                          .BYTE   0                       ;;TERMINATOR
017141     000                  $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
017142  000010                  $CNTLU: .ASCIZ  /^U/<15><12>            ;;CONTROL "U"
017152  052536  005015     000  $CNTLG: .ASCIZ  /^G/<15><12>            ;;CONTROL "G"
017157     136  006507  000012  $MSWR:  .ASCIZ  <15><12>/SWR = /
017164  005015  053523  020122  $MNEW:  .ASCIZ  /  NEW = /
017172  020075     000
017175     040  047040  053505
017202  036440  000040

                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;**************************************************************
                                        ;POWER DOWN ROUTINE
017206  012737  017352  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC        ;;SET FOR FAST UP
017214  012737  000340  000026          MOV     #340,@#PWRVEC+2         ;;PRIO:7
017222  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
017224  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
017226  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
017230  010346                          MOV     R3,-(SP)                ;;PUSH R3 ON STACK
017232  010446                          MOV     R4,-(SP)                ;;PUSH R4 ON STACK
017234  010546                          MOV     R5,-(SP)                ;;PUSH R5 ON STACK
017236  017746  161676                  MOV     @$WR,-(SP)              ;;PUSH @$WR ON STACK
017242  010637  017356                  MOV     SP,$SAVR6               ;;SAVE SP
017246  012737  017260  000024          MOV     #$PWRUP,@#PWRVEC        ;;SET UP VECTOR
017254  000000                          HALT
017256  000776                          BR      .-2                     ;;HANG UP

                                        ;;**************************************************************
                                        ;POWER UP ROUTINE
017260  012737  017352  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC        ;;SET FOR FAST DOWN
017266  013706  017356                  MOV     $SAVR6,SP               ;;GET SP
017272  005037  017356                  CLR     $SAVR6                  ;;WAIT LOOP FOR THE TTY
```

```
017276  005237  017356          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
017302  001375                          BNE     1$              ;;OF  WORD
017304  012677  161630                  MOV     (SP)+,@#SWR     ;;POP STACK INTO @#SWR
017310  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
017312  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
017314  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
017316  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
017320  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
017322  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
017324  012737  017206  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
017332  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
017340  104400                          TYPE                    ;;REPORT THE POWER FAILURE
017342  017360                  $PWRMG: .WORD   PWRMSG          ;;POWER FAIL MESSAGE POINTER
017344  012716                          MOV     (PC)+,(SP)      ;;RESTART AT RESTRT
017346  002310                  $PWRAD: .WORD   RESTRT          ;;RESTART ADDRESS
017350  000002                          RTI
017352  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
017354  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
017356  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
017360  005015  042522  052123  PWRMSG: .ASCIZ  <15><12>/RESTARTED FROM PWR FAIL/
017366  051101  042524  020104
017374  051106  046517  050040
017402  051127  043040  044501
017410  000114

                                .EVEN
                                .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.

017412  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
017414  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
017420  005740                          TST     -(R0)           ;;BACKUP BY 2
017422  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
017424  006300                          ASL     R0              ;;POSITION FOR INDEXING
017426  016000  017434                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
017432  000200                          RTS     R0              ;;GO TO ROUTINE

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
017434                          $TRPAD:
017434  014404                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
017436  015160                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
017440  015134                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
017442  015174                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
017444  015362                          $TYPDS  ;;CALL=TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)

017446  016502                          $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
```

```
3154
3155  017450  016432                          $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
3156  017452  016714                          $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
3157  017454  017034                          $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
3158
3159          070000                          .=70000                 ;PUT MESSAGES ON TOP OF
3160                                                                  ;MEM WORKING AREA
3161
3162                                  .SBTTL  ASCII MESSAGES
3163  070000  005015     000         MSG2:   .ASCIZ  <15><12>
3164  070003     040  040502  020122 MSG3:   .ASCIZ  / BAR DN/
3165  070010  047104     000
3166  070013     040  047516  041440 MSG4:   .ASCIZ  / NO CNVRT/
3167  070020  053116  052122     000
3168  070025     040  036530     000 MSG5:   .ASCIZ  / X=/
3169  070031     040  036531     000 MSG6:   .ASCIZ  / Y=/
3170
3171  070035     102  051525  052040 EM1:    .ASCIZ  /BUS TIMEOUT ER/
3172  070042  046511  047505  052125
3173  070050  042440  000122
3174  070054  042522  020107  042522 EM2:    .ASCIZ  'REG READ/WRITE ER'
3175  070062  042101  053457  044522
3176  070070  042524  042440  000122
3177  070076  052502  020123  042522 EM3:    .ASCIZ  /BUS RESET CLR ER/
3178  070104  042523  020124  046103
3179  070112  020122  051105     000
3180  070117     123  020106  046103 EM4:    .ASCIZ  /SF CLR Z ER/
3181  070124  020122  020132  051105
3182  070132     000
3183  070133     123  020106  046103 EM5:    .ASCIZ  /SF CLR ALL ER/
3184  070140  020122  046101  020114
3185  070146  051105     000
3186  070151     132  044440  041516 EM6:    .ASCIZ  /Z INC ER ON ADC START/
3187  070156  042440  020122  047117
3188  070164  040440  041504  051440
3189  070172  040524  052122     000
3190  070177     132  047440  043126 EM7:    .ASCIZ  /Z OVFLO FLAG ER/
3191  070204  047514  043040  040514
3192  070212  020107  051105     000
3193  070217     132  047440  043126 EM10:   .ASCIZ  /Z OVFLO INTR ER/
3194  070224  047514  044440  052116
3195  070232  020122  051105     000
3196  070237     111  052116  020122 EM11:   .ASCIZ  /INTR AT WRONG BR LEVEL/
3197  070244  052101  053440  047522
3198  070252  043516  041040  020122
3199  070260  042514  042526  000114
3200  070266  044514  052123  046440 EM12:   .ASCIZ  /LIST MD NPR DATA ER/
3201  070274  020104  050116  020122
3202  070302  040504  040524  042440
3203  070310  000122
3204  070312  044514  052123  046440 EM13:   .ASCIZ  /LIST MD CA INC ER/
3205  070320  020104  040503  044440
3206  070326  041516  042440  000122
3207  070334  044514  052123  046440 EM14:   .ASCIZ  /LIST MD WC INC ER/
3208  070342  020104  041527  044440
3209  070350  041516  042440  000122
```

```
3210  070356  041527  020117  046106  EM15:   .ASCIZ  /WCO FLAG OR INTR ER/
3211  070364  043501  047440  020122
3212  070372  047111  051124  042440
3213  070400  000122
3214  070402  044524  042515  052517  EM16:   .ASCIZ  /TIMEOUT FLAG ER/
3215  070410  020124  046106  043501
3216  070416  042440  000122
3217  070422  042101  051522  046440  EM17:   .ASCIZ  /ADRS MAKE ER/
3218  070430  045501  020105  051105
3219  070436     000
3220  070437     112  054517  052123  EM20:   .ASCIZ  /JOYSTICK MD FAILED TO OVERRIDE LIST MD/
3221  070444  041511  020113  042115
3222  070452  043040  044501  042514
3223  070460  020104  047524  047440
3224  070466  042526  051122  042111
3225  070474  020105  044514  052123
3226  070502  046440  000104
3227  070506  050116  020122  047111  EM21:   .ASCIZ  /NPR INC ER/
3228  070514  020103  051105     000
3229  070521     111  041516  047411  EM22:   .ASCIZ  /INC    OVFLO FLAG OR INTR ER/
3230  070526  043126  047514  043040
3231  070534  040514  020107  051117
3232  070542  044440  052116  020122
3233  070550  051105     000
3234  070553     127  051117  020104  EM23:   .ASCIZ  /WORD INC ER/
3235  070560  047111  020103  051105
3236  070566     000
3237  070567     102  052131  020105  EM24:   .ASCIZ  /BYTE INC ER/
3238  070574  047111  020103  051105
3239  070602     000
3240  070603     101  041504  047440  EM25:   .ASCIZ  /ADC OR JOYSTICK LOGIC ER/
3241  070610  020122  047512  051531
3242  070616  044524  045503  046040
3243  070624  043517  041511  042440
3244  070632  000122
3245  070634  044524  044515  043516  EM26:   .ASCIZ  /TIMING MARK FAILED ON ADRS MAKE/
3246  070642  046440  051101  020113
3247  070650  040506  046111  042105
3248  070656  047440  020116  042101
3249  070664  051522  046440  045501
3250  070672  000105
3251  070674  047516  041440  053116  EM27:   .ASCIZ  /NO CNVRT FAILED/
3252  070702  052122  043040  044501
3253  070710  042514  000104
3254  070714  051105  050122  020103  DH1:    .ASCIZ  /ERRPC   TSTNUM  BUSADR  EXPCT   RCVD/
3255  070722  020040  051524  047124
3256  070730  046525  020040  052502
3257  070736  040523  051104  020040
3258  070744  054105  041520  020124
3259  070752  020040  041522  042126
3260  070760     000
3261  070761     105  051122  041520  DH2:    .ASCIZ  /ERRPC   TSTNUM  XYHOLD  GDADR   BDADR/
3262  070766  020040  052040  052123
3263  070774  052516  020115  054040
3264  071002  044131  046117  020104
3265  071010  043440  040504  051104
```

```
3266  071016  020040  041040  040504
3267  071024  051104     000
3268  071027     105  051122  041520  DH3:    .ASCIZ  /ERRPC   TSTNUM  XYHOLD  ADRS    GDDAT   BDDAT/
3269  071034  020040  052040  052123
3270  071042  052516  020115  054040
3271  071050  044131  046117  020104
3272  071056  040440  051104  020123
3273  071064  020040  043440  042104
3274  071072  052101  020040  041040
3275  071100  042104  052101     000
3276  071105     105  051122  041520  DH4:    .ASCIZ  /ERRPC   TSTNUM/
3277  071112  020040  052040  052123
3278  071120  052516  000115
3279
3280                                          .EVEN
3281  071124  001116  001600  001122  DT1:    $ERRPC,TSTNUM,$BDADR,$GDDAT,$BDDAT,0
3282  071132  001124  001126  000000
3283  071140  001116  001600  013226  DT2:    $ERRPC,TSTNUM,XYHOLD,$GDADR,$BDADR,0
3284  071146  001120  001122  000000
3285  071154  001116  001600  013226  DT3:    $ERRPC,TSTNUM,XYHOLD,$BDADR,$GDDAT,$BDDAT,0
3286  071162  001122  001124  001126
3287  071170  000000
3288  071172  001116  001600  000000  DT4:    $ERRPC,TSTNUM,0
3289          000001                          .END
```

```
ABASE = 164000            491*    588     629
ACDW1 = 000000            589     631
ACDW2 = 000000            589
ACPJOP= 000000            588     603
ADDW0 = 000000            589
ADDW1 = 000000            589
ADDW10= 000000            588
ADDW11= 000000            589
ADDW12= 000000            589
ADDW13= 000000            588
ADDW14= 000000            588
ADDW15= 000000            588
ADDW2 = 000000            588
ADDW3 = 000000            588
ADDW4 = 000000            588
ADDW5 = 000000            588
ADDW6 = 000000            588
ADDW7 = 000000            588
ADDW8 = 000000            588
ADDW9 = 000000            588
ADEVCT= 000000            588     594
ADEVM = 000001            494*    588     630
AENV  = 000000            588     599
AENVM = 000000            588     600
AFATAL= 000000            589     591
AMADR1= 000000            588     616
AMADR2= 000000            588     620
AMADR3= 000000            588     623
AMADR4= 000000            588     626
AMAMS1= 000000            588     610
AMAMS2= 000000            588     618
AMAMS3= 000000            588     621
AMAMS4= 000000            588     624
AMSGAD= 000000            588     596
AMSGLG= 000000            588     597
AMSGTY= 000000            588     590
AMTYP1= 000000            588     611
AMTYP2= 000000            589     619
AMTYP3= 000000            589     622
AMTYP4= 000000            588     627
APASS = 000000            588     5
APRIOR= 000340            493*    5
APTCSU= 000040           2515     2
APTENV= 000001           2508     2       2618*   2797
APTSIZ= 000200            857     2
APTSPO= 000100           2510     2       2619*
ASWREG= 000000            588     6 1
ATESTN= 000000            588     592
AUNIT = 000000            588     595
AUSWR = 000000            588     602
AVECT1= 160270            492*    588      627
AVECT2= 000000            588     628
BIT0  = 000001            476*
BIT00 = 000001            466*    476
BIT01 = 000002            465*    475
BIT02 = 000004            464*    474
```

DZNCA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
BIT03 = 000010           463*   473
BIT04 = 000020           462*   472
BIT05 = 000040           461*   471
BIT06 = 000100           460*   470
BIT07 = 000200           459*   469
BIT08 = 000400           458*   468   2902
BIT09 = 001000           457*   467   2808   2910
BIT1  = 000002           475*
BIT10 = 002000           456*  2785
BIT11 = 004000           455*  1198   1225   1251   1277   2917
BIT12 = 010000           454*  1158   1159   1166   1168   1182   1184   1206   1226   1358   1361   1364   1366
                        1375   1378   1393   1417   1430
BIT13 = 020000           453*  2104   2124   2129   2138   2350   2353   2355   2792
BIT14 = 040000           452*  2352   2888
BIT15 = 100000           451*  2048
BIT2  = 000004           474*
BIT3  = 000010           473*
BIT4  = 000020           472*
BIT5  = 000040           471*
BIT6  = 000100           470*
BIT7  = 000200           469*
BIT8  = 000400           468*
BIT9  = 001000           467*
BPTVEC= 000014           483*
CKSWR = 104406          2107   2780   2781   2807   2822   2886   2887   3155*
CR    = 000015           391*  2554   2564
CRLF  = 000200           392*   897   2525   2564
DDISP = 177570           398*   569    845
DELAY   012702          2068   2081   2109   2194*
DH1     070714           651    657    663    669    675    681    687    693    699    711    717    723    729
                         753    777    783   3254*
DH2     070761           735    741   3261*
DH3     071027           705    747    759    765   3268*
DH4     071105           771   3276*
DISPLA  001142           569*   845*   853*  2784*  2932*
DISPRE  000174           502*   853
DNCS    014104          2424   2426*
DSWR  = 177570           397*   568    844
DT1     071124           652    658    664    670    676    682    688    694    700    712    718    724    730
                         754    778    784   3281*
DT2     071140           736    742   3283*
DT3     071154           706    748    760    766   3285*
DT4     071172           772   3288*
DUAL    013212          1445*  1723*  1745*  1767*  1789*  2236   2264*  2348   2378   2448   2474
EMTVEC= 000030           486*   829*   830*
EM1     070035           650   3171*
EM10    070217           692   3193*
EM11    070237           698   3196*
EM12    070266           704   3200*
EM13    070312           710   3204*
EM14    070334           716   3207*
EM15    070356           722   3210*
EM16    070402           728   3214*
EM17    070422           734   3217*
EM2     070054           656   3174*
EM20    070437           740   3220*
```

DZNCA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
EM21    070506          746    3227#
EM22    070521          752    3229#
EM23    070553          758    3234#
EM24    070567          764    3237#
EM25    070603          770    3240#
EM26    070634          776    3245#
EM27    070674          782    3251#
EM3     070076          662    3177#
EM4     070117          668    3180#
EM5     070133          674    3183#
EM6     070151          680    3186#
EM7     070177          686    3190#
ERRVEC= 000004          479#    842     843*    854*    916*    927*   2893    2894*   2896*   2899*
GNS     = ****** U      501     896    3147    3148    3149    3150    3151    3153    3155    3156    3157
GOOD    013230         1938*   1941*   1943*   1983    2245    2248*   2249    2250*   2261    2262*   2271#   2285*   2289*
                       2313*   2314*   2315*   2322*   2329*   2339    2340*   2342*   2343*   2344*   2345*   2356*   2359*
                       2368    2369*   2371*   2372*   2373*   2374*   2375*   2381*   2383*   2384*   2394    2395*   2397*
                       2398*   2399*   2402*   2404*   2414    2415*   2417*   2418*   2419*   2420*   2421    2423*   2425*
                       2426*   2429*   2439    2440*   2442*   2443*   2444*   2445*   2451*   2453*   2454*   2464    2465*
                       2467*   2468*   2469*   2470*   2471*   2477*   2480*
GTSWR = 104405          891    3153#
HT    = 000011          389#   2523    2564
IOTVEC= 000020          484#    827*    828*
JSM     013362         2287    2313#
LF    = 000012          390#   2558    2564
LOCSAV  013240         2249*   2262    2275#
MAKE    012746         1450    1466    1484    1503    1523    1557    1591    1625    1659    1693    1728    1750    1772
                       1794    2226#
MKLOOP  013050         1451    1467    1485    1504    1524    1559    1592    1626    1660    1694    1729    1751    1773
                       1795    2241#
MSG2    070000         2106    3163#
MSG3    070003         2126    3164#
MSG4    070013         2131    3166#
MSG5    070025         2140    3168#
MSG6    070031         2146    3169#
M128X8  014004         2393    2396    2404#   2413    2416
M32X16  013702         2367    2370    2383#
M32X8   014372         2463    2466    2479#
M64X16  013560         2338    2341    2358#
M64X8   014244         2438    2441    2453#   2479
NCADR   001552          791#   2060    2069    2082    2258
NCBRL   001574          807#    861*    862*    863*    864
NCBRL1  001576          808#    864*    865*   1200    1231    1254    1280    1411
NCCSR   001546          789#    866     917     934     939*    940     954     956*    958    1157    1159    1166    1179
                       1182    1194    1198*   1201    1206    1219    1225*   1235    1245    1251*   1261    1271    1277*
                       1286    1301*   1329*   1355    1358    1364    1375    1389*   1391    1405*   1410    1418    1426
                       1431    1813*   1829    1831    1841    1843    1850    1852    1875*   1891    1893    1902    1904
                       1911    1913    1936*   1952    1954    1964    1966    1973    1975    2000*   2006    2010    2018
                       2023    2036*   2038    2040    2045    2062*   2066*   2100*   2110    2112    2114    2122    2205*
NCIREG  001560          795#   1048    1051*   1052    1064    1066*   1068    1093    1095*   1097
NCOFF   001550          790#    967     972*    973     985     987*    989    1812*   1874*   1935*   1999*   2035*   2203*
NCSFF   001562          796#    870     953*   1079*   1096*   1108*   1109*   1131*   1133*   1153*   1156*   1165*   1176*
                       1178*   1192*   1199*   1222*   1227*   1248*   1253*   1274*   1279*   1300*   1326*   1330*   1386*
                       1392*   1393    1402*   1414*   1446*   1462*   1479*   1498*   1519*   1553*   1587*   1621*   1655*
                       1689*   1724*   1746*   1768*   1790*   1811*   1824*   1839*   1849*   1873*   1886*   1900*   1910*
                       1934*   1947*   1962*   1972*   1998*   2003*   2034*   2044*   2061*   2064*   2065*   2067*   2080*
```

```
                    2099*  2108*  2121*  2243*
NCVCT0  001564       800#   872   1995*  2027*
NCVCT2  001566       801#  1996*  2027   2028*
NCVCT4  001570       802#  1193*  1210*  1218*  1237*  1249*  1263*  1275*  1288*  1403*  1436*
NCVCT6  001572       803#   877   1195*  1210   1211*  1220*  1237   1238*  1250*  1263   1264*  1276*  1288
                    1289*  1404*  1436   1437*
NCXYH   001552       792#  1306*  1332*  1390*  1409*  1825*  1840*  1887*  1901*  1948*  1963*  2004*  2037*
                    2063*  2120   2256*
NCZHWC  001554       793#   998   1001*  1002   1032*  1038   1040   1077*  1080   1082   1116   1118   1134
                    1136   1155*  1197*  1224*  1278*  1302*  1328*  1350   1353   1388*  1406*
NCZLCA  001556       794#  1014   1017*  1018   1031*  1034   1036   1078*  1084   1086   1110   1112   1132*
                    1140   1142   1154*  1177*  1196*  1223*  1252*  1303*  1327*  1343   1346   1387*  1407*
ND2     013550      2349   2354   2356#
ND3     013672      2379   2381#
ND6     014234      2449   2451#
ND7     014362      2475   2477#
NODUAL  013042      2237   2240#
OFFSET  013234      1447*  1463*  1482*  1490*  1507*  1509*  1520*  1537*  1554*  1571*  1588*  1605*  1622*
                    1639*  1656*  1673*  1690*  1707*  1725*  1735*  1747*  1757*  1769*  1779*  1791*  1801*
                    2203   2273#  2346   2376   2400   2427   2446   2472
PC      =000007      410#  1449*  1450*  1451*  1465*  1466*  1467*  1483*  1484*  1485*  1502*  1503*  1504*
                    1522*  1523*  1524*  1556*  1557*  1558*  1590*  1591*  1592*  1624*  1625*  1626*  1658*
                    1659*  1660*  1692*  1693*  1694*  1727*  1728*  1729*  1749*  1750*  1751*  1771*  1772*
                    1773*  1793*  1794*  1795*  2068*  2081*  2109*  2167*  2170*  2180*  2185   2197*  2207*
                    2244*  2263*  2316*  2323*  2330*  2338*  2341*  2357*  2359*  2367*  2370*  2382*  2385*
                    2393*  2396*  2403*  2405*  2413*  2416*  2430*  2438*  2441*  2452*  2455*  2463*  2466*
                    2478*  2479*  2481*  2513*  2532*  2539*  2546*  2560*  2562*  2595*  2612*  2794*  2800*
                    2821*  2823*  2861*  2990*  3111
PIRQ    = 177772     396#
PIRQVE= 000240       490#
PR0     = 000000     413#
PR1     = 000040     414#
PR2     = 000100     415#
PR3     = 000140     416#
PR4     = 000200     417#
PR5     = 000240     418#
PR6     = 000300     419#  2005
PR7     = 000340     420#   898   1191   1195   1217   1220   1221   1246   1250   1272   1276   1296   1322
                    1385   1401   1404   1994
PS      = 177776     393#   394
PSW     = 177776     394#   898*  1191*  1200*  1217*  1228*  1246*  1254*  1272*  1280*  1296*  1322*  1385*
                    1401*  1411*  1994*  2005*
PWRMSG  017360      3110   3117#
PWRVEC= 000024       485#   833#   834*  3079*  3080*  3089*  3095*  3107*  3108*
RDCHR   = 104407    3049   3156#
RDLIN   = 104410    3157#
REGLD   012714      1449   1465   1483   1502   1522   1556   1590   1624   1658   1692   1727   1749   1771
                    1793   2203#
RESTRT  002310       898#  2186   3112
RESVEC= 000010       480#
RES00   013406      2291   2322#
RES01   013414      2293   2329#
RES02   013424      2295   2338#
RES03   013570      2297   2367#
RES04   013716      2305   2393#
RES04X  013346      2299   2305#
```

```
RES05   014014          2306    2413#
RES05X  013352          2301    2306#
RES06   014136          2307    2438#
RES06X  013356          2303    2307#
RES07   014260          2304    2463#
RNGDN   013210          2231    2260    2263#
R0      =%000000         401#    866*    868*    870     872*    875*    877     917*    919     920     921     936*    937
                         939     945*    969*    970     972     977*    1107*   1111    1121*   1130*   1135    1145*   1221*
                        1228    1230*   1231    1247*   1256*   1273*   1282*   1298*   1299    1303    1304*   1308    1323*
                        1327    1331*   1335    1336    1341    1372*   1408*   1415*   1423*   1810*   1822*   1872*   1884*
                        1897    1933*   1945*   1959    1997*   2007*   2015    2059*   2091*   2098*   2101*   2103*   2177*
                        2180    2289*   2290    2292    2294    2296    2298    2300    2302    2506    2507*   2512    2517
                        2520*   2572    2580*   2584    2585    2587*   2588*   2589    2611*   2711    2721*   2725    2741
                        2742    2755*   2833    2834*   2835*   2842*   2843*   2844*   2845*   2846*   2847    2852    2857*
                        2859*   2863    2865    3081    3106*   3131    3132*   3133    3134*   3135*   3136*   3137*
R1      =%000001         402#    867*    868     869*    873*    874*    875     876*    918*    922*    1325*   1328    1348
                        1371*   1815*   1818*   1820*   1860    1877*   1880*   1882*   1921    2573    2610*   2712    2725*
                        2726    2730    2754*   3082    3105*
R2      =%000002         403#   1324*   1332    1333    1334    1370*   1816*   1821*   1827    1878*   1883*   1889    1939*
                        1944*   1950    2713    2724*   2728*   2731    2738*   2739*   2740    2745*   2753*   3093    3104*
R3      =%000003         404#   2654    2663*   2669*   2670*   2673*   2678*   2679*   2680    2689*   2714    2722*   2723*
                        2737*   2740*   2749*   2750*   2752*   3045    3046*   3047    3050*   3051    3055    3057    3059*
                        3061*   3084    3103*
R4      =%000004         405#   1836    2655    2657*   2658*   2659*   2660    2661*   2675    2677*   2685*   2688*   3085
                        3102*
R5      =%000005         406#   2194*   2195*   2656    2662*   2664*   2666*   2667*   2668*   2669    2687*   2715    2717*
                        2719*   2726*   2730*   2745    2751*   3086    3101*
R6      =%000006         407#    409     821*    822*    823
R7      =%000007         408#    410
SAMAK   013242          2244    2285#
SAMDN   013020          2235#
SDCM    013256          2288#
SETUP1  002062           861#
SETUP2  002126           868#    871
SETUP3  002156           875#    878
SET13   014076          2422    2425#
SOFCSR  013236          1448*   1464*   1481*   1501*   1521*   1555*   1589*   1623*   1657*   1691*   1726*   1748*   1770*
                        1792*   2204*   2205    2274#   2286    2289
SP      =%000006         409#    825*    842*    850*    854     899*    925     1205    1229    1234    1255    1259    1291
                        1285    1412    1413    1422    2014    2141*   2147*   2174*   2226*   2241*   2257    2506*   2507
                        2517*   2519    2520    2521*   2523    2525    2527    2533    2535*   2537*   2545*   2549    2553
                        2554    2558    2572*   2573*   2580    2581*   2592    2593*   2594*   2604    2605*   2610    2611
                        2646*   2647    2648    2649*   2654*   2655*   2656*   2662    2687    2688    2689    2690*   2691*
                        2711*   2712*   2713*   2714*   2715*   2716*   2717    2720*   2733    2735*   2737    2747    2749
                        2751    2752    2753    2754    2755    2757*   2758*   2789    2810*   2813*   2833*   2838*   2859
                        2863*   2893*   2896    2898    2899    2928    2929    2933*   2950*   2951*   2952    2959*   2962*
                        2963*   2967*   2968*   2972    2975*   2979    2981    2983    2984*   2991    2993    2995*   2996
                        2998*   2999*   3000*   3001*   3002*   3017*   3018*   3021*   3022*   3023    3027*   3028*   3029
                        3032    3034    3036*   3045*   3050    3061    3062*   3063*   3064*   3081*   3082*   3083*   3084*
                        3085*   3086*   3087*   3088    3096*   3100    3101    3102    3103    3104    3105    3106    3111*
                        3131*   3132
STACK = 001100           384#    825     899
START   001620           816     818#
START1  001604           507     815#
START2  001614           505     817#
STKLMT= 177774           395#
```

```
ST'ST54  001602            813*   815*   817*   901
SWR      001140            568*   823    844*   846    952*   859*   889    2104   2124   2129   2138   2785   2792
                          2804   2808   2888   2902   2904   2910   2917   2946   2983*  3087   3100*
SWRCK    015776           2794   2820*
SWREG    000176            503*   852    889    2946   2959
SW0    = 000001            448*
SW00   = 000001            438*   448
SW01   = 000002            437*   447
SW02   = 000004            436*   446
SW03   = 000010            435*   445
SW04   = 000020            434*   444
SW05   = 000040            433*   443
SW06   = 000100            432*   442
SW07   = 000200            431*   441
SW08   = 000400            430*   440
SW09   = 001000            429*   439
SW1    = 000002            447*
SW10   = 002000            428*
SW11   = 004000            427*
SW12   = 010000            426*
SW13   = 020000            425*
SW14   = 040000            424*
SW15   = 100000            423*
SW2    = 000004            446*
SW3    = 000010            445*
SW4    = 000020            444*
SW5    = 000040            443*
SW6    = 000100            442*
SW7    = 000200            441*
SW8    = 000400            440*
SW9    = 001000            439*
TAM      013232           1444*  1478*  1497*  1518*  1526   1540   1543*  1552*  1560   1574   1577*  1586*  1594
                          1608   1611*  1620*  1628   1642   1645*  1654*  1662   1676   1679*  1688*  1696   1710
                          1713*  1722*  1744*  1766*  1788*  2204   2246   2259   2272*
TBITVE=  000014            481*
TEST54   012054            904   2052*
TKVEC  = 000060            488*
TPVEC  = 000064            489*
TRAPVE=  000034            487*   831*   832*
TRTVEC=  000014            482*
TSTNUM   001600            812*  2820*  3281   3283   3285   3288
TST1     002344            909*
TST10    003140           1028*
TST11    003232           1039   1046*
TST12    003320           1062*
TST13    003364           1069   1075*
TST14    003454           1085   1092*
TST15    003516           1098   1104*
TST16    003636           1127*
TST17    003764           1151*
TST2     002454            932*
TST20    004102           1167   1174*
TST21    004170           1183   1190*
TST22    004340           1216*
TST23    004510           1243*
TST24    004652           1269*
```

```
TST25   005010          1294#
TST26   005142          1320#
TST27   005514          1369    1376    1384#
TST3    002554           951#
TST30   005606          1394    1400#
TST31   006052          1442#
TST32   006142          1452    1460#
TST33   006224          1468    1477#
TST34   006316          1496#
TST35   006412          1515#
TST36   006562          1531    1536    1541    1549#
TST37   006732          1565    1570    1575    1583#
TST4    002626           959     965#
TST40   007102          1599    1604    1609    1617#
TST41   007252          1633    1638    1643    1651#
TST42   007422          1667    1672    1677    1685#
TST43   007572          1701    1706    1711    1719#
TST44   007706          1734    1741#
TST45   010022          1756    1763#
TST46   010136          1778    1785#
TST47   010252          1800    1807#
TST5    002720           983#
TST50   010606          1823    1835    1848    1857    1869#
TST51   011134          1885    1896    1909    1918    1930#
TST52   011500          1946    1958    1971    1980    1992#
TST53   011736          2033#
TST54   012054          2056#
TST55   012270          2073    2079    2086    2097#
TST6    002764           990     996#
TST7    003052          1012#
TYPDS = 104404          2175    3151#
TYPE  = 104400           883    2106    2126    2131    2140    2146    2173    2176    2528    2681    2756    2787    2795
                        2832    2849    2851    2854    2856    2860    2867    2957    2958    2961    2974    2985    3004
                        3053    3056    3060    3109    3147#
TYPOC = 104401          2840    2864    2960    3148#
TYPON = 104403          3150#
TYPOS = 104402          2143    2149    3149#
XADC    013216          2134*   2136*   2141    2232*   2233*   2234    2235    2266#   2288    2313
XTEMP   013224          2269#   2339*   2345    2346*   2347*   2350    2352*   2353*   2355*   2356    2368*   2375    2376*
                        2377*   2380*   2381    2394*   2399    2400*   2401*   2402    2414*   2426    2427*   2428*   2429
                        2439*   2445    2446*   2447*   2450*   2451    2464*   2471    2472*   2473*   2476*   2477
XYAD    013222          2120*   2136    2137    2268#
XYHOLD  013226          1305*   1334*   1858*   1919*   1981*   2235*   2238*   2240*   225.    2270#   3283    3285
XYPTRN  013214          2206*   2227*   2229*   2230    2232    2265#
YADC    013220          2135*   2137*   2147    2234*   2240    2267#   2314    2329    2340    2369    2395    2415    2440
                        2465
$APTHD  001000           528     534#
$ASTAT= ****** U        2598    2613
$ATYC   014712          2569    2571#
$ATY1   014666          2567#
$ATY3   014674          2513    2568#
$ATY4   014704          2570#   2800
$AUTOB  001134           565#    893#   2954    3075
$BASE   001250           629#    867
$BDADR  001122           560#    919#    934#    954#    967#    985#    998#   1014#   1036#   1040#   1048#   1064#   1082#
                        1086#   1093#   1110#   1116#   1134#   1140#   1157#   1179#   1194#   1219#   1245#   1271#   1299#
```

```
                    1335* 1346* 1353* 1355* 1391* 1410* 1453  1469  1487  1506  1528  1562  1596
                    1630  1664  1698  1731  1753  1775  175   1831* 1843* 1852* 1859* 1893* 1904*
                    1913* 1954* 1966* 1975* 1982* 2006* 2038* 2060* 2114* 2258* 3281  3283  3285
$BDDAT  001126       562*  915*  940*  941*  942   958*  973*  974   989*  1002* 1003  1018* 1019
                    1034* 1038* 1052* 1053  1068* 1080* 1084* 1097* 1112* 1113  1118* 1136* 1137
                    1142* 1161* 1168* 1184* 1201* 1202* 1208* 1235* 1261* 1296* 1308* 1309  1336*
                    1337  1343* 1344  1350* 1351  1361* 1367* 1378* 1418* 1419* 1426* 1427* 1431*
                    1432* 1433  1533  1567  1601  1635  1669  1703  1832* 1833* 1845* 1854* 1861*
                    1862  1894* 1906* 1915* 1922* 1923  1956* 1968* 1977* 1984* 1985  2010* 2011*
                    2018* 2019* 2025* 2042* 2048* 2069* 2074* 2082* 2087* 2116* 2261* 3281  3285
$BELL   001164       580*  2787  2816
$CDWI   001254       631*
$CHARC  014662      2530* 2540* 2547  2556* 2561*
$CKSWR  016432      2946* 3155
$CMTAG  001100       548*  820   821   829   835   836
$CM3  =  000000      578*
$CNTLG  017157      2957  3070*
$CNTLU  017152      2974  3069*
$CPUOP  001222       603*
$CRLF   001171       582*  2529  2564  2795  2816  2832  2851  2856  2860  2985  3069
$DBLK   015576      2722  2756  2764*
$DEVCT  001204       594*
$DEVM   001252       630*
$DOAGN  012656      2169  2178  2184*
$DTBL   015566      2725  2760*
$ENDAD  012646       514  2180*
$ENDCT  012614      2171*
$ENDMG  012665      2173  2188*
$ENULL  012662      2176  2187*
$ENV    001214       599*  887   2508  2576  2600  2797
$ENVM   001215       600*  857   2510  3515  2578
$EOP    012560      2050  2161*
$EOPCT  012606      2168* 2172
$ERFLG  001103       551*  2782* 2816  2876  2906  2908  2914* 2936
$ERMAX  001115       557*  837*  2908  2931* 2936
$ERROR  015606       829   2779*
$ERRPC  001116       558*  2789* 2790* 2791  2816  2838  3281  3283  3295  3288
$ERRTB  001256       648*  2846
$ERRTY  016014      2821  2831*
$ERTTL  001112       555*  2788* 2816
$ESCAP  001162       579*  836*  2811  2813  2816  2930*
$ETABL  001214       598*
$ETEND  001256       540   632*
$FATAL  001176       591*  2604*
$FFLG   015132      2567* 2570* 2598  2607* 2615*
$FILLC  001156       576*  2533  2564
$FILLS  001155       575*  2564
$GDADR  001120       559*  1453  1469  1487  1506  1528  1562  1596  1630  1664  1698  1731  1753
                    1775  1797  1920* 2245* 2251  3283
$GDDAT  001124       561*  914*  937*  938*  942   955*  970*  971*  974   986*  1000* 1001  1003
                    1006* 1016* 1017  1019  1022* 1030* 1050* 1051  1053  1056* 1065* 1076* 1094*
                    1111* 1113  1117* 1135* 1137  1141* 1158* 1164* 1180* 1226* 1260* 1297* 1305
                    1306  1309  1314* 1333* 1337  1341* 1342* 1344  1348* 1349* 1351  1356  1360*
                    1366* 1377* 1417* 1425* 1430* 1433  1533  1567  1601  1635  1669  1703  1844*
                    1853* 1860* 1862  1905* 1914* 1921* 1923  1955* 1967* 1976* 1983* 1985  2009*
                    2017* 2022* 2039* 2047* 2076* 2089* 2115* 2253* 2255* 3281  3285
```

DZNCA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
$GET42  012636          2177#
$GTSWR  016502          2958#   3153
$HD   = 000003           366     367
$HIBTS  001000           535#
$ICNT   001104           552#   2921*   2922    2924*   2935
$ILLUP  017352          3079    3095    3114#
$INTAG  001135           566#   2986    3075
$ITEMB  001114           556#   2791*   2799    2816    2835
$LF     001172           583#   2564    2816    3060    3069
$LFLG   015131          2608*   2614#
$LPADR  001106           553#    838*    910*   2057*   2912*   2928*   2933    2935
$LPERR  001110           554#    839*    913*    935*    968*    999*   1015*   1049*   1106*   1129*   1307*   1443*   1461*
                        1480*   1499*   1517*   1551*   1585*   1619*   1653*   1687*   1721*   1743*   1765*   1787*   1809*
                        1871*   1932*   2058*   2810    2912    2929*   2935
$MADR1  001226           616#
$MADR2  001232           620#
$MADR3  001236           623#
$MADR4  001242           626#
$MAIL   001174           536     540     589#    856     887     911     2508    2797    2927
$MAMS1  001224           610#
$MAMS2  001230           618#
$MAMS3  001234           621#
$MAMS4  001240           624#
$MBADR  001002           536#
$MFLG   015130          2568*   2574    2609*   2613#
$MNEW   017175          2961    3073#
$MSGAD  001210           596#   2584*   2587
$MSGLG  001212           597#   2589*
$MSGTY  001174           590#   2582    2590*   2602    2606*
$MSWR   017164          2958    3071#
$MTYP1  001225           611#
$MTYP2  001231           619#
$MTYP3  001235           622#
$MTYP4  001241           625#
$MXCNT  016430          2925    2935#
$NULL   001154           574#   2535    2564
$NWTST= 000001           906#    929#    948#    962#    980#    993#   1009#   1025#   1043#   1059#   1072#   1089#   1101#
                        1124#   1148#   1171#   1187#   1213#   1240#   1266#   1291#   1317#   1381#   1397#   1439#   1457#
                        1474#   1493#   1512#   1546#   1580#   1614#   1648#   1682#   1716#   1738#   1760#   1782#   1804#
                        1866#   1927#   1989#   2030#   2053#   2094#
$OCNT   015356          2653*   2682*   2695#
$OMODE  015360          2648*   2652*   2657    2660*   2671*   2697#
$OVER   016414          2889    2905    2913    2923    2932#
$PASS   001202           593#    856*   2165*   2166*   2174    2187    2919    2936
$PASTM  001006           538#
$PWRAD  017346          3112#
$PWRDN  017206           833    3079#   3107
$PWRMG  017342          3110#
$PWRUP  017260          3089    3095#
$QUES   001170           581#   2564    2816    3004    3053    3069
$RDCHR  016714          3017#   3156
$RDDEC= ****** U        3158
$RDLIN  017034          3045#   3157
$RDOCT= ****** U        3158
$RDSZ = 000010          3038#
$RTNAD  012660          2186#
```

```
$R2A  = ****** U          3158
$SAVRE= ****** U          3158
$SAVR6  017356            3088*  3096   3097*  3098*  3116#
$SCOPE  016150             827   2885#
$SETUP= 000117             814#   826    827    829    831    833    835    836    838    983    984   2163   2780
                          2907   2815   2886   2941   3075
$STUP = 177777             814#
$SVLAD  016360            2897   2926#
$SVPC = 000210             512#   517
$SWR  = 167400             366    367#   368#   374    375    376    377    378    379    380    578    579    580
                           835    836    838    839    910    933    952    966    984    997   1013   1029   1047
                          1063   1076   1093   1105   1128   1152   1175   1191   1217   1244   1270   1295   1321
                          1385   1401   1443   1461   1478   1497   1516   1550   1584   1618   1652   1686   1720
                          1742   1764   1786   1808   1870   1931   1993   2034   2057   2098   2158   2164   2179
                          2185   2187   2771   2772   2773   2774   2775   2785   2792   2804   2808   2816   2877
                          2878   2879   2880   2881   2888   2900   2902   2903   2906   2907   2908   2915   2916
                          2917   2929   2932   2935   3113
$SWREG  001216             601#   859
$SWRMK= 000000             380    381   2881   2882   2904
$TESTN  001200             592#   911*  2927*
$TIMES  001160             578#   835*   933*   952*   966*   984*   997*  1013*  1029*  1047*  1063*  1105*  1128*
                          1175*  1244*  1270*  1295*  1321*  1516*  1550*  1584*  1618*  1652*  1686*  1720*  1742*
                          1764*  1787*  1808*  1870*  1931*  1993*  2164*  2915*  2922   2925*  2935
$TKB    001146             571#  2939   2950   2967   3021   3027
$TKS    001144             570#  2939   2948   2964   2988*  3019   3025
$TN   = 000056             366#   369#   906    910#   911    929    933#   948    952#   959    962    966#   980
                           984#   990    993    997#  1009   1013#  1025   1029#  1039   1043   1047#  1059   1063#
                          1069   1072   1076#  1085   1089   1093#  1098   1101   1105#  1124   1128#  1148   1152#
                          1167   1171   1175#  1183   1187   1191#  1213   1217#  1240   1244#  1266   1270#  1291
                          1295#  1317   1321#  1369   1376   1381   1385#  1394   1397   1401#  1439   1443#  1452
                          1457   1461#  1468   1474   1478#  1493   1497#  1512   1516#  1531   1536   1541   1546
                          1550#  1565   1570   1575   1580   1584#  1599   1604   1609   1614   1618#  1633   1638
                          1643   1648   1652#  1667   1672   1677   1682   1686#  1701   1706   1711   1716   1720#
                          1734   1738   1742#  1756   1760   1764#  1778   1782   1786#  1800   1804   1808#  1823
                          1835   1848   1857   1866   1870#  1885   1896   1909   1918   1927   1931#  1946   1958
                          1971   1980   1989   1993#  2030   2034#  2053   2057#  2073   2079   2086   2094   2098#
$TPB    001152             573#  2553*  2564
$TPFLG  001157             577#  2502   2564
$TPS    001150             572#  2551   2564
$TRAP   017412             831   3131#
$TRP  = 000011            3139#  3148#  3149#  3150#  3151#  3152#  3153   3154#  3155   3156#  3157#  3158#
$TRPAD  017434            3136   3146#
$TSTM   001004             537#
$TSTNM  001102             550#   903#   912*  2163*  2784   2816   2820   2876   2904   2926*  2927   2932   2936
$TTYIN  017142            3046   3047   3064   3068#
$TYPBN= ****** U          3152
$TYPDS  015362            2710#  3151
$TYPE   014404            2502#  2595   3139   3147
$TYPEC  014616            2532   2539   2546   2551#  2552   2990
$TYPEX  014664            2557   2559   2562#
$TYPOC  015160            2651#  3148
$TYPON  015174            2650   2653#  3150
$TYPOS  015134            2646#  3149
$UNIT   001206             595#
$UNITM  001010             539#
$USWR   001220             602#
```

DZNCA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
$VECT1  001244            627#    861     873
$VECT2  001246            629#
$XTSTR  016164           2891#
$$SET4= 000000           2179#
$DFILL  015357           2647*   2651*   2661    2696#
$$OCAT= ****** U         2794    2888
.       = 071200          497#    501#    506#    512     513#    515#    517#    518#    524     525#    527#    529#    547#
                          584     824     838     839    2187    2191    2564    2616#   2764#   2816    2870#   2935    2936
                         2939    3068#   3069    3075    3091    3115    3159#
.$ASTA= ****** U         2568    2571
.$X   = 001000            524#    529
```

```
COMMEN   491#
ENDCOM   491#
ERROR    385#  926  944  960  976  991 1005 1021 1037 1041 1055 1070 1083 1087 1099
         1115 1120 1139 1144 1163 1169 1185 1203 1209 1236 1262 1297 1311 1339 1347
         1354 1362 1368 1379 1395 1420 1428 1435 1455 1471 1489 1508 1530 1535 1564
         1569 1598 1603 1632 1637 1666 1671 1700 1705 1733 1755 1777 1799 1834 1846
         1855 1864 1895 1907 1916 1925 1957 1969 1978 1987 2012 2020 2026 2043 2049
         2071 2077 2084 2090 2117
ESCAPE   491#
GETPRI   491#
GETSWR   491#  884#
MULT     491#
NEWTST   491#  906  929  948  962  980  993 1009 1025 1043 1059 1072 1089 1101 1124
         1148 1171 1187 1213 1240 1266 1291 1317 1381 1397 1439 1457 1474 1493 1512
         1546 1580 1614 1648 1682 1716 1738 1760 1782 1804 1866 1927 1989 2030 2053
         2094
OK       787#  942  974 1003 1019 1053 1113 1137 1309 1337 1344 1351 1433 1532 1566
         1600 1634 1668 1702
OK1      787# 1453 1469 1487 1506 1528 1562 1596 1630 1664 1698 1731 1753 1775 1797
POP      491# 2610 2611 2751 3100 3101
PUSH     491# 2571 2573 2594 2710 3081 3087
REPORT   491#
SCOPE    386#  932  951  965  983  996 1012 1028 1046 1062 1075 1092 1104 1127 1151
         1174 1190 1216 1243 1269 1294 1320 1384 1400 1442 1460 1477 1496 1515 1549
         1583 1617 1651 1685 1719 1741 1763 1785 1807 1869 1930 1992 2033 2097 2162
SETPRI   491#
SETTRA  3139#  3148 3149 3150 3151 3153 3155 3156 3157
SETUP    491#  818
SKIP     491#  959  990 1039 1069 1085 1098 1167 1193 1369 1376 1394 1452 1468 1531
         1536 1541 1565 1570 1575 1599 1604 1609 1633 1638 1643 1667 1672 1677 1701
         1706 1711 1734 1756 1778 1800 1823 1835 1848 1857 1885 1896 1909 1918 1946
         1958 1971 1980 2073 2079 2086
SLASH    491#
SPACE    491#
STARS    491#  510  521  523  530  543  584  587  906  908  929  931  948  950  962
         964  980  982  993  995 1009 1011 1025 1027 1043 1045 1059 1061 1072 1074
         1089 1091 1101 1103 1124 1126 1148 1150 1171 1173 1187 1189 1213 1215 1240
         1242 1266 1268 1291 1293 1317 1319 1381 1383 1397 1399 1439 1441 1457 1459
         1474 1476 1493 1495 1512 1514 1546 1548 1580 1582 1614 1616 1648 1650 1682
         1684 1716 1718 1738 1740 1760 1762 1782 1784 1904 1906 1866 1968 1927 1929
         1989 1991 2030 2032 2053 2055 2094 2096 2155 2191 2193 2199 2202 2209 2225
         2277 2294 2309 2312 2318 2321 2325 2328 2332 2337 2361 2366 2387 2392 2407
         2412 2432 2437 2457 2462 2487 2566 2623 2700 2767 2816 2819 2826 2873 2939
         2941 3009 3038 3077 3093 3125
SWRSJ    491#  840#
TRMTRP  3139#
TYPBIN   491#
TYPDEC   491#  2174
TYPNAM   491#  879
TYPNUM   491#
TYPOCS   356#  491#  2141 2147
TYPOCT   491#  2838 2862 2959
TYPTXT   491#
$$CMRE   541#
$$CMTM   541#
$$ESCA   491#
```

```
$$NEWT    491#   906   929   948   962   980   993  1009  1025  1043  1059  1072  1089  1101  1124
         1148  1171  1197  1213  1240  1266  1291  1317  1381  1397  1439  1457  1474  1493  1512
         1546  1590  1614  1648  1692  1716  1738  1760  1782  1804  1866  1927  1989  2030  2053
         2094
$$SET     3139#  3148  3149  3150  3151  3153  3155  3156  3157
$$SETM     856#
$$SKIP    491#   959   990  1039  1069  1085  1098  1167  1183  1369  1376  1334  1452  1468  1531
         1536  1541  1565  1570  1575  1599  1604  1609  1633  1638  1643  1667  1672  1677  1701
         1706  1711  1734  1756  1778  1800  1823  1835  1848  1857  1885  1896  1909  1918  1946
         1958  1971  1980  2073  2079  2086
.EQUAT     356#   381
.HEADE     356#
.SETUP     356#   814
.SWRHI     356#   370
.SWRLO     381#
.$ACTI     356#   508
.$APTB     356#   585#
.$APTH     356#   519
.$APTY     356#  2564
.$CATC     356#   495
.$CMTA     356#   541
.$EOP      356#  2153
.$ERRO     356#  2765
.$ERRT     356#  2824
.$POWE     356#  3075
.$RDCC     356#
.$READ     356#  2936
.$SCOP     356#  2871
.$TRAP     356#  3123
.$TYPD     356#  2698
.$TYPE     356#  2495
.$TYPO     356#  2621
```

ADD	969	976	977	1342	1372	1490	1509	1537	1571	1605	1639	1673	1707	1735	1757
	1779	1801	2226	2241	2345	2356	2375	2381	2399	2402	2426	2429	2445	2451	2471
	2477	2521	2581	2593	2605	2649	2659	2730	2846	2975	2984				
ASL	2418	2843	2844	2845	2998	2999	3000	3135							
ASLB	2735														
ASR	2227	2343	2344	2372	2373	2374	2383	2398	2419	2420	2443	2444	2468	2469	2479
	2480	2588													
BCC	979	1491	1510	1736	1758	1780	1802	2736							
BCS	1057	1538	1572	1606	1640	1674	1708								
BEQ	858	888	902	923	943	959	975	990	1004	1020	1035	1039	1054	1069	1081
	1085	1096	1114	1119	1138	1143	1167	1183	1232	1310	1338	1345	1352	1357	1359
	1376	1394	1424	1434	1454	1470	1488	1507	1529	1534	1563	1568	1597	1602	1631
	1636	1665	1670	1699	1704	1732	1754	1776	1798	1823	1828	1842	1851	1863	1885
	1890	1903	1912	1924	1946	1951	1965	1974	1986	2016	2178	2231	2237	2247	2252
	2260	2291	2293	2295	2297	2299	2301	2303	2349	2351	2379	2449	2475	2511	2524
	2559	2575	2579	2599	2601	2676	2783	2786	2809	2812	2848	2853	2866	2903	2905
	2907	2911	2920	2955	2982	2997									
BGE	2923														
BGT	2169	2683	2744	2394	3035										
BHI	2909														
BIC	862	874	938	941	971	1202	1419	1427	1432	2011	2019	2074	2087	2166	2233
	2248	2315	2347	2353	2358	2377	2384	2401	2404	2423	2428	2447	2454	2473	2573
	2951	2968	2995	3022	3028	3036									
BIS	2067	2080	2204	2238	2352	2355	2380	2425	2450	2476	2678	2679	2738	2739	3002
BISB	2935														
BIT	1159	1166	1182	1206	1358	1364	1375	1393	1829	1841	1850	1891	1902	1911	1952
	1964	1973	2023	2104	2112	2124	2129	2138	2251	2286	2290	2292	2294	2296	2298
	2300	2302	2360	2786	2792	2808	2888	2902	2910	2917					
BITB	957	2510	2515	2547	2578										
BLOS	3048														
BLT	2538	2684	2727	2743	2992	3033									
BMI	2041	2075	2734												
BNE	824	847	871	878	882	886	890	946	1122	1146	1160	1207	1257	1283	1365
	1416	1527	1541	1561	1575	1595	1609	1629	1643	1663	1677	1697	1711	1819	1830
	1881	1892	1942	1953	2008	2024	2070	2083	2092	2102	2105	2113	2125	2130	2139
	2196	2228	2287	2509	2516	2518	2526	2534	2548	2555	2577	2583	2586	2603	2674
	2732	2793	2798	2836	2858	2889	2918	2947	2953	2973	2990	2997	3024	3030	3052
	3058	3099													
BPL	1539	1573	1607	1641	1675	1709	1837	1898	1960	2046	2089	2111	2123	2422	2503
	2552	2672	2718	2748	2805	2949	2965	3020	3026						
BR	816	849	892	895	924	1204	1233	1258	1284	1363	1369	1373	1421	1429	1452
	1468	1486	1505	1525	1531	1536	1542	1544	1559	1565	1570	1576	1578	1593	1599
	1604	1610	1612	1627	1633	1638	1644	1646	1661	1667	1672	1678	1690	1695	1701
	1706	1712	1714	1730	1734	1752	1756	1774	1778	1796	1800	1935	1948	1957	1996
	1909	1918	1958	1971	1980	2013	2021	2073	2079	2086	2118	2127	2132	2152	2254
	2354	2424	2505	2531	2541	2550	2557	2569	2591	2650	2665	2686	2729	2746	2803
	2841	2868	2891	2897	2900	2913	2916	2976	3003	3005	3031	3054	3091	3115	
BVC	1007	1023	1315												
CLR	817	822	835	836	856	914	915	955	969	986	1030	1065	1076	1094	1117
	1141	1161	1164	1180	1208	1211	1238	1260	1264	1289	1304	1331	1360	1367	1377
	1390	1409	1425	1437	1444	1445	1447	1478	1481	1482	1497	1500	1518	1552	1596
	1620	1654	1688	1722	1744	1766	1788	1814	1815	1816	1833	1838	1844	1853	1876
	1877	1878	1894	1899	1905	1914	1937	1938	1939	1956	1961	1967	1976	1997	2001
	2017	2025	2028	2042	2047	2089	2116	2134	2135	2163	2164	2250	2295	2322	2663
	2721	2724	2834	2915	2930	2962	2963	3097							
CLRB	2530	2556	2607	2608	2609	2750	2914	3059							

DZNCA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
CMP      823   846   870   877   889   925   942   974  1003  1019  1053  1113  1137  1205  1229
        1231  1234  1255  1259  1281  1285  1309  1337  1344  1351  1412  1413  1422  1433  1453
        1469  1487  1506  1528  1533  1562  1567  1596  1601  1630  1635  1664  1669  1698  1703
        1731  1753  1775  1797  1836  1862  1897  1923  1959  1985  2014  2015  2230  2257  2742
        2998  2922  2946  2952  2972  2979  2991  2993  3023  3029  3032  3034  3047
CMPB     997  2508  2523  2525  2533  2554  2558  2576  2797  2904  2908  2954  2986  3051  3057
COM     1370  1820
COMB    1882  1943
DEC      922  1006  1022  1056  1314  1415  1423  1822  1884  1945  2091  2101  2167  2195  2842
DECB    2537  2540  2671  2682
EMT      385
HALT     501  2504  2806  3090  3114
INC      881   945  1121  1145  1256  1282  1349  1371  1543  1577  1611  1645  1679  1713  1818
        2007  2165  2606  2677  2685  2728  2788  2921  3001  3098
INCB    1880  1941  2560  2782  2926
IOT      386
JMP      505   507   904  2050  2185  2304  2305  2306  2307
JSR     1449  1450  1451  1465  1466  1467  1483  1484  1485  1502  1503  1504  1522  1523  1524
        1556  1557  1558  1590  1591  1592  1624  1625  1626  1658  1659  1660  1692  1693  1694
        1727  1728  1729  1749  1750  1751  1771  1772  1773  1793  1794  1795  2068  2091  2109
        2180  2244  2338  2341  2367  2370  2393  2396  2413  2416  2438  2441  2463  2466  2479
        2513  2532  2539  2546  2595  2794  2800  2821  2990
MOV      815   821   825   827   828   829   830   831   832   833   834   838   839   842   843
         844   845   850   852   853   854   859   861   864   866   867   868   872   873   875
         898   899   910   911   913   916   917   919   919   927   933   934   935   936   937
         939   940   952   953   954   956   958   966   967   968   970   972   973   984   985
         987   989   997   998   999  1000  1001  1002  1013  1014  1015  1016  1017  1018  1029
        1031  1032  1034  1036  1038  1040  1047  1048  1049  1050  1051  1052  1063  1064  1066
        1068  1077  1078  1079  1080  1082  1084  1086  1093  1095  1096  1097  1105  1106  1107
        1108  1109  1110  1111  1112  1116  1118  1128  1129  1130  1131  1132  1133  1134  1135
        1136  1140  1142  1153  1154  1155  1156  1157  1158  1165  1168  1175  1176  1177  1178
        1179  1184  1191  1192  1193  1194  1195  1196  1197  1198  1199  1200  1201  1210  1217
        1218  1219  1220  1221  1222  1223  1224  1225  1226  1227  1228  1235  1237  1244  1245
        1246  1247  1248  1249  1250  1251  1252  1253  1254  1261  1263  1270  1271  1272  1273
        1274  1275  1276  1277  1278  1279  1280  1286  1288  1295  1296  1297  1298  1299  1300
        1301  1302  1303  1305  1306  1307  1308  1321  1322  1323  1324  1325  1326  1327  1328
        1329  1330  1332  1333  1334  1335  1336  1341  1343  1346  1348  1350  1353  1355  1361
        1366  1378  1385  1386  1387  1388  1389  1391  1392  1401  1402  1403  1404  1405  1406
        1407  1408  1410  1411  1414  1417  1418  1426  1430  1431  1436  1443  1446  1448  1461
        1462  1463  1464  1479  1480  1498  1499  1501  1516  1517  1519  1520  1521  1550  1551
        1553  1554  1555  1584  1585  1587  1588  1589  1618  1619  1621  1622  1623  1652  1653
        1655  1656  1657  1686  1687  1689  1690  1691  1720  1721  1723  1724  1725  1726  1742
        1743  1745  1746  1747  1748  1764  1765  1767  1769  1769  1770  1786  1787  1789  1790
        1791  1792  1808  1809  1810  1811  1812  1813  1821  1824  1825  1831  1832  1839  1840
        1843  1845  1849  1852  1854  1858  1859  1860  1861  1870  1871  1872  1873  1874  1875
        1883  1886  1887  1893  1900  1901  1904  1906  1910  1913  1915  1919  1920  1921  1922
        1931  1932  1933  1934  1935  1936  1944  1947  1948  1954  1955  1962  1963  1966  1969
        1972  1975  1977  1981  1982  1983  1984  1993  1994  1995  1996  1998  1999  2000  2003
        2004  2005  2006  2009  2010  2018  2022  2027  2034  2035  2036  2037  2038  2039  2044
        2048  2057  2058  2059  2060  2061  2062  2063  2064  2065  2066  2069  2076  2082  2098
        2099  2100  2103  2108  2114  2115  2120  2121  2141  2147  2170  2174  2177  2194  2203
        2205  2206  2229  2232  2234  2243  2245  2249  2253  2255  2256  2258  2261  2262  2289
        2339  2346  2368  2376  2394  2400  2414  2427  2439  2446  2464  2472  2506  2507  2512
        2520  2535  2572  2573  2580  2584  2589  2590  2592  2594  2604  2610  2611  2646  2654
        2655  2656  2662  2669  2687  2688  2689  2690  2691  2711  2712  2713  2714  2715  2716
        2717  2722  2725  2745  2751  2752  2753  2754  2755  2757  2758  2784  2789  2810  2813
```

```
            2933  2838  2847  2852  2857  2859  2863  2893  2894  2896  2899  2912  2924  2925  2929
            2929  2932  2933  2959  2983  2988  3017  3018  3045  3046  3061  3062  3063  3064  3073
            3080  3081  3082  3083  3084  3085  3086  3087  3088  3089  3095  3096  3100  3101  3102
            3103  3104  3105  3106  3107  3108  3111  3131  3132  3136
MOVB         837   893   903   912  2136  2137  2235  2240  2288  2313  2314  2329  2340  2369  2395
            2415  2440  2465  2517  2545  2553  2567  2568  2570  2647  2648  2651  2652  2653  2657
            2660  2661  2680  2720  2723  2737  2740  2749  2791  2799  2820  2927  2931  2950  2967
            3021  3027  3050  3055  3134
NEG         2658  2719
NOP          909  1826  1888  1949  2056  2181  2182  2183
RESET        900   957   988  1033  1067  1181  1374  2179
ROL         2664  2666  2667  2668  2670
ROR         2453
RTI          851  2522  2692  2759  2815  2934  2989  3037  3065  3113
RTS         2197  2207  2263  2316  2323  2330  2357  2359  2382  2385  2403  2405  2430  2452  2455
            2478  2481  2562  2612  2823  2861  3137
SUB          865  1230  2587  2726  2790
SWAB         863  2342  2371  2397  2417  2442  2467
TRAP        3139  3148  3149  3150  3151  3153  3155  3156  3157
TST          885   901   920   921  1356  1526  1540  1560  1574  1594  1608  1628  1642  1662  1676
            1696  1710  1827  1889  1950  2040  2045  2122  2236  2246  2259  2348  2378  2421  2448
            2474  2519  2527  2549  2582  2600  2602  2675  2731  2741  2804  2811  2865  2895  2919
            2981  2996  3133
TSTB        2110  2502  2551  2574  2585  2598  2733  2747  2906  2948  2964  3019  3025
.ASCII       581   582
.ASCIZ       580   583   897  2188  2869  3069  3070  3071  3073  3117  3163  3164  3166  3168  3169
            3171  3174  3177  3180  3183  3186  3190  3193  3196  3200  3204  3207  3210  3214  3217
            3220  3227  3229  3234  3237  3240  3245  3251  3254  3261  3268  3276
.BLKB       3068
.BLKW       2764
.BYTE        550   551   556   557   565   566   574   575   576   577   599   600   610   611   618
             619   621   622   624   625  2144  2145  2150  2151  2187  2613  2614  2615  2693  2694
            2695  2696  2801  2802  3066  3067
.DSABL      3006
.ENABL       356  2939
.END        3289
.ENDC        361   377   379   380   381   385   477   491   506   511   515   517   522   524   531
             544   548   550   578   579   580   581   585   588   610   618   621   624   627   628
             629   630   631   634   814   825   826   829   831   833   935   836   838   840   861
             883   889   895   897   907   908   909   910   911   912   930   931   932   933   934
             949   950   951   952   953   960   963   964   965   966   967   981   982   983   984
             985   991   994   995   996   997   998  1010  1011  1012  1013  1014  1026  1027  1029
            1029  1030  1040  1044  1045  1046  1047  1048  1060  1061  1062  1063  1064  1070  1073
            1074  1075  1076  1086  1090  1091  1092  1093  1099  1102  1103  1104  1105  1106  1125
            1126  1127  1128  1129  1149  1150  1151  1152  1168  1172  1173  1174  1175  1176  1184
            1188  1189  1190  1191  1214  1215  1216  1217  1241  1242  1243  1244  1245  1267  1268
            1269  1270  1271  1292  1293  1294  1295  1296  1318  1319  1320  1321  1322  1370  1377
            1382  1383  1384  1385  1395  1398  1399  1400  1401  1440  1441  1442  1443  1453  1458
            1459  1460  1461  1469  1475  1476  1477  1478  1494  1495  1496  1497  1513  1514  1515
            1516  1517  1532  1537  1542  1547  1548  1549  1550  1551  1566  1571  1576  1581  1582
            1583  1584  1585  1600  1605  1610  1615  1616  1617  1618  1619  1634  1639  1644  1649
            1650  1651  1652  1653  1668  1673  1678  1683  1684  1685  1686  1687  1702  1707  1712
            1717  1718  1719  1720  1721  1735  1739  1740  1741  1742  1743  1757  1761  1762  1763
            1764  1765  1779  1783  1784  1785  1786  1787  1801  1805  1806  1807  1808  1809  1924
            1836  1849  1858  1867  1868  1869  1870  1871  1886  1897  1910  1919  1928  1929  1930
            1931  1932  1947  1959  1972  1981  1990  1991  1992  1993  1994  2031  2032  2033  2034
```

```
            2054  2055  2056  2057  2058  2074  2080  2087  2095  2096  2097  2098  2145  2146  2151
            2152  2156  2157  2158  2160  2163  2169  2172  2173  2177  2179  2185  2187  2188  2191
            2192  2194  2200  2203  2210  2226  2278  2285  2310  2313  2319  2322  2326  2329  2333
            2338  2362  2367  2389  2393  2408  2413  2433  2438  2458  2463  2488  2517  2567  2568
            2571  2598  2613  2624  2701  2768  2771  2782  2789  2794  2795  2796  2804  2815  2816
            2817  2820  2827  2842  2871  2874  2877  2882  2888  2890  2901  2904  2905  2906  2908
            2910  2917  2921  2926  2928  2932  2935  2936  2939  2940  2942  2970  3006  3010  3038
            3039  3046  3048  3051  3053  3069  3075  3078  3087  3088  3094  3100  3101  3111  3113
            3117  3126  3132  3135  3147  3148  3149  3150  3151  3152  3153  3154  3155  3156  3157
            3158
.EQUIV       385   386   394   409   410   439   440   441   442   443   444   445   446   447   448
             467   468   469   470   471   472   473   474   475   476
.EVEN        589   897  2616  2870  3122  3280
.IF          357   377   378   379   380   381   383   449   477   504   510   513   515   521   523
             530   543   547   549   578   579   580   584   585   587   610   618   621   624   627
             628   629   630   631   632   634   814   820   825   927   829   831   833   835   836
             838   856   882   883   884   887   896   906   908   910   911   929   931   933   934
             948   950   952   953   959   962   964   966   967   980   982   984   985   990   993
             995   997   998  1009  1011  1013  1014  1025  1027  1029  1030  1039  1043  1045  1047
            1048  1059  1061  1063  1064  1069  1072  1074  1076  1085  1089  1091  1093  1098  1101
            1103  1105  1106  1124  1126  1128  1129  1148  1150  1152  1167  1171  1173  1175  1176
            1183  1187  1189  1191  1213  1215  1217  1240  1242  1244  1245  1266  1268  1270  1271
            1291  1293  1295  1296  1317  1319  1321  1322  1369  1376  1381  1383  1385  1394  1397
            1399  1401  1439  1441  1443  1452  1457  1459  1461  1468  1474  1476  1478  1493  1495
            1497  1512  1514  1516  1517  1531  1536  1541  1546  1548  1550  1551  1565  1570  1575
            1580  1582  1584  1585  1599  1604  1609  1614  1616  1618  1619  1633  1638  1643  1648
            1650  1652  1653  1667  1672  1677  1682  1684  1686  1687  1701  1706  1711  1716  1718
            1720  1721  1734  1738  1740  1742  1743  1756  1760  1762  1764  1765  1778  1782  1784
            1786  1787  1800  1804  1806  1808  1809  1823  1835  1848  1857  1866  1868  1870  1871
            1885  1896  1909  1918  1927  1929  1931  1932  1946  1958  1971  1980  1989  1991  1993
            1994  2030  2032  2034  2053  2055  2057  2058  2073  2079  2086  2094  2096  2098  2144
            2145  2150  2151  2155  2156  2157  2158  2159  2160  2162  2168  2171  2173  2177  2179
            2185  2187  2188  2191  2193  2199  2202  2209  2225  2277  2284  2309  2312  2318  2321
            2325  2328  2332  2337  2361  2366  2387  2332  2407  2412  2432  2437  2457  2462  2497
            2508  2566  2568  2571  2598  2613  2623  2700  2767  2770  2781  2785  2792  2794  2795
            2797  2804  2808  2815  2816  2819  2826  2841  2857  2873  2876  2881  2887  2888  2900
            2902  2903  2904  2906  2907  2908  2917  2919  2927  2929  2934  2935  2936  2938  2940
            2941  2942  2970  3009  3010  3038  3046  3047  3051  3052  3068  3069  3075  3077  3087
            3088  3093  3100  3101  3109  3111  3113  3117  3125  3131  3135  3139  3148  3149  3150
            3151  3152  3153  3155  3156  3157  3158
.IFF          377   379   380   381   383   511   515   517   522   524   531   544   547   550   579
              585   588   825   882   883   906   907   908   909   910   930   931   932   933   949
              950   951   952   960   963   964   965   966   981   982   983   984   991   994   995
              996   997  1010  1011  1012  1013  1026  1027  1028  1029  1040  1044  1045  1046  1047
             1060  1061  1062  1063  1070  1073  1074  1075  1076  1086  1090  1091  1092  1093  1099
             1102  1103  1104  1105  1125  1126  1127  1128  1149  1150  1151  1152  1168  1172  1173
             1174  1175  1184  1188  1189  1190  1191  1214  1215  1216  1217  1241  1242  1243  1244
             1267  1268  1269  1270  1292  1293  1294  1295  1318  1319  1320  1321  1370  1377  1382
             1383  1384  1385  1395  1398  1399  1400  1401  1440  1441  1442  1443  1453  1458  1459
             1460  1461  1469  1475  1476  1477  1478  1494  1495  1496  1497  1513  1514  1515  1516
             1532  1537  1542  1547  1548  1549  1550  1566  1571  1576  1581  1582  1583  1584  1600
             1605  1610  1615  1616  1617  1618  1634  1639  1644  1649  1650  1651  1652  1668  1673
             1678  1683  1684  1685  1686  1702  1707  1712  1717  1718  1719  1720  1735  1739  1740
             1741  1742  1757  1761  1762  1763  1764  1779  1783  1784  1785  1786  1801  1805  1806
             1807  1808  1824  1836  1849  1858  1867  1868  1869  1870  1886  1897  1910  1919  1928
             1929  1930  1931  1947  1959  1972  1981  1990  1991  1992  1993  2031  2032  2033  2034
```

```
            2053    2054    2055    2056    2057    2074    2080    2087    2095    2096    2097    2098    2145    2146    2151
            2152    2156    2159    2163    2169    2172    2187    2192    2194    2200    2203    2210    2226    2278    2285
            2310    2313    2319    2322    2326    2329    2333    2338    2362    2367    2388    2393    2408    2413    2433
            2438    2458    2463    2488    2567    2624    2701    2768    2770    2785    2815    2816    2817    2820    2827
            2842    2871    2974    2901    2904    2905    2908    2935    2936    2939    2942    3010    3012    3017    3038
            3039    3048    3052    3069    3078    3094    3109    3126    3132
.IFT         897    2795    2916    3012    3017
.IFTF        897    2794    2914    2957    3010    3013
.IIF         356     361     366     367     374     375     376     377     380     381     501     584     588     826     829
             835     836     838     839     883    2142    2148    2157    2163    2164    2175    2187    2191    2564    2771
            2772    2773    2774    2775    2780    2807    2815    2816    2839    2864    2877    2878    2879    2880    2981
            2882    2886    2915    2916    2932    2935    2936    2939    2960    3061    3069    3075    3147    3148    3149
            3150    3151    3153    3155    3156    3157
.IRP         814     906     911     929     948     962     980     993    1009    1025    1043    1059    1072    1089    1101
            1124    1148    1171    1187    1213    1240    1266    1291    1317    1381    1397    1439    1457    1474    1493
            1512    1546    1580    1614    1648    1682    1716    1738    1760    1782    1804    1866    1927    1989    2030
            2053    2094    2572    2573    2594    2610    2611    2711    2751    2781    2887    3081    3087    3100    3101
.LIST        356     380     491     501     578     585     588     814     840     883     884     897     906     910     929
             933     948     952     962     966     980     984     993     997    1009    1013    1025    1029    1043    1047
            1059    1063    1072    1076    1089    1093    1101    1105    1124    1128    1148    1152    1171    1175    1187
            1191    1213    1217    1240    1244    1266    1270    1291    1295    1317    1321    1381    1385    1397    1401
            1439    1443    1457    1461    1474    1478    1493    1497    1512    1516    1546    1550    1580    1584    1614
            1618    1648    1652    1682    1686    1716    1720    1738    1742    1760    1764    1782    1785    1804    1808
            1866    1870    1927    1931    1989    1993    2030    2034    2053    2057    2094    2098    2163    2179    2815
            2981    3038    3139    3147    3148    3149    3150    3151    3152    3153    3154    3155    3156    3157    3159
.MACRO       381     541     787     856    3139
.MCALL       356     491     585     840     884
.MEXIT       633
.NLIST       356     380     491     501     578     585     588     814     840     883     884     897     906     910     929
             933     948     952     962     966     980     984     993     997    1009    1013    1025    1029    1043    1047
            1059    1063    1072    1076    1089    1093    1101    1105    1124    1128    1148    1152    1171    1175    1187
            1191    1213    1217    1240    1244    1266    1270    1291    1295    1317    1321    1381    1385    1397    1401
            1439    1443    1457    1461    1474    1478    1493    1497    1512    1516    1546    1550    1580    1584    1614
            1618    1648    1652    1682    1686    1716    1720    1738    1742    1760    1764    1782    1786    1804    1808
            1866    1870    1927    1931    1989    1993    2030    2034    2053    2057    2094    2098    2163    2179    2484
            2815    2881    3038    3139    3147    3148    3149    3150    3151    3152    3153    3154    3155    3156    3157
            3158
.PAGE        508     541     634     649     697     745     787     914     906    2153    2191    2482
.REM           1
.REPT        501
.SBTTL       370     381     495     504     508     519     541     585     634     814     819     879     884     906     929
             948     962     980     993    1009    1025    1043    1059    1072    1089    1101    1124    1148    1171    1187
            1213    1240    1266    1291    1317    1381    1397    1439    1457    1474    1493    1512    1546    1580    1614
            1648    1682    1716    1738    1760    1782    1804    1866    1927    1989    2030    2053    2094    2153    2482
            2485    2564    2621    2698    2765    2824    2871    2936    3075    3123    3139    3162
.TITLE       356
.WORD        501     502     503     516     535     536     537     538     539     540     549     552     553     554     555
             558     559     560     561     562     563     564     567     568     569     590     591     592     593     594
             595     596     597     601     602     603     616     620     623     626     627     628     629     630     631
            2168    2171    2186    2514    2561    2596    2697    2850    2855    3110    3112
```

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

```
*.DZNCA.SEQ/SOL/CRF=DZNCA.P11
RUN-TIME:  64 40 8 SECONDS
RUN-TIME RATIO:  333/112=2.9
CORE USED:  26K  (51 PAGES)
```