DX11-B

DX11-B ADRSNG TST
CZDXJB0

AH-A830B-MC
FICHE 1 OF 1

```
                        .REM    %
```

IDENTIFICATION

PRODUCT CODE: AC-A829B-MC

PRODUCT NAME: CZDXJB0 DX11-B ADRSNG TST

DATE: APRIL, 1980

MAINTAINER: DIAGNOSTIC PROGRAMMING

AUTHOR: R. MISNER

```
47
48                              1.0     GENERAL PROGRAM INFORMATION
49
50                              1.1     ABSTRACT
51                                      --------
52
53                                      THIS PROGRAM IS A SUPPLEMENT TO DZDXA AND DZDXF DX11-B
54                                      MAINTENANCE MODE DIAGNOSTICS AND IS A FUNCTIONAL CHECK THE
55                                      ABILITY OF THE DX TO ACCESS ALL AREAS OF MEMORY USING DIFFERENT
56                                      SOURCES FOR THE NPR DATA ADDRESS. IT IS ASSUMED THAT DZDXA AND
57                                      DZDXF HAVE ALREADY BEEN RUN. THE THREE NPR ADDRESS SOURCES
58                                      RESULT FROM DATA WRITES TO MEMORY, STATUS POINTER WORD (SPW)
59                                      FETCHES, AND DEVICE STATUS TABLE (DST) FETCHES.
60                                      DIAGNOSIS IS BASED ON THE ASSUMPTION THAT THE CPU CAN ACCESS
61                                      MEMORY CORRECTLY WHILE THE DX'S ADDRESSING CAPABILITY
62                                      IS SUSPECT.
63                                      ERROR MESSAGES DESCRIBE THE FAILING ADDRESS SOURCE AND THE
64                                      EXPECTED LOCATION OF THE DX ACCESS. THE ACTUAL DX ADDRESS
65                                      OF THE ACCESS CANNOT BE DETERMINED.
66                                      THE OPERATOR INTERFACE IS DESIGNED TO BE COMPATIBLE WITH
67                                      DZDXA AND DZDXF.
68
69                              1.2     SYSTEM REQUIREMENTS
70                                      -------------------
71
72                                      THIS PROGRAM REQUIRES ANY PDP11 PROCESSOR WITH AT LEAST
73                                      8K OF MEMORY BUT NOT MORE THAN 128K; THE DX11-B AND A CONSOLE
74                                      TERMINAL. THE PROGRAM WILL RUN UNDER XXDP,ACT,SLIDE, AND
75                                      STAND ALONE. THIS PROGRAM HAS NOT BEEN TESTED ON CPU'S WITHOUT
76                                      A SWITCH REGISTER.
77
78                              1.3     RELATED DOCUMENTS AND STANDARDS
79                                      -------------------------------
80
81                                      PROGRAM IS ASSEMBLED USING SYSMAC MD-11-DZQAC-C3 AND
82                                      CONFORMS TO THE ACT INTERFACE AUTOCAT-11-QZAUB-B-D
83                                                                    --------------------
84
85
86                              1.4     DIAGNOSTIC HIERARCHY PREREQUSITES
87                                      ---------------------------------
88
89                                      IT IS ASSUMED THAT DZDXA,DZDXF, AND APPROPRIATE MEMORY
90                                      DIAGNOSTICS ARE RUN ERROR FREE IN ORDER FOR THE ERROR
91                                      MESSAGES IN THIS DIAGNOSTIC TO BE MEANINGFUL.
92
93
94
95                              1.5     ASSUMPTIONS
96                                      -----------
97
98                                      THIS PROGRAM ASSUMES THAT THE CPU, UNIBUS ADDRESSING,
99                                      AND PORTIONS OF THE DX11-B TESTED BY DZDXA AND DZDXF ARE
100                                     OPERATING CORRECTLY IN ORDER FOR THE ERROR MESSAGES TO BE
101                                     MEANINGFUL.
102
```

```
                2.0     OPERATING INSTRUCTIONS
                        ----------------------

                2.1     LOADING AND STARTING
                        --------------------

                        USE STANDARD LOADING PROCEDURES FOR PDP11 UNDER THE
                        ABSOLUTE LOADER, XXDP, SLIDE, OR ACT. START AT ADDRESS 200.
                        THE PROGRAM WILL PRINT OUT THE MAINDEC NUMBER AND TITLE AND
                        A PROMPTING MESSAGE AS FOLLOWS:
                        ''TYPE:  <D>,FOR DEFAULT PARAMETERS
                                 <P>,FOR PREVIOUS PARAMETERS
                                 <S>,FOR SELECT PARAMETERS
                                 <N>,FOR START WITH THIS TEST NO.

                        D,P,S,N?''

                        THE OPERATOR MUST RESPOND WITH A D OR S IF THIS IS THE
                        FIRST START SINCE THE PROGRAM HAS BEEN LOADED INTO MEMORY. IF,
                        IN RESPONSE TO THIS MESSAGE, THE OPERATOR TYPES AN S<CR> THE
                        FOLLOWING DIAGLOG COULD TAKE PLACE:
                        ''TEST NUMBER:          1<CR>
                         BASE ADDRESS:          176200<CR>
                         VECTOR ADDRESS:        300<CR>
                         DX PRIORITY LEVEL:     4<CR>''

                        NOTE: THE ABOVE RESPONSES ARE THE DEFAULT PARAMETERS
                        AND ARE EQUIVALENT TO TYPING A D<CR> IN RESPONSE TO D,P,S,N?
                        A <CR> RESPONSE FOR A PARTICULAR ENTRY WILL SELECT THE DEFAULT
                        VALUE. A CONTROL C AT ANY TIME WILL TAKE THE PROGRAM BACK TO THE
                        PARAMETER ENTRY SECTION. RESPONSE TO THE CONTROL C MAY BE SLOW
                        IF THE PROGRAM IS IN A PARTICULARLY LONG TEST. ALL PARAMETERS ARE
                        CHECKED TO SEE IF THEY ARE IN A LOGICAL RANGE. IF NOT, THE ENTRY
                        IS REQUESTED AGAIN.


                ''TEST NUMBER:''
                        THE PROGRAM IS ASKING FOR THE STARTING TEST NUMBER. 1
                        THRU 4 ARE VALID RESPONSES. THIS IS THE STARTING TEST
                        NUMBER. TO LOOP ON A PARTICULAR TEST TYPE THE TEST NUMBER IN
                        RESPONSE TO THIS MESSAGE AND SET SWITCH 14 ON THE CONSOLE.


                ''BASE ADDRESS:''

                        THIS IS A REQUEST FOR THE BASE ADDRESS OF THE DX IN THE
                        UNIBUS ADDRESS SPACE. IT IS ALSO THE ADDRESS OF THE DXDS. ALL
                        OTHER DX CONTROL AND STATUS REGISTER ADDRESSES ARE GENERATED
                        FROM THIS ENTRY.


                ''VECTOR ADDRESS:''

                        THIS IS A REQUEST FOR THE DX11-B INTERRUPT VECTOR
                        ADDRESS.
```

```
                        'DX PRIORITY LEVEL''

                                THIS IS A REQUEST FOR THE DX'S INTERRUPT PRIORITY LEVEL.
                                IT IS NOT USED BY THIS PARTICULAR PROGRAM.


                        ''SET SWITCHES:''

                                THIS IS A REMINDER TO SET THE CONSOLE SWITCHES BEFORE
                                STARTING THE PROGRAM. AFTER SETTING SWITCHES, RESPOND WITH A
                                <CR>. CONSOLE SWITCHES HAVE THE FOLLOWING MEANING:

                                SW<15>          HALT ON ERROR
                                SW<14>          LOOP ON ERROR OR TEST
                                SW<13>          INHIBIT ERROR PRINTOUT
                                SW<12>          PRINT ONLY SHORT ERROR REPORT
                                SW<11>          INHIBIT ITERATIONS
                                SW<10>          INHIBIT MEMORY MGT USE

                                TESTING WILL COMMENCE AS SOON AS A <CR> IS GIVEN
                                IN RESPONSE TO THE ''SET SWITCHES:'' MESSAGE.

                                THIS PROGRAM AUTO SIZES THE MEMORY. THE CONTROL UNIT
                                ADDRESS AND DEVICES PER CONTROL UNIT ARE ALSO AUTO SIZED.
                                THIS PARTICULAR PROGRAM DOES NOT NEED THE 'LEGAL COMMANDS''
                                FACILITY USED BY OTHER DX DIAGNOSTICS.


        -               2.2     SPECIAL ENVIRONMENTS
                                ---------------------

                                THIS PROGRAM WILL RUN IN UNATTENDED MODE UNDER ACT.


                        2.3     PROGRAM OPTIONS
                                ---------------

                                SEE SECTION 2.1 FOR PARAMETER SELECTION AND OPTIONS.
                                NOTE THAT ALL CONSOLE SWITCH CHANGES ARE EFFECTIVE IMMEDIATELY
                                EXCEPT SW<10> ''INHIBIT MEMORY MANAGEMENT USES''. IN ORDER TO
                                CHANGE THE MEANING OF SW<10> SET THE SWITCH THEN RESTART THE
                                PROGRAM VIA CONTROL C OR START AT ADDRESS 200.


                        2.4     EXECUTION TIMES
                                ---------------

                                SINGLE PASS EXECUTION TIME IS VARIABLE DEPENDING ON
                                MEMORY SIZE. A PDP-11/45 WITH 65K WORDS OF MEMORY REQUIRES
                                50 SECONDS. EACH ROUTINE HAS A SINGLE ITERATION REGARDLESS
                                OF MODE. END PASS IS SIGNALED AFTER ALL TESTS HAVE BEEN RUN
                                ONCE.
```

3.0 ERROR INFORMATION

THE GENERAL ERROR REPORT FORMAT IS AS FOLLOWS:

LINE 1

ERROR PC: XXXXXX ERROR NO. YYYYY

THIS GIVES THE PROGRAM COUNTER (XXXXXX) WHERE THE ERROR
WAS DETECTED AND A UNIQUE ERROR NUMBER OR IDENTIFIER (YYYYYY)

LINE 2

THIS LINE GIVES A UNIQUE ERROR MESSAGE FOR EACH ERROR
TYPE EXPLAINING THE SUSPECTED CAUSE OF THE FAILURE. THIS IS
THE LAST LINE OF THE MESSAGE IF SWITCH 12 IS ON.

LINE 3

TEST NO. XXXXXX

THIS IDENTIFIES THE TEST NUMBER (XXXXXX - 1-4) WHERE THE
FAILURE WAS DETECTED.

LINE 4

VIRT ADDR OF BUFFER BASE: XXXXXX

THIS GIVES THE VIRTUAL ADDRESS OF THE LOWEST LOCATION
IN THE BUFFER WHICH THE DX WAS ACCESSING AT THE TIME OF ERROR.

LINE 5

PHY ADDR OF BUFFER: XXXXXX

THIS GIVES THE PHYSICAL ADDRESS OF THE BUFFER BASE. ONLY
OUTPUT IF MEMORY MANAGEMENT IS IN USE

LINE 6

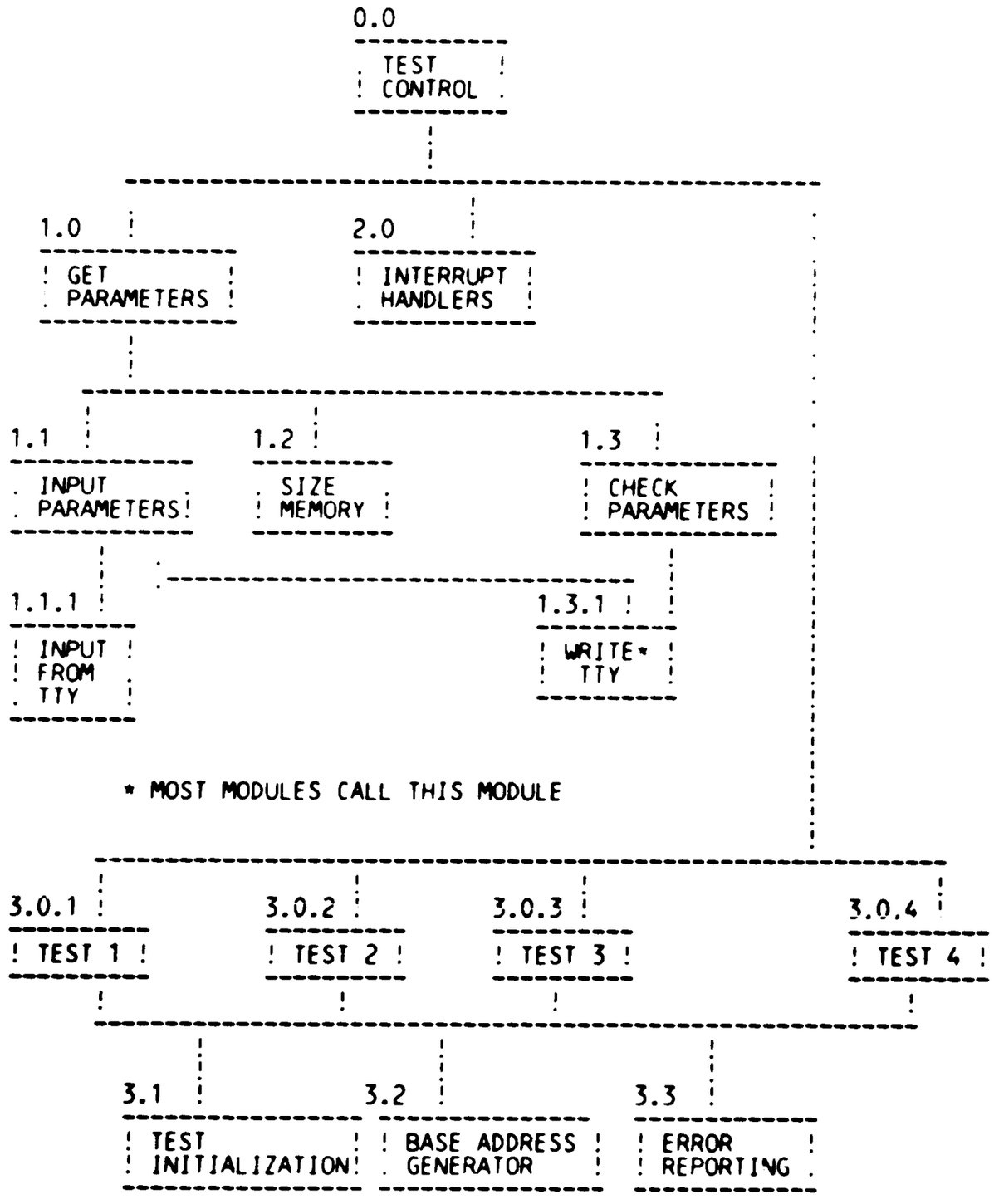
EA BITS IN OCT: XXXXXX

THE LOW TO BITS OF XXXXXX ARE THE EXTENDED ADDRESS BITS
USED IN ACCESSING THE FAILING MEMORY LOCATION. ONLY OUTPUT IF
MEMORY MANAGEMENT IN USE.

LINE 7

```
                                VIRT ADDR OF ACCESS ERROR: XXXXXX

                                THIS MESSAGE IS ONLY OUTPUT FOR TEST 2. IT GIVES THE
                                EXPECTED WORD LOCATION OF THE FAILING ACCESS.

                                LINE 7-8 ALTERNATE
                                ACTUAL: XXXXX EXPECTED: XXXXXX

                                IN SOME TESTS, THIS GIVES THE ACTUAL AND EXPECTED BYTE
                                OF DATA. THIS WILL GIVE INSIGHT INTO FAILING ADDRESSES AS THERE
                                IS A CORRESPONDENCE BETWEEN THIS DATA AND THE LOCATION OF THE
                                FAILURE.

                        NOTE:   AT THE END OF THE FIRST PASS EXECUTION THE FOLLOWING
                                MESSAGE IS PRINTED:

                                1ST IBM DEV RCGNZD. OCTAL: XXXXXX
                                SIZE OF 1ST DEV GROUP: YYYYYY

                                THIS IS NOT AN ERROR MESSAGE BUT IS AN INDICATION OF
                                HOW THE JUMPERS ARE SET UP ON MODULE A20. THIS INFOR-
                                MATION IS PROVIDED ONLY AS A CONVENIENCE FOR THE
                                OPERATOR. SEE MAINTENANCE MANUAL EK-DX11B-MM-002
                                PARA. 2.4.2 FOR MORE INFORMATION.

                        4.0     PROGRAM ORGANIZATION
```

4.1 BLOCK DIAGRAM



SEQ 0008

4.2 TEST DESCRIPTIONS

4.2.1 TST 1

```
                                        MODULE 3.0.1 FAST NPR DATA TEST

                                        THIS IS A FUNCTIONAL TEST TO CHECK THE ABILITY OF THE DX TO DO NPR'S
                                        OF DATA INTO MEMORY WITH LARGE BYTE COUNTS. THE DXBA IS CHECKED
                                        FOR ITS ABILITY TO COUNT AND ALL CARRIES IN THE COUNTER ARE
                                        EXERCISED. THE DATA TRANSFERS ACCESS ALL AVAILABLE MEMORY EXCEPT
                                        WHERE THE PROGRAM IS LOCATED, AND THE AREA WHICH MAY BE OCCUPIED BY
                                        A LOADER.

                                4.2.2 TST 2

                                        MODULE 3.0.2 NPR DATA ADDR UNIQUENESS TEST

                                        THIS MODULE TESTS THE DX11'S ABILITY TO UNIQUELY ADDRESS EACH
                                        WORD OF MEMORY. THIS TEST IS ACCOMPLISHED BY WRITING A BACKGROUND
                                        IN A BLOCK OF MEMORY. A WORD IS THEN TRANSFERRED FROM THE DX
                                        TO THE MEMORY BLOCK. THE CPU CHECKS THAT THE WORD ARRIVES AT
                                        ITS EXPECTED DESTINATION. IF YES THE WORD IS RESTORED TO MATCH
                                        THE BACKGROUND, AND THE NEXT WORD IS TESTED. THIS IS REPEATED
                                        FOR EACH WORD IN THE BLOCK AND FOR EACH BLOCK IN MEMORY.

                                NOTE:   THIS TEST MAY NOT DETECT ADDRESSING PROBLEMS
                                        WHICH ARE COMMON TO THE CPU AND DX11. I.E. UNIBUS ADDRESSING
                                        PROBLEMS.

                                4.2.3 TST 3

                                        MODULE 3.0.3 SPW ADDRESSING TEST

                                        THIS MODULE TESTS THE ABILITY OF THE DX TO ACCESS THE STATUS
                                        POINTER WORD (SPW) TABLE IN ALL POSSIBLE LOCATIONS IN MEMORY.
                                        A BASE ADDRESS FOR THE SPW IS GENERATED.
                                        THIS SPW TABLE IS FILLED WITH A PATTERN OF 000DEV
                                        WHERE DEV IS EQUAL TO THE DISPLACEMENT OF THE WORD FROM THE
                                        SPW BASE; I.E.
                                        SPW BASE     =000000
                                        SPW BASE +1 =000001
                                                    ..
                                                    ..
                                                    ..
                                                    ..
                                        SPW BASE+255=000377

                                        GIVING A TOTAL OF 256 WORDS FOR THE SPW TABLE WITH IMMEDIATE
                                        STATUS EQUAL TO THE DEVICE NUMBER.
                                        AN ISS SEQUENCE IS INITIATED WITH A DEVICE # OF 0. IF
                                        ADRECC AND ADRECD DON'T COME ON FOR THIS DEV, NO TESTING IS DONE.
                                        THE DEV # IS INCREMENTED, AND ISS IS DONE AGAIN.
                                        IF ADRECC AND ADRECD COME ON IT MEANS THE DX JUMPERS ARE CUT TO
                                        RESPOND TO THIS DEV NUMBER. THE ISS SEQUENCE IS THEN CONTINUED
                                        UNITL THE STATUS IS IN THE DXOS REG. THIS STATUS IS THEN
                                        CHECKED AND SHOULD BE EQUAL TO THE DEV # IF THE SPW WAS
                                        ACCESSED CORRECTLY.

                                        THIS PROCEDURE IS REPEATED FOR ALL 256 POSSIBLE DEVICES AT
                                        ALL POSSIBLE LOCATIONS FOR THE SPW.
```

```
                                THE FIRST RECOGNIZED DEV NO. AND COUNT OF THE FIRST
                                GROUP OF RECOGNIZED DEVICES IS SAVED AND PRINTED AT END
                                PASS TIME. IF NO DEV IS RECOGNIZED AND ERROR MESSAGE
                                IS OUTPUT AT THE END OF THE TEST.

                        4.2.4 TST 4

                                MODULE 3.0.4 DEVICE STATUS TABLE (DST) ACCESS TEST

                                THIS TEST CHECKS THE ABILITY TO ACCESS ALL BYTES OF THE
                                DST IN ALL AREAS OF MEMORY. THE OBJECTIVE IS TO CHECK LOADINC
                                OF THE DXBA FROM THE SPW DST BASE, AND  COMMAND (CUCR)

                                A VALID IBM DEVICE FROM TST3 IS CHECKED. THEN A BASE ADDR
                                IS GENERATED FOR THE DST. EACH BYTE OF THE DST IS THEN LOADED
                                WITH ITS OFFSET FROM THE DST BASE.

                                A SPW ENTRY FOR THE VALID IBM DEVICE IS GENERATED WITH AP-
                                PROPRIATE DST BASE AND ZERO IMMEDIATE STATUS TO ASSURE A DST
                                ACCESS. AN ISS SEQUENCE IS DONE WITH THE VALID DEV AND
                                EVERY POSSIBLE IBM COMMAND THEREBY ACCESSING EVERY LOCATION
                                IN THE DST. AFTER THE ISS, THE STATUS WILL EQUAL THE COMMAND
                                OR AND ERROR IS FLAGGED.
                                %
```

```
                                        .NLIST  CND,MC,TOC
                                        .LIST   ME
                                        .ENABL  ABS,AMA
                                .TITLE  CZDXJB0  DX11-B ADDRESSING TEST
                                ;*COPYRIGHT (C) -1977-
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY R. MISNER
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
                                .SBTTL  BASIC DEFINITIONS

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

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

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

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

```
                                ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
          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
                                .SBTTL  MEMORY MANAGEMENT DEFINITIONS

                                ;*KT11 VECTOR ADDRESS

000250                          MMVEC=  250

                                ;*KT11 STATUS REGISTER ADDRESSES

177572                          SR0=    177572
177574                          SR1=    177574
177576                          SR2=    177576
172516                          SR3=    172516

                                ;*KERNEL ''I'' PAGE DESCRIPTOR REGISTERS

172300                          KIPDR0= 172300
172302                          KIPDR1= 172302
172304                          KIPDR2= 172304
172306                          KIPDR3= 172306
172310                          KIPDR4= 172310
172312                          KIPDR5= 172312
172314                          KIPDR6= 172314
172316                          KIPDR7= 172316

                                ;*KERNEL ''I'' PAGE ADDRESS REGISTERS

172340                          KIPAR0= 172340
172342                          KIPAR1= 172342
172344                          KIPAR2= 172344
172346                          KIPAR3= 172346
172350                          KIPAR4= 172350
172352                          KIPAR5= 172352
172354                          KIPAR6= 172354
172356                          KIPAR7= 172356

000200                          DONE=200
000004                          SOSIEN=4
020000                          SEL0=20000
040000                          HLD0=40000
002000                          CMD0=2000
001000                          SRV0=1000
004000                          ADR0=4000
```

```
                                       .SBTTL  TRAP CATCHER

                000000                         .=0
                                       ;*ALL UNUSED LOCATIONS FPOM 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  001164                         JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
                                       .SBTTL  ACT11 HOOKS

                                       ;;**********************************************************************
                                       ;HOOKS REQUIRED BY ACT11
                000204                         $SVPC=.                  ;SAVE PC
                000046                         .=46
000046  001566                                 $ENDAD                   ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                000052                         .=52
000052  040000                                 .WORD   40000            ;;2)SET LOC.52 TO 40000
                000204                         .=$SVPC                  ;; RESTORE PC
                001000                         .=1000
```

```
                                       .SBTTL  COMMON TAGS

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

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

```
                                .SBTTL  ERROR POINTER TABLE

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

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


001164                          $ERRTB:
```

```
                                                .SBTTL  MODULE 0.0      TEST CONTROL
                                        ;       MODULE 0.0      TEST CONTROL
                                        ;
                                        ;
                                        ;
                                        ;
                                        ;
001164                                  START:
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001164  012706  001100                          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001170  005026                          10$:    CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001172  022706  001140                          CMP     #SWR,R6 ;;DONE?
001176  001374                                  BNE     .-6             ;;LOOP BACK IF NO
001200  012706  001100                          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
001204  012737  005654  000034                  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001212  012737  000340  000036                  MOV     #340,@#TRAPVEC+2;LEVEL 7
001220  012737  005752  000024                  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001226  012737  000340  000026                  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A ''-1'', SETUP FOR A SOFTWARE SWITCH REGISTER.
001234  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001240  012737  001274  000004                  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001246  012737  177570  001140                  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001254  012737  177570  001142                  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001262  022777  177777  177650                  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001270  001012                                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
001272  000403                                  BR      65$             ;;BRANCH IF NO TIMEOUT
001274  012716  001302                  64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001300  000002                                  RTI
001302  012737  000176  001140          65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001310  012737  000174  001142                  MOV     #DISPREG,DISPLAY
001316  012637  000004                  66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001322  000005                                  RESET
001324  005037  011654                          CLR     FIRSTP          ;CLEAR 1ST PASS SW
001330  005037  011754                          CLR     DEVMSS          ;SET UP TO PRINT DEVICE ADDR CONFIG FIRST TIME ONLY
001334  004737  001600          RESTART:JSR     PC,GETPRM       ;TALK TO THE OPERATOR
001340  013737  011672  011674                  MOV     COUNT,CNT       ;SET UP ITERATION COUNT
001346  013737  011634  011666  GO:     MOV     ATESTN,CTESTN   ;GET FIRST TEST NO.
001354  005037  011700                          CLR     LOCK            ;CLEAR LOOP LOCK SW
001360  005037  011676                          CLR     ERR             ;RESET ERROR INDICATOR
001364  013702  011666          TST:    MOV     CTESTN,R2
001370  006302                          ASL     R2              ;MULTIPLY BY TWO FOR WORD BNDRY
001372  000172  001376                  JMP     @TSTAB(R2)      ;GO TO TEST THRU DISPATCH TABLE
001376  000000                  TSTAB:  HALT                    ;TEST 0 DOESN'T EXIST
001400  006134                          TST1
001402  006656                          TST2
001404  007124                          TST3
001406  007636                          TST4
                                        ;WHEN TEST IS DONE IT ALWAYS RETURNS HERE AFTER ONE ITERATION
001410                                  ENDTST:
001410  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
001412  012746  001420                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
```

```
                                        RTI                     ;;POP NEW PC AND PS
                                64$:
                                
001420  032777  040000  177512          BIT     #40000,@SWR     ;IS LOOP SWITCH ON
001426  001356                          BNE     TST             ;BR IF YES
001430  005237  011666                  INC     CTESTN          ;INCREMENT TO NEXT TEST
001434  023737  011666  011616          CMP     CTESTN,HTESTN   ;HAVE ALL TEST BEEN RUN
001442  003750                          BLE     TST             ;BR IF NO, DISPATCH TO NEXT ONE
001444  005337  011674                  DEC     CNT             ;DECREMENT ITERATION COUNT
001450  001336                          BNE     GO              ;START OVER IF NOT EXPIRED
001452  005737  011754          EOP:    TST     DEVMSS          ;HAVE WE PRINTED DEVICE CONFIG YET?
001456  001026                          BNE     10$             ;BR IF YES
001460  005737  011752                  TST     DEVCT           ;WERE DEVICES RECOGNIZED?
001464  001423                          BEQ     10$             ;BR IF NO, NOTHING TO SAY
001466  004537  005134                  JSR     R5,PRINT        ;PRINT DEV NO. ASCII
001472  013161                          .WORD   DEVMS
001474  000000                          .WORD   0
001476  004537  005134                  JSR     R5,PRINT        ;PRINT DEV NO. OCTAL
001502  011750                          .WORD   FRSTDV
001504  100001                          .WORD   100001
001506  004537  005134                  JSR     R5,PRINT        ;PRINT DEVICE COUNT ASCII
001512  013217                          .WORD   DEVCMS
001514  000000                          .WORD   0
001516  004537  005134                  JSR     R5,PRINT        ;PRINT DEVICE COUNT OCT
001522  011752                          .WORD   DEVCT
001524  100001                          .WORD   100001
001526  012737  000001  011754          MOV     #1,DEVMSS       ;SET DEV CONFIG PRINTED SWITCH
001534  005737  011652          10$:    TST     KT              ;IS KT IN USE?
001540  001402                          BEQ     15$             ;BR IF NO
001542  005037  177572                  CLR     @#SR0           ;TURN OFF MEM MGT
001546  004537  005134          15$:    JSR     R5,PRINT
001552  012711                          .WORD   BELL
001554  000000                          .WORD   0               ;PRINT CONTROL PARAMETER
001556  013700  000042                  MOV     @#42,R0         ;ARE WE UNDER ACT
001562  001405                          BEQ     END1            ;BR IF NO, START OVER
001564  000005                          RESET
001566  004710                  $ENDAD: JSR     PC,@R0          ;GO TO MONITOR
001570  000240                          NOP                     ;ROOM FOR
001572  000240                          NOP                     ;    ACT11
001574  000240                          NOP                     ;    STUFF
001576  000656                  END1:   BR      RESTART         ;START OVER
```

```
                                                .SBTTL  MODULE 1.0  GET PARAMETERS
                                        ;       MODULE  1.0     GET PARAMETERS
                                        ;
                                        ;       CALLED AS FOLLOWS
                                        ;
                                        ;       JSR     PC,GETPRM
                                        ;
                                        ;       THIS MODULE ASSURES THAT ALL PARAMETERS ARE AQUIRED FOR
                                        ;       OPERATION OF THE DIAGNOSTIC.  TO DO THIS IT ESTABLISHES
                                        ;       A DIALOG WITH THE OPERATOR THROUGH THE INPUT PARAM MODULE.
                                        ;       MEMORY IS SIZED BY THE 'SIZE' MODULE, AND ALL PARMS ARE
                                        ;       CHECKED FOR VALIDITY BY THE CHECKP MODULE.
                                        ;
                                        ;       ^C AT ANY TIME CLEARS THE 1ST PASS SWITCH AND EVERYTHING
                                        ;       STARTS OVER.  CR IN RESPONSE TO ANY QUESTION CAUSES THE DEFAULT
                                        ;       VALUE TO BE SUBSTITUTED.
                                        ;
                                        ;       THIS MODULE NEVER RETURNS TO CALLER UNLESS ALL PARAMETERS
                                        ;       HAVE BEEN ESTABLISHED WITH GOOD VALUES.
                                        ;
001600  005737  011654        GETPRM: TST     FIRSTP          ;IS THIS THE FIRST PASS
001604  001402                        BEQ     5$              ;BR IF YES
001606  000137  002270                JMP     ENDPRM          ;DON'T GET PARM IF WE ALREADY HAVE THEM
001612  005777  176224        5$:     TST     @42             ;ARE WE RUNNING IN AUTO MODE
001616  001415                        BEQ     10$             ;BRANCH IF MACHINE IS ATTENDED
001620  013737  011604  011634        MOV     DTESTN,ATESTN   ;SETUP DEFAULTS
001626  013737  011606  011636        MOV     DUBA,AUBA       ;SETUP DEFAULTS
001634  013737  011610  011640        MOV     DVECT,AVECT     ;SETUP DEFAULTS
001642  013737  011612  011642        MOV     DPRIOR,APRIOR   ;SETUP DEFAULTS
001650  000564                        BR      90$             ;GO SEE ABOUT MEMORY SIZING
001652  022737  001566  000042  10$:  CMP     #$ENDAD,@#42    ;UNDER ACT11?
001660  001560                        BEQ     90$             ;NO OPERATOR INTERACTION
001662  005737  011670                TST     PARMV           ;ARE PARAMETERS GOOD
001666  001004                        BNE     15$             ;BRANCH IF YES BECAUSE THIS
                                                              ;ALSO MEANS HEADER ISN'T NEEDED
001670  004537  005134                JSR     R5,PRINT        ;CALL THE PRINTER
001674  012266                        HEADER                  ;ADDRESS OF HEADER ASCIZ
001676  000000                        .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
001700  004537  002272        15$:    JSR     R5,INPRM        ;GET D,P,S, OR N FROM OPER.
001704  000001                        .WORD   1               ;THIS TELLS INPRM TO GET D,P,S,N
                                        ;INPRM WILL RETURN IN R0 THE CHAR TYPED
001706  122700  000104                CMPB    #'D,R0          ;WAS DEFAULT PARMS REQUESTED?
001712  001015                        BNE     20$             ;BR IF NO
001714  013737  011604  011634        MOV     DTESTN,ATESTN   ;DEFAULT SETUP
001722  013737  011606  011636        MOV     DUBA,AUBA       ;DEFAULT SETUP
001730  013737  011610  011640        MOV     DVECT,AVECT     ;DEFAULT SETUP
001736  013737  011612  011642        MOV     DPRIOR,APRIOR   ;DEFAULT SETUP
001744  000526                        BR      90$             ;GO SEE ABOUT MEMORY SIZING
001746  122700  000120        20$:    CMPB    #'P,R0          ;DOES OPERATOR WANT PREVIOUS
001752  001011                        BNE     30$             ;BR IF NO
001754  005737  011670                TST     PARMV           ;ARE PARMS SET UP
001760  001401                        BEQ     25$             ;BR IF PARAMETERS NOT VALID
001762  000517                        BR      90$             ;USE PREVIOUS PARMS, NO ACTION NEEDED
001764  004537  005134        25$:    JSR     R5,PRINT        ;TROUBLE, P TYPED AND PARMS
001770  012517                        .WORD   INVM            ;NEVER SET UP PROPERLY
```

```
                                                .WORD   0               ;CONTROL WORD,CRLF + DONT CONVERT
                                                BR      10$             ;GO TO TRY AGAIN
                                30$:    CMPB    #'S,R0          ;ARE PARMS TO BE SELECTED?
                                                BNE     40$             ;BR IF NO
                                35$:    CLR     PARMV           ;CLEAR PARM VALID SW
                                                JSR     R5,INPRM        ;ASK INPUT MODULE TO GET TEST NO:
                                                .WORD   2               ;TEST NUMBER WILL BE RETURNED IN R0
                                                                        ;IN OCTAL
                                                MOV     R0,ATESTN       ;SAVE TEST NUMBER IN ACTIVE PARM TABLE
                                                JSR     R5,CHECKP       ;CHECK IF TEST NO OK
                                                .WORD   2               ;PARAMETER INDICATES VERIFY TEST #
                                                TST     R0              ;R0=0 IF CHECK OK
                                                BNE     35$             ;STAY HERE UNTIL OPERATOR GETS IT RIGHT
                                36$:    JSR     R5,INPRM        ;ASK INPUT MODULE TO GET BASE ADDR
                                                .WORD   3               ;PARAMETER INDICATING GET UBA BASE
                                                MOV     R0,AUBA         ;SAVE UB BASE ADDR IN ACTIVE TABLE
                                                JSR     R5,CHECKP       ;CHECK FOR VALID ENTRY
                                                .WORD   3               ;PARM TO INDICATE CHECK UBA
                                                TST     R0              ;R0=0 IF CHECK OK
                                                BNE     36$             ;STAY HERE UNTIL HE GETS IT RIGHT
                                37$:    JSR     R5,INPRM        ;ASK INPUT MODULE TO GET VECTOR ADDR
                                                .WORD   4               ;PARM INDICATING VECTOR NEEDED
                                                MOV     R0,AVECT        ;SAVE VECTOR IN ACTIVE PARM TABLE
                                                JSR     R5,CHECKP       ;CHECK FOR VALID VECTOR
                                                .WORD   4               ;PARM INDICATING CHECK VECTOR
                                                TST     R0              ;R0=0 IF VECTOR OK
                                                BNE     37$             ;STAY HERE UNTIL HE GETS IT RIGHT
                                38$:    JSR     R5,INPRM        ;GO GET DX PRIORITY
                                                .WORD   5
                                                MOV     R0,APRIOR       ;SAVE PRIORITY
                                                JSR     R5,CHECKP       ;CHECK PRIORITY ENTRY
                                                .WORD   5               ;PARM INDICATING PRIORITY CHECK
                                                TST     R0
                                                BNE     38$             ;STAY HERE UNTIL HE GETS IT RIGHT
                                                JSR     R5,PRINT        ;PRINT SET SWITCHES MESSAGE
                                                .WORD   SETSW           ;ADDR OF MESSAGE
                                                .WORD   0
                                                JSR     R5,INASC        ;DUMMY INPUT REQUEST TO WAIT FOR
                                                .WORD   1               ;OPERATOR TO SET SWITCHES
                                                BR      90$             ;GO SEE ABOUT MEMORY SIZING
                                40$:    CMPB    #'N,R0          ;WAS AN 'N' INPUT
                                                BNE     10$             ;BR IF NONE OF DPSN WERE INPUT
                                45$:    JSR     R5,INPRM        ;CALL FOR TEST NUMBER TO BE INPUT
                                                .WORD   2               ;PARM SAYS GET TEST #.
                                                MOV     R0,ATESTN       ;SAVE STARTING TEST #
                                                JSR     R5,CHECKP       ;HAVE TEST # CHECKED FOR PROPER
                                                .WORD   2               ;RANGE
                                                TST     R0              ;IS TEST # OK?
                                                BNE     45$             ;STAY HERE UNTIL HE GETS IT RIGHT
                                                TST     PARMV           ;WE HAVE STARTING TEST #, ARE OTHER PARMS OK?
                                                BNE     90$             ;BR IF OK
                                                JSR     R5,PRINT        ;PRINT
                                                INVPRM                  ;POINTER TO 'NEED MORE PARMS'' MSG
                                                .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
                                                BR      10$             ;GO BACK AND DO IT ALL AGAIN
                                ;
```

```
    897                                 ;
    898  002222  004737  004212         90$:    JSR     PC,SIZE         ;CALL MEMORY SIZING MODULE
    899  002226  012700  000015                 MOV     #15,R0  ;START OF LOOP TO SET UP DX ACCESS ADDRESSES
    900  002232  012701  011534                 MOV     #DXDSA,R1       ;GET ADDR OF START OF TABLE
    901  002236  013702  011636                 MOV     AUBA,R2         ;GET DX REG BASE ADDRESS
    902  002242  010221                 95$:    MOV     R2,(R1)+        ;MOVE UBA TO TABLE
    903  002244  062702  000002                 ADD     #2,R2           ;STEP TO NEXT UBA
    904  002250  005300                         DEC     R0              ;DECREMENT  COUNTER
    905  002252  001373                         BNE     95$             ;FILL THE 13 LOCATIONS OF THE TABLE
    906  002254  012737  000001  011670         MOV     #1,PARMV        ;SET PARAMETER VALID SWITCH
    907  002262  012737  000001  011654         MOV     #1,FIRSTP       ;SET NOT FIRST PASS SWITCH
    908  002270  000207                 ENDPRM: RTS     PC              ;RETURN TO CALLER, END OF MODULE
```

```
                                        .SBTTL  MODULE 1.1  INPUT PARAMETERS

                                ;       MODULE 1.1      INPUT PARAMETERS
                                ;
                                ;       CALLED AS FOLLOWS
                                ;
                                ;       JSR     R5,INPRM
                                ;       .WORD   X
                                ;
                                ;               WHERE X DEFINES WHAT PARAMETER TO GET AS
                                ;               FOLLOWS:
                                ;               X=1     D,P,S,ORN  PARAMETER SELECTION
                                ;               X=2     STARTING TEST #
                                ;               X=3     UBA BASE ADDR
                                ;               X=4     DX VECTOR ADDR
                                ;               X=5     DX PRIORITY
                                ;
                                ;       THE REQUESTED PARAMETER IS RETURNED IN R0 ALL CONVERTED
                                ;       TO OCTAL AND DEFAULT SUBSTITUTED WHEN INDICATED.  INPUTS
                                ;       ARE CHECKED FOR EVERYTHING EXCEPT THAT THE NUMBERS ARE
                                ;       IN THE CORRECT RANGE.  CHECKING INCLUDES NUMERIC IF
                                ;       REQUIRED, OCTAL IF REQUIRED, <CR> FOR DEFAULT, ETC.
                                ;       THIS MODULE DESTROYS R1,R0
002272  012501                  INPRM:  MOV     (R5)+,R1        ;GET THE PARAMETER FROM THE
                                                                ;CALLING CODE.
002274  120127  000001                  CMPB    R1,#1           ;IS PARM 1 - GET D,P,S, ORN?
002300  001016                          BNE     10$             ;BR IF NO.
002302  004537  005134                  JSR     R5,PRINT        ;PRINT
002306  012555                          .WORD   PROMPT          ;D,P,S,N?
002310  000000                          .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
002312  004537  002532                  JSR     R5,INASC        ;READ THE INPUT -
002316  000001                          .WORD   1               ;INASC RETURNS POINTER TO
002320  111100                          MOVB    (R1),R0         ;INPUT IN R1
                                                                ;1ST CHAR TO R0
002322  120027  000015                  CMPB    R0,#CR          ;WAS IT DEFAULTLED?
002326  001100                          BNE     90$             ;NO, ALL DONE
002330  113700  000120                  MOVB    'P,R0           ;INSERT DEFAULT OF P=PREVIOUS PARMS.
002334  000475                          BR      90$             ;ALL DONE

002336  120127  000002          10$:    CMPB    R1,#2           ;IS PARM 2 - GET TEST #
002342  001014                          BNE     20$             ;BR IF NO
002344  004537  005134                  JSR     R5,PRINT        ;PRINT
002350  012567                          .WORD   TESTN           ;ADDR OF STARTING TESTN MSG.
002352  000000                          .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
002354  004537  002532                  JSR     R5,INASC        ;GET OCTAL INPUT
002360  000002                          .WORD   2               ;PARM TO GET OCTAL INPUT
002362  011100                          MOV     (R1),R0         ;GET INPUT #
002364  001061                          BNE     90$             ;IF NON ZERO, WE'RE DONE
002366  013700  011604                  MOV     DTESTN,R0       ;SUBSTITUTE THE DEFAULT
002372  000456                          BR      90$             ;DONE
002374  120127  000003          20$:    CMPB    R1,#3           ;WAS UBA REQUESTED
002400  001014                          BNE     30$             ;BR IF NO
002402  004537  005134                  JSR     R5,PRINT        ;PRINT
002406  012612                          .WORD   BASEA           ;ASK FOR BASE ADDR
002410  000000                          .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
```

```
 965  002412  004537  002532                JSR     R5,INASC        ;GET THE OCTAL
 966  002416  000002                        .WORD   2               ;INPUT
 967  002420  011100                        MOV     (R1),R0         ;GET THE NUMBER INPUT
 968  002422  001042                        BNE     90$             ;BR IF NOT DEFAULTED
 969  002424  013700  011606                MOV     DUBA,R0         ;SUBSTITUTE THE DEFAULT
 970  002430  000437                        BR      90$             ;DONE
 971  002432  120127  000004        30$:    CMPB    R1,#4           ;DOES CALLER WANT VECTOR?
 972  002436  001014                        BNE     40$             ;BR IF NO
 973  002440  004537  005134                JSR     R5,PRINT        ;PRINT
 974  002444  012630                        .WORD   VECTA           ;ASK FOR VECTOR ADDR
 975  002446  000000                        .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
 976  002450  004537  002532                JSR     R5,INASC        ;GET THE INPUT
 977  002454  000002                        .WORD   2               ;IN OCTAL
 978  002456  011100                        MOV     (R1),R0         ;GET THE NUMBER INPUT
 979  002460  001023                        BNE     90$             ;BR IF NOT DEFAULTED
 980  002462  013700  011610                MOV     DVECT,R0        ;SUBSTITUTE THE DEFAULT
 981  002466  000420                        BR      90$             ;DONE
 982  002470  120127  000005        40$:    CMPB    R1,#5           ;DOES CALLER WANT PRIORITY?
 983  002474  001014                        BNE     80$             ;BR IF INVALID CALL
 984  002476  004537  005134                JSR     R5,PRINT        ;PRINT
 985  002502  012650                        .WORD   PRILV           ;ASK FOR PRIORITY LEVEL
 986  002504  000000                        .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
 987  002506  004537  002532                JSR     R5,INASC        ;REQUEST INPUT
 988  002512  000002                        .WORD   2               ;IN OCTAL
 989  002514  011100                        MOV     (R1),R0         ;GET THE INPUT
 990  002516  001004                        BNE     90$             ;BR IF OK,  NOT DEFAULTED
 991  002520  013700  011612                MOV     DPRIOR,R0       ;SUBSTITUTE THE DEFAULT
 992  002524  000401                        BR      90$
 993                                        .
 994  002526  000000        80$:    HALT                            ;HALT ON INVALID CALL TO THIS MOD
 995  002530  000205        90$:    RTS     R5                      ;RETURN TO CALLER
 996                                .SBTTL  MODULE 1.1.1  INPUT FROM TTY
 997
```

```
                                ;       MODULE 1.1.1    INPUT FROM TTY
                                ;
                                ;       CALLED AS FOLLOWS
                                ;
                                ;       JSR     R5,INASC
                                ;       .WORD   X
                                ;
                                ;       WHERE   X=1     FOR ASCII INPUT
                                ;               X=2     FOR OCTAL INPUT
                                ;
                                ;       A POINTER TO THE INPUT MESSAGE IS RETURNED IN R1.  THIS
                                ;       ROUTINE DESTROYS R0,R1
                                ;
002532  004737  002620          INASC:  JSR     PC,$TKINT       ;INITIALIZE THE TTY INPUT
002536  012500                          MOV     (R5)+,R0        ;GET THE PASSED PARAMETER
002540  120027  000001                  CMPB    R0,#1           ;IS IT A REQUEST FOR ASCII?
002544  001004                          BNE     10$             ;BR IF NO.
002546  104410                          RDLIN                   ;TRAP TO .$READ MACRO
002550  013637  013304                  MOV     @(SP)+,INBUF    ;GET THE INPUT DATA
002554  000406                          BR      20$             ;GO GET RESULT
                                ;
002556  120027  000002          10$:    CMPB    R0,#2           ;IS IT A REQUEST FOR OCTAL
002562  001006                          BNE     30$             ;BR IF NO
002564  104411                          RDOCT
002566  012637  013304                  MOV     (SP)+,INBUF     ;;POP STACK INTO INBUF
002572  012701  013304          20$:    MOV     #INBUF,R1       ;SET POINTER TO INPUT
002576  000205                          RTS     R5              ;RETURN TO CALLER
002600  000000                  30$:    HALT                    ;INVALID CALL TO MODULE
                                ;
                                ;
                                .SBTTL  TTY INPUT ROUTINE
                                ;;*******************************************************************
                                .ENABL  LSB
002602  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
002604  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
002606  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
002610  000010                  $TKQSRT: .BLKB  10              ;;TTY KEYBOARD QUEUE
        002620                  $TKQEND=.
                                ;*TK INITIALIZE ROUTINE
                                ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
                                ;
                                ;*CALL:
                                ;*      JSR     PC,$TKINT
                                ;*      RETURN
                                ;
002620  005037  002602          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
002624  012737  002610  002604          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
002632  013737  002604  002606          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
002640  012737  002670  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
002646  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;'BR' LEVEL 4
002654  005777  176266                  TST     @$TKB           ;;CLEAR DONE FLAG
002660  012777  000100  176256          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
```

```
 1054  002666  000207                          RTS     PC              ;;RETURN TO CALLER
 1055
 1056                                  ;*TK SERVICE ROUTINE
 1057                                  ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
 1058                                  ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
 1059                                  ;*IT IN THE QUEUE.
 1060                                  ;*IF THE CHARACTER IS A ''CONTROL-C'' (^C) $TKINT IS CALLED AND
 1061                                  ;*UPON RETURN EXIT IS MADE TO THE ''CONTROL-C'' RESTART ADDRESS (START)
 1062                                  ;
 1063  002670  117746  176252          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
 1064  002674  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
 1065  002700  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
 1066  002704  001007                          BNE     1$              ;;BRANCH IF NO
 1067  002706  104401  004010                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
 1068  002712  004737  002620                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
 1069  002716  005726                          TST     (SP)+           ;;CLEAN UP STACK
 1070  002720  000137  001164                  JMP     START           ;;CONTROL C RESTART
 1071  002724  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
 1072  002730  001004                          BNE     2$              ;;BRANCH IF NO
 1073  002732  022737  000176  001140          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
 1074  002740  001500                          BEQ     6$              ;;GO TO SWR CHANGE
 1075
 1076  002742                          2$:
 1077  002742  022737  000010  002602          CMP     #10,$TKCNT      ;;IS THE QUEUE FULL?
 1078  002750  001004                          BNE     3$              ;;BRANCH IF NO
 1079  002752  104401  004004                  TYPE    ,$BELL          ;;RING THE TTY BELL
 1080  002756  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
 1081  002760  000451                          BR      5$              ;;EXIT
 1082  002762  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
 1083  002766  001021                          BNE     32$             ;;BRANCH IF NO
 1084  002770  005077  176150                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 1085  002774  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
 1086  002776  105777  176142          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
 1087  003002  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
 1088  003004  117746  176136                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
 1089  003010  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 1090  003014  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 1091  003020  001366                          BNE     31$             ;;BRANCH IF NO
 1092  003022  012777  000100  176114          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
 1093  003030  000002                          RTI                     ;;RETURN
 1094  003032  005237  002602          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
 1095  003036  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
 1096  003042  002405                          BLT     4$              ;;BRANCH IF YES
 1097  003044  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
 1098  003050  003002                          BGT     4$              ;;BRANCH IF YES
 1099  003052  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
 1100  003056  112677  177522          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
 1101  003062  005237  002604                  INC     $TKQIN          ;;UPDATE THE POINTER
 1102  003066  023727  002604  002620          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
 1103  003074  001003                          BNE     5$              ;;BRANCH IF NO
 1104  003076  012737  002610  002604          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
 1105  003104  000002                  5$:     RTI                     ;;RETURN
 1106
 1107                                  ;;**************************************************************
 1108                                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 1109                                  ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
```

```
                                        ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
                                        ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
003106  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
003114  001124                          BNE     15$             ;;EXIT IF NOT
003116  105777  176022                  TSTB    @$TKS           ;;IS A CHAR WAITING?
003122  100121                          BPL     15$             ;;IF NOT, EXIT
003124  117746  176016                  MOVB    @$TKB,-(SP)     ;;YES
003130  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
003134  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
003140  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                                ;;AND EXIT

                                        ;;**************************************************************************
                                        ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                        ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                        ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
003142  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
003150  001674                          BEQ     2$              ;;BRANCH IF YES
003152  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
003154  004737  002620                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
003160  005077  175760                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
003164  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

003172  104401  004022                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
003176  104401  004027          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
003202  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
003206  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
003210  104401  004040                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
003214  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
003216  005046                          CLR     -(SP)           ;;THE NEW SWR
003220  105777  175720          7$:     TSTB    @$TKS           ;;CHAR THERE?
003224  100375                          BPL     7$              ;;IF NOT TRY AGAIN

003226  117746  175714                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
003232  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII

003236  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
003242  001015                          BNE     9$              ;;BRANCH IF NOT
003244  104401  004010                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
003250  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
003254  123727  001135  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
003262  001003                          BNE     8$              ;;BRANCH IF NO
003264  012777  000100  175652          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
003272  000137  001164          8$:     JMP     START           ;;CONTROL-C RESTART

003276  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
003302  001005                          BNE     10$             ;;BRANCH IF NOT
003304  104401  004015                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
003310  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
003314  000737                          BR      19$             ;;LET'S TRY IT AGAIN

003316  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
003322  001022                          BNE     16$             ;;BRANCH IF NO
003324  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
```

```
 1166  003330  001403                          BEQ     11$             ;;BRANCH IF YES
 1167  003332  016677  000002  175600          MOV     2(SP),@SWR      ;;SAVE NEW SWR
 1168  003340  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
 1169  003344  104401  001161          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
 1170  003350  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
 1171  003356  001003                          BNE     15$             ;;BRANCH IF NOT
 1172  003360  012777  000100  175556          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
 1173  003366  000002                  15$:    RTI                     ;;RETURN
 1174  003370  004737  005356          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
 1175  003374  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
 1176  003400  002420                          BLT     18$             ;;BRANCH IF YES
 1177  003402  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
 1178  003406  003015                          BGT     18$             ;;BRANCH IF YES
 1179  003410  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
 1180  003414  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
 1181  003420  001403                          BEQ     17$             ;;BRANCH IF YES
 1182  003422  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
 1183  003424  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
 1184  003426  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
 1185  003430  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
 1186  003434  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
 1187  003440  000667                          BR      7$              ;;GET THE NEXT ONE
 1188  003442  104401  001160          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
 1189  003446  000720                          BR      20$             ;;SIMULATE CONTROL-U
 1190                                  .DSABL  LSB
 1191
 1192
 1193                                  ;;**************************************************************************
 1194                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 1195                                  ;*CALL:
 1196                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
 1197                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 1198                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
 1199                                  ;
 1200
 1201  003450  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
 1202  003452  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
 1203  003460  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
 1204  003464  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
 1205  003466  012746  003474                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
 1206  003472  000002                          RTI                     ;;POP NEW PC AND PS
 1207  003474                          64$:
 1208  003474  005737  002602          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
 1209  003500  001775                          BEQ     1$
 1210  003502  005337  002602                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
 1211  003506  117766  177074  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
 1212  003514  005237  002606                  INC     $TKQOUT         ;;UPDATE THF POINTER
 1213  003520  023727  002606  002620          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
 1214  003526  001003                          BNE     2$              ;;BRANCH IF NO
 1215  003530  012737  002610  002606          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
 1216  003536  000002                  2$:     RTI                     ;;RETURN
 1217                                  ;;**************************************************************************
 1218                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 1219                                  ;*CALL:
 1220                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 1221                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
```

```
                                        ;*                                  ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
                                        ;
003540  010346                  $RDLIN: MOV     R3,-(SP)            ;;SAVE R3
003542  005046                          CLR     -(SP)               ;;CLEAR THE RUBOUT KEY
003544  012703  003774            1$:   MOV     #$TTYIN,R3          ;;GET ADDRESS
003550  022703  004004            2$:   CMP     #$TTYIN+10,R3       ;;BUFFER FULL?
003554  101456                          BLOS    4$                  ;;BR IF YES
003556  104407                          RDCHR                       ;;GO READ ONE CHARACTER FROM THE TTY
003560  112613                          MOVB    (SP)+,(R3)          ;;GET CHARACTER
003562  122713  000177            10$:  CMPB    #177,(R3)           ;;IS IT A RUBOUT
003566  001022                          BNE     5$                  ;;BR IF NO
003570  005716                          TST     (SP)                ;;IS THIS THE FIRST RUBOUT?
003572  001007                          BNE     6$                  ;;BR IF NO
003574  112737  000134  003772          MOVB    #'\,9$              ;;TYPE A BACK SLASH
003602  104401  003772                  TYPE    ,9$
003606  012716  177777                  MOV     #-1,(SP)            ;;SET THE RUBOUT KEY
003612  005303                    6$:   DEC     R3                  ;;BACKUP BY ONE
003614  020327  003774                  CMP     R3,#$TTYIN          ;;STACK EMPTY?
003620  103434                          BLO     4$                  ;;BR IF YES
003622  111337  003772                  MOVB    (R3),9$             ;;SETUP TO TYPEOUT THE DELETED CHAR.
003626  104401  003772                  TYPE    ,9$                 ;;GO TYPE
003632  000746                          BR      2$                  ;;GO READ ANOTHER CHAR.
003634  005716                    5$:   TST     (SP)                ;;RUBOUT KEY SET?
003636  001406                          BEQ     7$                  ;;BR IF NO
003640  112737  000134  003772          MOVB    #'\,9$              ;;TYPE A BACK SLASH
003646  104401  003772                  TYPE    ,9$
003652  005016                          CLR     (SP)                ;;CLEAR THE RUBOUT KEY
003654  122713  000025            7$:   CMPB    #25,(R3)            ;;IS CHARACTER A CTRL U?
003660  001003                          BNE     8$                  ;;BR IF NO
003662  104401  004015                  TYPE    ,$CNTLU             ;;TYPE A CONTROL 'U'
003666  000726                          BR      1$                  ;;GO START OVER
003670  122713  000022            8$:   CMPB    #22,(R3)            ;;IS CHARACTER A '^R'?
003674  001011                          BNE     3$                  ;;BRANCH IF NO
003676  105013                          CLRB    (R3)                ;;CLEAR THE CHARACTER
003700  104401  001161                  TYPE    ,$CRLF              ;;TYPE A 'CR' & 'LF'
003704  104401  003774                  TYPE    ,$TTYIN             ;;TYPE THE INPUT STRING
003710  000717                          BR      2$                  ;;GO PICKUP ANOTHER CHACTER
003712  104401  001160            4$:   TYPE    ,$QUES              ;;TYPE A '?'
003716  000712                          BR      1$                  ;;CLEAR THE BUFFER AND LOOP
003720  111337  003772            3$:   MOVB    (R3),9$             ;;ECHO THE CHARACTER
003724  104401  003772                  TYPE    ,9$
003730  122723  000015                  CMPB    #15,(R3)+           ;;CHECK FOR RETURN
003734  001305                          BNE     2$                  ;;LOOP IF NOT RETURN
003736  105063  177777                  CLRB    -1(R3)              ;;CLEAR RETURN (THE 15)
003742  104401  001162                  TYPE    ,$LF                ;;TYPE A LINE FEED
003746  005726                          TST     (SP)+               ;;CLEAN RUBOUT KEY FROM THE STACK
003750  012603                          MOV     (SP)+,R3            ;;RESTORE R3
003752  011646                          MOV     (SP),-(SP)          ;;ADJUST THE STACK AND PUT ADDRESS OF THE
003754  016666  000004  000002          MOV     4(SP),2(SP)         ;;      FIRST ASCII CHARACTER ON IT
003762  012766  003774  000004          MOV     #$TTYIN,4(SP)
003770  000002                          RTI                         ;;RETURN
003772     000                    9$:   .BYTE   0                   ;;STORAGE FOR ASCII CHAR. TO TYPE
003773     000                          .BYTE   0                   ;;TERMINATOR
003774  000010                  $TTYIN: .BLKB   10                  ;;RESERVE 10 BYTES FOR TTY INPUT
004004  177607  000377          $BELL:  .ASCIZ  <207><377><377>     ;;CODE FOR BELL
004010  041536  005015  000     $CNTLC: .ASCIZ  /^C/<15><12>        ;;CONTROL 'C'
```

```
 1278  004015     136  006525  000012  $CNTLU: .ASCIZ  /^U/<15><12>     ;;CONTROL 'U'
 1279  004022  043536  005015     000  $CNTLG: .ASCIZ  /^G/<15><12>     ;;CONTROL 'G'
 1280  004027     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
 1281  004034  036440  000040
 1282  004040  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
 1283  004046  020075     000
 1284          004052                  .EVEN
 1285                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
 1286
 1287                                  ;;*****************************************************************
 1288                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
 1289                                  ;*CHANGE IT TO BINARY.
 1290                                  ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
 1291                                  ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
 1292                                  ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
 1293                                  ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
 1294                                  ;*CALL:
 1295                                  ;*      RDOCT                           ;;READ AN OCTAL NUMBER
 1296                                  ;*      RETURN HERE                     ;;LOW ORDER BITS ARE ON TOP OF THE STACK
 1297                                  ;*                                      ;;HIGH ORDER BITS ARE IN $HIOCT
 1298
 1299  004052  011646                  $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
 1300  004054  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
 1301  004062  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 1302  004064  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 1303  004066  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 1304  004070  104410                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
 1305  004072  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
 1306  004074  010037  004200                  MOV     R0,5$           ;;AND SAVE IT
 1307  004100  005001                          CLR     R1              ;;CLEAR DATA WORD
 1308  004102  005002                          CLR     R2
 1309  004104  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
 1310  004106  001420                          BEQ     3$              ;;IF ZERO GET OUT
 1311  004110  122716  000060                  CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
 1312  004114  003026                          BGT     4$              ;;IS AN OCTAL DIGIT
 1313  004116  122716  000067                  CMPB    #'7,(SP)
 1314  004122  002423                          BLT     4$
 1315  004124  006301                          ASL     R1              ;;*2
 1316  004126  006102                          ROL     R2
 1317  004130  006301                          ASL     R1              ;;*4
 1318  004132  006102                          ROL     R2
 1319  004134  006301                          ASL     R1              ;;*8
 1320  004136  006102                          ROL     R2
 1321  004140  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
 1322  004144  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
 1323  004146  000756                          BR      2$              ;;LOOP
 1324  004150  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
 1325  004152  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
 1326  004156  010237  004210                  MOV     R2,$HIOCT
 1327  004162  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 1328  004164  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 1329  004166  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 1330  004170  000002                          RTI                     ;;RETURN
 1331  004172  005726                  4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
 1332  004174  105010                          CLRB    (R0)            ;;SET A TERMINATOR
 1333  004176  104401                          TYPE                    ;;TYPE UP THRU THE BAD CHAR.
```

```
 1334  004200  000000                  5$:     .WORD   0
 1335  004202  104401  001160                  TYPE    ,$QUES          ;;'?' 'CR' & 'LF'
 1336  004206  000730                  BR      1$              ;;TRY AGAIN
 1337  004210  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
 1338                                  .SBTTL  MODULE 1.2  SIZE MEMORY
 1339
```

```
1340                                    ;       MODULE 1.2      SIZE MEMORY
1341                                    ;
1342                                    ;       CALLED AS FOLLOWS
1343                                    ;
1344                                    ;       JSR     PC,SIZE
1345                                    ;
1346                                    ;       THIS MODULE SETS UP MEMORY SIZE INCLUDING EXTENDED
1347                                    ;       ADDRESSING BITS IN ACTIVE PARAMETER TABLE
1348                                    ;
1349                                    ;
1350                                    ;
1351  004212  005037  004516           SIZE:   CLR     $KT11           ;DON'T TRY FOR MEM MGT
1352  004216  004737  004460                   JSR     PC,$SIZE        ;GET MEM SIZE WITH KT OFF
1353  004222  005037  011652                   CLR     KT              ;RESET KT AVAILABLE SWITCH
1354  004226  013737  004762  011646         MOV     $LSTAD,MMAX     ;SAVE END OF CORE WITH KT OFF
1355  004234  013737  004762  011644         MOV     $LSTAD,MSIZE    ;GET END OF CORE TO WORK WITH
1356  004242  023727  004762  017776         CMP     $LSTAD,#17776   ;LESS THAN 8K?
1357  004250  103415                         BLO     35$             ;BRANCH IF YES, DON'T PRESERVE LOADERS
1358  004252  162737  000276  011644         SUB     #276,MSIZE      ;SAVE SPACE FOR ABS LOADER
1359  004260  005737  000042                 TST     @#42            ;RUNNING UNDER ACT OR XXDP
1360  004264  001407                         BEQ     35$             ;BR IF NO
1361  004266  023737  000042  000046         CMP     @#42,@#46       ;RUNNING UNDER XXDP CHAIN?
1362  004274  001403                         BEQ     35$             ;BR IF NO
1363  004276  162737  005500  011644         SUB     #5500,MSIZE     ;SAVE SPACE FOR LOADERS
1364  004304  005037  011650           35$:  CLR     EASIZE          ;CLEAN UP EXTENDED ADDR
1365  004310  005037  011704                 CLR     EABITS
1366  004314  037727  174620  002000         BIT     @SWR,#SW10      ;CHECK IF MM TO BE USED
1367  004322  001055                         BNE     30$             ;BR IF NO
1368  004324  012737  000200  004516   10$:  MOV     #200,$KT11
1369  004332  004737  004460           20$:  JSR     PC,$SIZE
1370  004336  005737  004516                 TST     $KT11           ;IS MM ACTIVE
1371  004342  100045                         BPL     30$             ;BR IF NO
1372  004344  012737  000001  011652         MOV     #1,KT           ;SET KT IN USE FLAG
1373  004352  013737  004764  011650         MOV     $LSTBK,EASIZE
1374  004360  062737  000040  011650         ADD     #40,EASIZE      ;INCR TO FIRST INVALID SETTING
1375  004366  013737  004764  011706         MOV     $LSTBK,PHYMAX   ;START GEN OF MAX PHYSICAL
1376  004374  006337  011706                 ASL     PHYMAX          ;  ADDR
1377  004400  006337  011706                 ASL     PHYMAX          ;    LOW
1378  004404  006337  011706                 ASL     PHYMAX          ;      16
1379  004410  006337  011706                 ASL     PHYMAX          ;        BITS
1380  004414  006337  011706                 ASL     PHYMAX
1381  004420  006337  011706                 ASL     PHYMAX
1382  004424  052737  003776  011706         BIS     #3776,PHYMAX    ;SET UP TO USE LAST 1K TOO
1383  004432  013737  004764  011704         MOV     $LSTBK,EABITS
1384  004440  042737  171777  011704         BIC     #171777,EABITS  ;SAVE ONLY EXT ADDR BITS
1385  004446  000337  011704                 SWAB    EABITS          ;MOVE TO BIT POS. 3,2
1386  004452  006337  011704                 ASL     EABITS          ;MOVE TO BIT POS. 4,3 FOR DX11
1387  004456  000207                   30$:  RTS     PC              ;RETURN
1388                                    ;
1389                                    .SBTTL  ROUTINE TO SIZE MEMORY
1390                                    ;
1391                                    ;;*****************************************************************
1392                                    ;*CALL:
1393                                    ;*      JSR     PC,$SIZE
1394                                    ;*      RETURN
1395                                    ;*$LSTAD WILL CONTAIN:
```

```
                                        ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
                                        ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
                                        ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
                                        ;*$KT11 IS THE MEMORY MANAGEMENT KEY
                                        ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
                                        ;*      MUST BE SETUP BEFORE THE CALL
                                        ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
                                        ;*      DETERMINED BY ROUTINE

004460  010046                 $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
004462  010146                         MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
004464  010246                         MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
004466  010346                         MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
004470  013746  000004                 MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
004474  013746  000006                 MOV     @#ERRVEC+2,-(SP)
004500  010600                         MOV     SP,R0           ;;SAVE THE STACK POINTER
                                ;;SET THE ERRVEC PS TO THE PRESENT PS
004502  104400                         TRAP                    ;;PUSH OLD PSW AND PC ON STACK
004504  012637  000006                 MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
004510  012701  003776                 MOV     #3776,R1        ;;SETUP ADDRESS
004514  105727                         TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
004516  000200                 $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
004520  100062                         BPL     $CORE           ;;BR IF NO
004522  012737  004660  000004         MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
004530  005737  177572                 TST     @#SR0           ;;KT11 ARE YOU THERE?
004534  052737  100000  004516         BIS     #100000,$KT11   ;;YES--SET KT11 KEY
004542  005046                         CLR     -(SP)           ;;INITIALIZE FOR 'PAR' LOADING
004544  012702  172340                 MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST 'PAR'
004550  012703  000010                 MOV     #^D8,R3         ;;LOAD EIGHT 'PAR.'S" AND EIGHT 'PDR.'S"
004554  012762  077406  177740 1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
004562  011622                         MOV     (SP),(R2)+      ;;LOAD 'PAR'
004564  062716  000200                 ADD     #200,(SP)       ;;UPDATE FOR NEXT 'PAR'
004570  077307                         SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
004572  012742  177600                 MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
004576  005042                         CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
004600  012737  004616  000004         MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
004606  012737  000020  172516         MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
004614  000401                         BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
004616  022626                 2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
004620  005237  177572         3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
004624  012737  004650  000004         MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
004632  005737  143776         4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
004636  062712  000040                 ADD     #40,(R2)        ;;MAKE A 1K STEP
004642  023712  172356                 CMP     @#KIPAR7,(R2)   ;;LAST ONE?
004646  101371                         BHI     4$              ;;NO--TRY IT
004650  011202                 $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
004652  005037  177572                 CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
004656  000421                         BR      $SIZEX
004660  042737  100000  004516 $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
004666  012737  004716  000004 $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
004674  005002                         CLR     R2              ;;SET UP BANK
004676  062701  004000         1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
004702  062702  000040                 ADD     #40,R2          ;;1K STEP
004706  005711                         TST     (R1)            ;;TRAP ON TIME OUT
004710  022701  177776                 CMP     #177776,R1      ;;LAST ONE
004714  001370                         BNE     1$              ;;NO--TRY AGAIN
```

```
1452  004716  162701  004000          $CROUT: SUB     #4000,R1
1453  004722  162702  000040          $SIZEX: SUB     #40,R2          ;;DROP BACK
1454  004726  010006                          MOV     R0,SP           ;;RESTORE THE STACK
1455  004730  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
1456  004734  012637  000004                  MOV     (SP)+,@#ERRVEC
1457  004740  010137  004762                  MOV     R1,$LSTAD       ;;LAST ADDRESS
1458  004744  010237  004764                  MOV     R2,$LSTBK       ;;LAST BANK
1459  004750  012603                          MOV     (SP)+,R3        ;;RESTORE R3
1460  004752  012602                          MOV     (SP)+,R2        ;;RESTORE R2
1461  004754  012601                          MOV     (SP)+,R1        ;;RESTORE R1
1462  004756  012600                          MOV     (SP)+,R0        ;;RESTORE R0
1463  004760  000207                          RTS     PC
1464  004762  000000          $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
1465  004764  000000          $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
1466
1467                                  .SBTTL  MODULE 1.3  CHECK PARAMETERS
```

```
                                        ;       MODULE 1.3      CHECK PARAMETERS
                                        ;
                                        ;       CALLED AS FOLLOWS
                                        ;
                                        ;       JSR     R5,CHECKP
                                        ;       .WORD   X
                                        ;       THIS MODULE HAS RESPONSIBILITY FOR VALIDATING PARAMETERS
                                        ;       INPUT BY THE OPERATOR.  IT COMPARES THE INPUT PARAMETER
                                        ;       AGAINST PARAMETER LIMITS DEFINED IN THE INPUT PARAMETER
                                        ;       LIMITS TABLE.  IF THE PARAMETER IS OUTSIDE THE SPECIFIED
                                        ;       LIMITS, AN ERROR MESSAGE IS OUTPUT AND ALL ONES IS RETURNED
                                        ;       IN R0.  IF THE PARAMETER IS OK, ALL ZEROS IS RETURNED IN
                                        ;       R0.
                                        ;
                                        ;       X DEFINES THE PARAMETER TO CHECK AS FOLLOWS:
                                        ;       X=2     STARTING TEST NUMBER
                                        ;       X=3     BASE ADDR OF DX CONTROL REGS
                                        ;       X=4     DX VECTOR ADDRESS
                                        ;       X=5     DX PRIORITY.
                                        ;       THE PARAMETER TO BE CHECKED IS PASSED IN R0.
                                        ;
004766  012501                  CHECKP: MOV     (R5)+,R1        ;GET PARAMETER TO BE CHECKED
004770  120127  000002                  CMPB    R1,#2           ;IS IT TEST NO?
004774  001007                          BNE     10$             ;BR IF NO
004776  020037  011614                  CMP     R0,LTESTN       ;CHECK LOW LIMIT
005002  002443                          BLT     90$             ;BR IF ERROR
005004  020037  011616                  CMP     R0,HTESTN       ;CHECK HIGH LIMIT
005010  003040                          BGT     90$             ;BR IF ERROR
005012  000446                          BR      80$             ;GO TO OK RETURN
005014  120127  000003          10$:    CMPB    R1,#3           ;IS IT BASE ADDR OF DX
005020  001007                          BNE     20$             ;BR IF NO
005022  020037  011620                  CMP     R0,LUBA         ;CHECK LOW LIMIT
005026  002431                          BLT     90$             ;BR IF BAD
005030  020037  011622                  CMP     R0,HUBA         ;CHECK HIGH LIMIT
005034  003026                          BGT     90$             ;BR IF BAD
005036  000434                          BR      80$             ;OK, RETURN
005040  120127  000004          20$:    CMPB    R1,#4           ;IS IT THE VECTOR?
005044  001007                          BNE     30$             ;BR IF NO
005046  020037  011624                  CMP     R0,LVEC         ;CHECK THE LOW LIMIT
005052  002417                          BLT     90$             ;BR IF BAD
005054  020037  011626                  CMP     R0,HVEC         ;CHECK THE HIGH LIMIT
005060  003014                          BGT     90$             ;BR IF BAD
005062  000422                          BR      80$             ;OK, RETURN
005064  120127  000005          30$:    CMPB    R1,#5           ;IS PARM PRIORITY LEVEL
005070  001007                          BNE     40$             ;BR IF NO
005072  020037  011630                  CMP     R0,LPRIOR       ;CHECK LOW LIMIT
005076  002405                          BLT     90$             ;BR IF BAD
005100  020037  011632                  CMP     R0,HPRIOR       ;CHECK HIGH LIMIT
005104  003002                          BGT     90$             ;BR IF BAD
005106  000410                          BR      80$             ;OK, RETURN
005110  000000          40$:    HALT                            ;ERROR, BAD MODULE CALL PARM
005112  004537  005134          90$:    JSR     R5,PRINT        ;PRINT PARM LIMIT ERROR
005116  012723                          .WORD   BADP            ;ADDR OF MESSAGE
005120  000000                          .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
005122  012700  177777                  MOV     #-1,R0          ;SET ERROR RETURN CODE
005126  000401                          BR      95$             ;RETURN
```

```
 1524  005130  005000                  80$:    CLR     R0              ;SET GOOD RETURN CODE
 1525  005132  000205                  95$:    RTS     R5              ;RETURN TO CALLER.
 1526
 1527                                          .SBTTL  MODULE 1.3.1  WRITE TTY
```

```
                                ;       MODULE 1.3.1    WRITE TTY
                                ;
                                ;       THIS MODULE HANDLES ALL OUTPUT TO THE OPERATOR
                                ;
                                ;       CALLED AS FOLLOWS
                                ;
                                ;       JSR     R5      PRINT
                                ;       .WORD   MSG             ;ADDRESS OF MESSAGE OR OCT #
                                ;       .WORD   CTL
                                ;       RETURN
                                ;
                                ;       CTL=    BIT0=1          ;DONT - CRLF BEFORE MESSAGE
                                ;               BIT15=1         ;CONVERT OCTAL TO ASCII AND PRINT
                                ;
005134                          PRINT:
005134  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
005136  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
005140  012500                          MOV     (R5)+,R0        ;GET ADDR OF MSG
005142  012501                          MOV     (R5)+,R1        ;GET PRINT CONTROL WORD
005144  032701  000001                  BIT     #1,R1           ;DO WE NEED A CRLF?
005150  001002                          BNE     10$             ;BR IF NO
005152  104401                          TYPE                    ;TRAP TO TYPE ROUTINE
005154  013300                          .WORD   #CRLF1          ;POINTER TO CRLF ASCII
005156  005701                  10$:    TST     R1              ;IS IT OCTAL?
005160  100003                          BPL     20$             ;BR IF NO
005162  011046                          MOV     (R0),-(SP)      ;PUT OCTAL # ON STACK
005164  104402                          TYPOC                   ;TYPE 6 CHAR VIA TRAP
005166  000404                          BR      40$             ;DONE
005170  010037  005176          20$:    MOV     R0,25$          ;PUT MESSAGE ADDR INLINE
005174  104401                          TYPE                    ;TRAP TO TYPE ASCII R1N
005176  000000                  25$:    .WORD   0               ;POINTER TO MSG GOES HERE
005200                          40$:
005200  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
005202  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
005204  000205                          RTS     R5              ;RETURN TO CALLER
                                ;TYPE TRAP CALL COMES HERE
                                .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
                                ;*
                                ;
005206  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
005212  100002                          BPL     1$              ;;BR IF YES
```

```
1584  005214  000000                        HALT                    ;;HALT HERE IF NO TERMINAL
1585  005216  000407                        BR      3$              ;;LEAVE
1586  005220  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
1587  005222  017600  000002              MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
1588  005226  112046                2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1589  005230  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
1590  005232  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
1591  005234  012600                60$:    MOV     (SP)+,R0        ;;RESTORE R0
1592  005236  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
1593  005242  000002                        RTI                     ;;RETURN
1594  005244  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
1595  005250  001430                        BEQ     8$
1596  005252  122716  000200              CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
1597  005256  001006                        BNE     5$
1598  005260  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
1599  005262  104401                        TYPE                    ;;TYPE A CR AND LF
1600  005264  001161                        $CRLF
1601  005266  105037  005422               CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
1602  005272  000755                        BR      2$              ;;GET NEXT CHARACTER
1603  005274  004737  005356        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
1604  005300  123726  001156        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
1605  005304  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
1606  005306  013746  001154               MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
1607                                                                ;;AND THE NULL CHAR.
1608  005312  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
1609  005316  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
1610  005320  004737  005356               JSR     PC,$TYPEC       ;;GO TYPE A NULL
1611  005324  105337  005422               DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
1612  005330  000770                        BR      7$              ;;LOOP
1613
1614                                  ;HORIZONTAL TAB PROCESSOR
1615
1616  005332  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
1617  005336  004737  005356        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
1618  005342  132737  000007  005422       BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
1619  005350  001372                        BNE     9$              ;;TAB STOP
1620  005352  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
1621  005354  000724                        BR      2$              ;;GET NEXT CHARACTER
1622  005356  105777  173566        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
1623  005362  100375                        BPL     $TYPEC
1624  005364  116677  000002  173560       MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
1625  005372  122766  000015  000002       CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
1626  005400  001003                        BNE     1$              ;;BRANCH IF NO
1627  005402  105037  005422               CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
1628  005406  000406                        BR      $TYPEX          ;;EXIT
1629  005410  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
1630  005416  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
1631  005420  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
1632  005422  000000        $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
1633  005424  000207        $TYPEX: RTS     PC
1634
1635                                  ;TYPOC TRAP CALL COMES TO $TYPOC TAG BELOW
1636                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
1637
1638                                  ;;**********************************************************************
1639                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
```

```
1640                                    ;*OCTAL (ASCII) NUMBER AND TYPE IT.
1641                                    ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1642                                    ;*CALL:
1643                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
1644                                    ;*      TYPOS                   ;;CALL FOR TYPEOUT
1645                                    ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
1646                                    ;*      .BYTE   M               ;;M=1 OR 0
1647                                    ;*                                      ;;1=TYPE LEADING ZEROS
1648                                    ;*                                      ;;0=SUPPRESS LEADING ZEROS
1649                                    ;*
1650                                    ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
1651                                    ;*$TYPOS OR $TYPOC
1652                                    ;*CALL:
1653                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
1654                                    ;*      TYPON                   ;;CALL FOR TYPEOUT
1655                                    ;*
1656                                    ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1657                                    ;*CALL:
1658                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
1659                                    ;*      TYPOC                   ;;CALL FOR TYPEOUT
1660
1661  005426  017646  000000            $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
1662  005432  116637  000001  005651            MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
1663  005440  112637  005653                    MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
1664  005444  062716  000002                    ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
1665  005450  000406                            BR      $TYPON
1666  005452  112737  000001  005651  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
1667  005460  112737  000006  005653            MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
1668  005466  112737  000005  005650  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
1669  005474  010346                            MOV     R3,-(SP)        ;;SAVE R3
1670  005476  010446                            MOV     R4,-(SP)        ;;SAVE R4
1671  005500  010546                            MOV     R5,-(SP)        ;;SAVE R5
1672  005502  113704  005653                    MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
1673  005506  005404                            NEG     R4
1674  005510  062704  000006                    ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
1675  005514  110437  005652                    MOVB    R4,$OMODE       ;;SAVE IT FOR USE
1676  005520  113704  005651                    MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
1677  005524  016605  000012                    MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
1678  005530  005003                            CLR     R3              ;;CLEAR THE OUTPUT WORD
1679  005532  006105                    1$:     ROL     R5              ;;ROTATE MSB INTO "C"
1680  005534  000404                            BR      3$              ;;GO DO MSB
1681  005536  006105                    2$:     ROL     R5              ;;FORM THIS DIGIT
1682  005540  006105                            ROL     R5
1683  005542  006105                            ROL     R5
1684  005544  010503                            MOV     R5,R3
1685  005546  006103                    3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
1686  005550  105337  005652                    DECB    $OMODE          ;;TYPE THIS DIGIT?
1687  005554  100016                            BPL     7$              ;;BR IF NO
1688  005556  042703  177770                    BIC     #177770,R3      ;;GET RID OF JUNK
1689  005562  001002                            BNE     4$              ;;TEST FOR 0
1690  005564  005704                            TST     R4              ;;SUPPRESS THIS 0?
1691  005566  001403                            BEQ     5$              ;;BR IF YES
1692  005570  005204                    4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
1693  005572  052703  000060                    BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
1694  005576  052703  000040            5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
1695  005602  110337  005646                    MOVB    R3,8$           ;;SAVE FOR TYPING
```

```
 1696  005606  104401  005646                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
 1697  005612  105337  005650          7$:     DECB    $OCNT           ;;COUNT BY 1
 1698  005616  003347                          BGT     2$              ;;BR IF MORE TO DO
 1699  005620  002402                          BLT     6$              ;;BR IF DONE
 1700  005622  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 1701  005624  000744                          BR      2$              ;;GO DO THE LAST DIGIT
 1702  005626  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
 1703  005630  012604                          MOV     (SP)+,R4        ;;RESTORE R4
 1704  005632  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 1705  005634  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 1706  005642  012616                          MOV     (SP)+,(SP)
 1707  005644  000002                          RTI                     ;;RETURN
 1708  005646     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 1709  005647     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 1710  005650     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 1711  005651     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 1712  005652  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
 1713                                          .SBTTL  MODULE 2.0  INTERRUPT HANDLERS
```

```
                                        ;
                                        ;       MODULE 2.0
                                        ;
                                        ;       INTERRUPT HANDLERS
                                        ;
                                        ;
                                        ;
                                        .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.

005654  016646  000002        $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
005660  042716  000020                BIC     #20,(SP)        ;; THE CALLER--DO NOT ALLOW
005664  012746  005672                MOV     #1$,-(SP)       ;; T-BIT TRAPS
005670  000002                        RTI                     ;;SET THE NEW STATUS
005672  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
005674  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
005700  005740                        TST     -(R0)           ;;BACKUP BY 2
005702  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
005704  006300                        ASL     R0              ;;POSITION FOR INDEXING
005706  016000  005726                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
005712  000200                        RTS     R0              ;;GO TO ROUTINE


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

005714  011646                $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
005716  016666  000004  000002        MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
005724  000002                        RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                        ;       -------
005726  005714                $TRPAD: .WORD   $TRAP2
005730  005206                        $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
005732  005452                        $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
005734  005426                        $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
005736  005466                        $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)

005740  003176                        $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING

005742  003106                        $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
005744  003450                        $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
005746  003540                        $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
005750  004052                        $RDOCT  ;;CALL=RDOCT    TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
006752  012737  006116  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
006760  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
006766  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
005770  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
005772  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
005774  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
005776  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
006000  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
006002  017746  173132                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
006006  010637  006122                  MOV     SP,$SAVR6       ;;SAVE SP
006012  012737  006024  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
006020  000000                          HALT
006022  000776                          BR      .-2             ;;HANG UP

                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
006024  012737  006116  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
006032  013706  006122                  MOV     $SAVR6,SP       ;;GET SP
006036  005037  006122                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
006042  005237  006122          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
006046  001375                          BNE     1$              ;;OF  WORD
006050  012677  173064                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
006054  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
006056  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
006060  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
006062  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
006064  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
006066  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
006070  012737  005752  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
006076  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
006104  104401                          TYPE                    ;REPORT THE POWER FAILURE
006106  006124          $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
006110  012716                  MOV     (PC)+,(SP)              ;;RESTART AT RESTART
006112  001334          $PWRAD: .WORD   RESTART                 ;;RESTART ADDRESS
006114  000002                  RTI
006116  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
006120  000776                  BR      .-2                     ;; BEFORE THE POWER DOWN WAS COMPLETE
006122  000000          $SAVR6: 0                               ;;PUT THE SP HERE
006124  005015  047520  042527  $POWER: .ASCIZ  <15><12>'POWER'
006132  000122
                                        .EVEN
```

```
1816                                            .SBTTL  MODULE 3.0.1  TST1  FAST NPR DATA TEST
1817                                    ;
1818                                    ;
1819                                    ;       MODULE 3.0.1  FAST NPR DATA TEST
1820                                    ;
1821                                    ;       THIS IS A FUNCTIONAL TEST TO CHECK THE ABILITY
1822                                    ;       OF THE DX TO DO NPR'S OF DATA INTO MEMORY
1823                                    ;       WITH LARGE BYTE COUNTS.  THE DXBA IS CHECKED FOR
1824                                    ;       ITS ABILITY TO COUNT AND ALL CARRIES IN THE COUNTER ARE
1825                                    ;       EXERCISED.  THE DATA TRANSFERS ACCESS ALL
1826                                    ;       AVAILABLE MEMORY EXCEPT WHERE THE PROGRAM IS LOCATED,
1827                                    ;       AND THE AREA WHICH MAY BE OCCUPIED BY A LOADER.
1828                                    ;
1829                                    ;
1830  006134  004537  006530           TST1:   JSR     R5,INIT         ;INITIALIZE THE SOFTWARE, PS, ETC
1831  006140  004737  006560                   JSR     PC,DXRES        ;PUT THE DX IN A GOOD KNOWN STATE
1832  006144  004537  010300                   JSR     R5,BUFAD        ;GET FIRST BUFFER BASE AND SET UP KT
1833  006150  100402                           .WORD   100402          ;CTL BIT 15=INITIALIZE, BUFFER BASE DIVISABLE
1834                                                                    ;BY 400, SET UP BUFMAX
1835  006152  013700  011640                   MOV     AVECT,R0        ;SET UP
1836  006156  012720  006230                   MOV     #INTR1,(R0)+    ;DX INTERRUPT
1837  006162  012710  000340                   MOV     #340,(R0)       ;AND DX PSW
1838  006166  012737  000402  011712   LOOP1:  MOV     #402,BUFSZE     ;BUFFER SIZE A LITTLE BIGGER THAN OFFSET
1839                                                                    ;TO CREATE A CARRY OUT OF HIGH ORDER BIT
1840  006174  004737  006510           10$:    JSR     PC,BUFCLR       ;CLEAR THE BUFFER
1841  006200  012737  100525  011600           MOV     #100525,DXM00   ;SET DATA PATTERN AND OPOUT
1842  006206  053777  011716  003324   GO1:    BIS     EACUR,@DXCSA    ;SET UP DX EXT ADDR BITS
1843  006214  013777  011722  003322           MOV     BFBAPH,@DXBAA   ;SET UP DX BUS ADDR REG.
1844  006222  004537  006366                   JSR     R5,DXGO         ;START THE DX
1845  006226  177376                           .WORD   -402            ;BYTE COUNT
1846
1847                                    ;INTERRUPT WILL RETURN HERE AFTER DATA TRANSFER IS COMPLETE
1848
1849  006230  005737  006506           INTR1:  TST     INTERR          ;DID WE GET HERE VIA INTER
1850  006234  001006                           BNE     8$              ;BR IF NO, STACK IS OK
1851  006236  012716  006244                   MOV     #5$,@SP         ;SET UP RETURN ADDR
1852  006242  000002                           RTI                     ;RELIEVE INTERRUPT STATUS
1853  006244  012705  006252           5$:     MOV     #8$,R5          ;SET UP JSR RETURN ADDR
1854  006250  000205                           RTS     R5              ;RELIEVE SUBROUTINE STATUS
1855  006252  013700  011710           8$:     MOV     BUFBAS,R0       ;GET STARTING ADDR
1856  006256  013701  011712                   MOV     BUFSZE,R1       ;GET BUFFER SIZE
1857  006262  123720  011600           10$:    CMPB    DXM00,(R0)+     ;CHECK A BYTE
1858  006266  001404                           BEQ     15$             ;BRANCH IF NO ERROR
1859  006270  004537  010752           12$:    JSR     R5,ERRR         ;REPORT AN ERROR
1860  006274  000001                           .WORD   1               ;ERROR ID
1861  006276  011766                           .WORD   ERR1            ;ADDR OF ERROR MESSAGE
1862  006300  005301                   15$:    DEC     R1              ;DECREMENT BUFFER COUNT
1863  006302  001367                           BNE     10$             ;LOOP UNTIL ALL OF BUFFER CHECKED
1864  006304  042777  000200  003226           BIC     #DONE,@DXCSA    ;TURN OFF FAST MODE
1865  006312  042777  000004  003240           BIC     #SOSIEN,@DXESA  ;TURN OFF FAST MODE
1866  006320  042777  000030  003212           BIC     #30,@DXCSA      ;CLEAR POSSIBLE CARRY IN EA BITS
1867  006326  105737  011600                   TSTB    DXM00           ;CHECK IF FIRST OF TWO PASSES
1868  006332  100404                           BMI     30$             ;BR IF NOT FIRST PASS
1869  006334  012737  100652  011600           MOV     #100652,DXM00   ;SET UP SECOND DATA PATTERN
1870  006342  000721                           BR      GO1             ;DO TRANSFER WITH THIS DATA
1871
```

```
1872  006344  004737  006510          30$:    JSR     PC,BUFCLR       ;PUT BUFFER BACK TO ZEROS
1873  006350  004537  010300          ENDT1:  JSR     R5,BUFAD        ;GET NEXT BUFFER ADDRESS
1874  006354  000402                          .WORD   402             ;BASE TO BE DIVISABLE BY 400
1875  006356  005701                          TST     R1              ;CHECK FOR OUT OF MEMORY
1876  006360  001702                          BEQ     LOOP1           ;BRANCH IF THERE IS MORE
1877  006362  000137  001410                  JMP     ENDTST          ;OTHERWISE WE'RE DONE WITH TEST
1878                                  ;
1879                                  ;  SUBROUTINE TO START THE DX IN SOSIEN MODE
1880                                  ;
1881  006366  013777  011600  003154  DXGO:   MOV     DXMOO,@DXMOA    ;SEND DATA AND OP OUT TO MAINT REG
1882  006374  012577  003146                  MOV     (R5)+,@DXBCA    ;SET UP BYTE COUNT FROM PASSED PARM.
1883  006400  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
1884  006402  012746  006410                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
1885  006406  000002                          RTI                     ;;POP NEW PC AND PS
1886  006410                          64$:
1887  006410  052777  000103  003122          BIS     #103,@DXCSA     ;SET  FUNCTION TO IBM WRITE
1888                                                                  ;AND ENABLE INTERRUPTS
1889  006416  052777  060000  003124          BIS     #SELO!HLDO,@DXMOA       ;FORCE DX SELECTION
1890  006424  042777  060000  003116          BIC     #SELO!HLDO,@DXMOA       ;DROP SELECT LINES
1891  006432  052777  002000  003110          BIS     #CMDO,@DXMOA            ;RAISE COMMAND OUT
1892  006440  042777  002000  003102          BIC     #CMDO,@DXMOA            ;DROP COMMAND OUT
1893  006446  052777  000004  003104          BIS     #SOSIEN,@DXESA          ;ENABLE FAST SERVICE OUT/SERV IN
1894                                                                  ;TO UNLEASH NPR'S
1895  006454  005037  006506                  CLR     INTERR          ;CLEAR NO INTERRUPT OCCURED FLAG
1896  006460  005001                          CLR     R1
1897  006462  005301                  10$:    DEC     R1              ;WAIT HERE FOR INTERRUPT
1898  006464  001376                          BNE     10$
1899  006466  004537  010752                  JSR     R5,ERRR         ;ERROR, INTERRUPT NEVER  CAME
1900  006472  000002                          .WORD   2               ;ERROR IDENTIFER
1901  006474  012012                          .WORD ERR2              ;ERROR MESSAGE POINTER
1902  006476  012737  000001  006506          MOV     #1,INTERR       ;SET NO INTERRUPT OCCURED FLAG
1903  006504  000205                          RTS     R5              ;CONTINUE TESTING
1904  006506  000000          INTERR: .WORD   0       ;SET TO 1 IF DX11 FAILED TO INTERRUPT
1905
1906
1907                                  ;       SUBROUTINE TO CLEAR THE CURRENT BUFFER
1908
1909  006510  013700  011710  BUFCLR: MOV     BUFBAS,R0       ;GET BUFFER START
1910  006514  013701  011712          MOV     BUFSZE,R1       ;GET BUFFER SIZE
1911  006520  105020          10$:    CLRB    (R0)+           ;CLEAR A BYTE
1912  006522  005301                  DEC     R1              ;COUNT IT
1913  006524  001375                  BNE     10$             ;LOOP TO CLEAR BUFFER
1914  006526  000207                  RTS     PC              ;RETURN TO CALLER
1915
```

```
                                                .SBTTL  MODULE 3.1  TEST INITIALIZATION
                                        ;
                                        ;       MODULE 3.1      TEST SOFTWARE INITIALIZATION
                                        ;
                                        ;
                                        ;       CALLED AS FOLLOWS:
                                        ;
                                        ;       JSR     R5,INIT
                                        ;       RETURN COMES HERE
                                        ;
                                        ;
006530                                  INIT:
006530  012746  000340                          MOV     #340,-(SP)      ;;PUT NEW PS ON STACK
006534  012746  006542                          MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
006540  000002                                  RTI                     ;;POP NEW PC AND PS
006542                                  64$:
006542  005037  011676                          CLR     ERR             ;RESET ERROR OCCURRED INDICATION
006546  010537  011702                          MOV     R5,LOCKAD       ;SET UP LOCK ON ERROR RETURN ADDR
006552  005037  006506                          CLR     INTERR          ;CLEAR MISSED INTR FLAG
006556  000205                                  RTS     R5
                                        ;
                                        ;       HARDWARE INITIALIZATION SUBROUTINE
                                        ;
006560  052777  000010  002772          DXRES:  BIS     #10,@DXESA      ;DISABLE DX TIME OUT, DEBUG ONLY
006566  042777  000200  002744                  BIC     #DONE,@DXCSA    ;CLEAR DONE AND LOCKO
006574  042777  000006  002736                  BIC     #6,@DXCSA       ;SET FUNCTION TO RESET
006602  052777  000001  002730                  BIS     #1,@DXCSA       ;ISSUE THE DX RESET
006610  042777  001230  002722                  BIC     #1230,@DXCSA    ;CLEAR POTENTIAL DISASTER OF ONLINE OR INTEN
                                                                        ;CLEAR EA BITS ALSO
006616  012777  014000  002716                  MOV     #SPW,@DXOSA     ;SET UP CU OFFSET TO SPW TABLE
006624  005737  011764                          TST     SPWMVD          ;IS SPW TABLE AT HOME BASE?
006630  001403                                  BEQ     5$              ;BR IF YES
006632  012777  016000  002702                  MOV     #SPW+2000,@DXOSA        ;SET UP ALTERNATE SPW LOCATION
006640  032777  100000  002706          5$:     BIT     #100000,@DXCBA  ;IS LOCKO ON?
006646  001402                                  BEQ     10$             ;BR IF NO
006650  000005                                  RESET                   ;WE'RE IN TROUBLE, DX DIDN'T RESET
006652  000742                                  BR      DXRES           ;GO BACK AND MAKE SURE IT'S CLEARED
006654  000207                          10$:    RTS     PC              ;RETURN
```

```
                        .SBTTL  MODULE 3.0.2  TST2  NPR ADDRESS UNIQUENESS
                ;
                ;
                ;       MODULE 3.0.2  NPR DATA ADDR UNIQUENESS TEST
                ;
                ;
                ;       THIS MODULE TESTS THE DX11'S ABILITY TO UNIQUELY ADDRESS EACH
                ;       WORD OF MEMORY.  THIS TEST IS ACCOMPLISHED BY WRITING A BACKGROUND
                ;       IN A BLOCK OF MEMORY.  A WORD IS THEN TRANSFERRED FROM THE DX TO THE
                ;       MEMORY BLOCK.  THE CPU CHECKS THAT THE WORD ARRIVES AT ITS
                ;       EXPECTED DESTINATION.  IF YES THE WORD IS RESTORED TO MATCH THE
                ;       BACKGROUND, AND THE NEXT WORD IS TESTED.  THIS IS REPEATED
                ;       FOR EACH WORD IN THE BLOCK AND FOR EACH BLOCK IN MEMORY.
                ;
                ;       NOTE:  THIS TEST MAY NOT DETECT ADDRESSING PROBLEMS
                ;       WHICH ARE COMMON TO THE CPU AND DX11.  I.E. UNIBUS ADDRESSING PROBLEMS.
                ;
006656 004537 006530    TST2:   JSR     R5,INIT         ;INITIALIZE THE SOFTWARE
006662 004737 006560            JSR     PC,DXRES        ;INITIALIZE THE HARDWARE
006666 004537 010300            JSR     R5,BUFAD        ;GET STARTING BUFFER ADDRESS
006672 101000                   .WORD   101000          ;PARM MEANS INIT AND SET UP1000 BYTE BUFFER
006674 013700 011640            MOV     AVECT,R0        ;GET THE DX VECTOR ADDR
006700 012720 006766            MOV     #INTR2,(R0)+    ;SET UP RETURN ADDR
006704 012710 000340            MOV     #340,(R0)       ;SET INTR PSW
006710 012737 001000 011712 LOOP2:  MOV     #1000,BUFSZE    ;ESTABLISH BUFFER SIZE
006716 004737 006510            JSR     PC,BUFCLR       ;CLEAR THE BUFFER
006722 013737 011710 011720     MOV     BUFBAS,DATAD    ;INITIALIZE DATA ADDR
006730 013737 011722 011724     MOV     BFBAPH,BFADPH   ;GET PHYSICAL BASE ADDR
006736 053777 011716 002574 LOP2:   BIS     EACUR,@DXCSA    ;SET EA BITS IN THE DX11
006744 012737 100377 011600     MOV     #100377,DXM00   ;DATA PATTERN TO BE WRITTEN BY DX
                ;
006752 013777 011724 002564     MOV     BFADPH,@DXBAA   ;SET UP DX DATA ADDRESS
006760 004537 006366            JSR     R5,DXGO         ;START THE DX
006764 177776                   .WORD   -2              ;BYTE COUNT = 2
                ;
                ;DX INTERRUPTS TO HERE AFTER NPR'ING A WORD
006766 005737 006506    INTR2:  TST     INTERR          ;DID WE COME HERE VIA DX INTR
006772 001006                   BNE     8$              ;BR IS NO,  STACK IS OK
006774 012716 007002            MOV     #5$,@SP         ;SET UP RETURN ADDR
007000 000002                   RTI                     ;RELIEVE INTERRUPT STATUS
007002 012705 007010    5$:     MOV     #8$,R5          ;SET UP RETURN ADDR
007006 000205                   RTS     R5              ;RELIEVE SUBROUTINE STATUS
007010 027727 002704 177777 8$: CMP     @DATAD,#-1      ;DID DATA GET WHERE IT WAS SUPPOSED TO?
                                        ;ASSUME CPU CAN ADDRESS MEMORY CORRECTLY
007016 001404                   BEQ     10$             ;BRANCH IF OK
007020 004537 010752            JSR     R5,ERRR         ;REPORT THE ERROR
007024 000003                   .WORD   3               ;ERROR NUMBER THREE
007026 012032                   .WORD   ERR3            ;ADDR OF ERROR MESSAGE
007030 005077 002664    10$:    CLR     @DATAD          ;RESTORE THE WORD UNDER TEST
007034 042777 000200 002476     BIC     #DONE,@DXCSA    ;TURN OFF INTERRUPTS
007042 042777 000004 002510     BIC     #SOSIEN,@DXESA  ;CLEAR FAST MODE
007050 042777 000030 002462     BIC     #30,@DXCSA      ;CLEAR POSSIBLE CARRY IN EA BITS
007056 062737 000002 011724     ADD     #2,BFADPH       ;BUMP THE PHYSICAL ADDRESS
007064 062737 000002 011720     ADD     #2,DATAD        ;BUMP DATA POINTER
007072 023737 011720 011714     CMP     DATAD,BUFMAX    ;DONE WITH BUFFER?
```

```
2012  007100  101716                  BLOS    LOP2            ;BRANCH IF NO
2013
2014  007102  004737  006510          JSR     PC,BUFCLR       ;RESTORE BUFFER
2015  007106  004537  010300          JSR     R5,BUFAD        ;GET NEXT BUFFER
2016  007112  001000                  .WORD   1000            ;SIZE =1000
2017  007114  005701                  TST     R1              ;DID WE GET ONE?
2018  007116  001674                  BEQ     LOOP2           ;BRANCH IF YES
2019  007120  000137  001410          JMP     ENDTST          ;OTHERWISE TERMINATE TEST
```

```
2020                                            .SBTTL          TST3  SPW ADDRESSING
2021
2022
2023                                            MODULE 3.0.3    SPW ADDRESSING TEST
2024
2025
2026                                            THIS MODULE TESTS THE ABILITY OF THE DX TO ACCESS THE STATUS
2027                                            POINTER WORD (SPW) TABLE IN ALL POSSIBLE LOCATIONS IN MEMORY.
2028                                            A BASE ADDRESS FOR THE SPW IS GENERATED.
2029                                            THIS SPW TABLE IS FILLED WITH A PATTERN OF 000DEV
2030                                            WHERE DEV IS EQUAL TO THE DISPLACEMENT OF THE WORD FROM THE
2031                                            SPW BASE; I.E.
2032                                            SPW BASE    =000000
2033                                            SPW BASE +1 =000001
2034                                                      ''
2035                                                     ''
2036                                                      ''
2037                                                      ''
2038                                            SPW BASE+255=000377
2039
2040                                            GIVING A TOTAL OF 256 WORDS FOR THE SPW TABLE WITH IMMEDIATE
2041                                            STATUS EQUAL TO THE DEVICE NUMBER.
2042                                            AN ISS SEQUENCE IS INITIATED WITH A DEVICE # OF 0.  IF
2043                                            ADRECC AND ADRECD DON'T COME ON FOR THIS DEV, NO TESTING IS DONE,
2044                                            THE DEV # IS INCREMENTED, AND ISS IS DONE AGAIN.
2045                                            IF ADRECC AND ADRECD COME ON IT MEANS THE DX JUMPERS ARE CUT TO
2046                                            RESPOND TO THIS DEV NUMBER. THE ISS SEQUENCE IS THEN CONTINUED
2047                                            UNTIL THE STATUS IS IN THE DXOS REG.  THIS STATUS IS THEN
2048                                            CHECKED AND SHOULD BE EQUAL TO THE DEV # IF THE SPW WAS
2049                                            ACCESSED CORRECTLY.
2050
2051                                            THIS PROCEDURE IS REPEATED FOR ALL 256 POSSIBLE DEVICES AT
2052                                            ALL POSSIBLE LOCATIONS FOR THE SPW.
2053
2054
2055                                            THE FIRST RECOGNIZED  DEV NO. AND COUNT OF THE FIRST
2056                                            GROUP OF RECOGNIZED DEVICES IS SAVED AND PRINTED AT END
2057                                            PASS TIME.  IF NO DEV IS RECOGNIZED AND ERROR MESSAGE
2058                                            IS OUTPUT AT THE END OF THE TEST.
2059
2060
2061  007124  004537  006530          TST3:   JSR     R5,INIT         ;INITIALIZE THE SOFTWARE
2062  007130  005037  011744                  CLR     RECORD          ;SET UP TO RECORD IBM DEV RECOG.
2063  007134  005037  011746                  CLR     FRSTD   ;SET TO RECORD FIRST DEV RECOGNIZED
2064  007140  005037  011752                  CLR     DEVCT           ;CLEAR DEVICE COUNTER
2065  007144  004737  006560                  JSR     PC,DXRES        ;RESET THE DX
2066  007150  012737  000400  011732          MOV     #400,CMD        ;INITIALIZE A COMMAND OF ZERO
2067  007156  004537  010300                  JSR     R5,BUFAD        ;GET THE FIRST BUFFER ADDR
2068  007162  102000                          .WORD   102000          ;SPW TO BE ON 2000 BOUNDARY
2069  007164  012737  002000  011712          MOV     #2000,BUFSZE    ;SET UP BUFFER SIZE
2070  007172  004737  006510                  JSR     PC,BUFCLR       ;CLEAR BUFFER
2071  007176  005037  011730          LOOP3:  CLR     DEV             ;START AT DEVICE 0
2072  007202  013700  011710                  MOV     BUFBAS,R0       ;GET SPW TABLE BASE
2073  007206  013720  011730          10$:    MOV     DEV,(R0)+       ;MOV IMMED. STATUS TO SPW EQUAL TO DEV
2074  007212  005237  011730                  INC     DEV             ;INC TO NEXT STATUS = NEXT DEV
2075  007216  022737  000377  011730          CMP     #255.,DEV       ;IS DST FULL?
```

```
2076  007224  103370                          BHIS    10$             ;DO MORE IF IT IS NOT
2077  007226  005037  011730                  CLR     DEV             ;START AT DEV 0
2078  007232  013777  011722  002302  LOP3:   MOV     BFBAPH,@DXOSA   ;SET UP OFFSET FOR SPW
2079  007240  053777  011716  002272          BIS     EACUR,@DXCSA    ;SET UP EXT ADDR BITS
2080  007246  004537  007514                  JSR     R5,PTYGEN       ;ASSIGN PARITY TO THE DEVICE BYTE
2081  007252  011730                          .WORD   DEV             ;ADDR OF DATA TO ASSIGN PARITY TO
2082  007254  113777  011730  002254          MOVB    DEV,@DXCAA      ;SET DEV TO CUAR
2083  007262  053777  011730  002260          BIS     DEV,@DXMOA      ;PUT DEV ADDR ON BUSOUT
2084  007270  052777  004000  002252          BIS     #ADRO,@DXMOA    ;RAISE ADDR OUT
2085  007276  032777  000001  002250          BIT     #1,@DXCBA       ;IS  ADRECD ON?
2086  007304  001452                          BEQ     20$             ;BR IF ADDR NOT RECOGNIZED
2087  007306  032777  000002  002240          BIT     #2,@DXCBA       ;IS ADRECC ON?
2088  007314  001446                          BEQ     20$             ;BR IF ADDR NOT RECOGNIZED
2089  007316  004537  007554                  JSR     R5,SAVDEV       ;GO SAVE RECOGNIZED DEV DATA
2090  007322  052777  060000  002220          BIS     #HLDO!SELO,@DXMOA       ;RAISE SEL OUT, HOLD OUT
2091  007330  042777  004000  002212          BIC     #ADRO,@DXMOA    ;DROP ADRO
2092  007336  043777  011730  002204          BIC     DEV,@DXMOA      ;REMOVE DEVICE FROM BUSO
2093  007344  053777  011732  002176          BIS     CMD,@DXMOA      ;PUT CMD =0 ON BUSOUT
2094  007352  052777  002000  002170          BIS     #CMDO,@DXMOA    ;RAISE CMD OUT
2095  007360  043777  011732  002162          BIC     CMD,@DXMOA      ;REMOVE CMD FROM BUSO
2096  007366  042777  002000  002154          BIC     #CMDO,@DXMOA    ;DROP CMD OUT
2097  007374  123777  011730  002140          CMPB    DEV,@DXOSA      ;IS STATUS EQUAL TO DEV?
2098  007402  001404                          BEQ     10$             ;BR IF OK
2099  007404  004537  010752                  JSR     R5,ERRR         ;REPORT ERROR
2100  007410  000004                          .WORD   4               ;ERROR ID
2101  007412  012071                          .WORD   ERR4            ;ERROR MESSAGE ADDR
2102  007414  052777  001000  002126  10$:    BIS     #SRVO,@DXMOA    ;RAISE SERVICE OUT TO RELIEVE STATUS
2103  007422  042777  001000  002120          BIC     #SRVO,@DXMOA    ;DROP SERVICE OUT
2104  007430  000402                          BR      30$
2105  007432  004537  007620          20$:    JSR     R5,DONTSV       ;GO HANDLE RECOGNIZED DEV DATA
2106  007436  004737  006560          30$:    JSR     PC,DXRES        ;RESTORE DX
2107  007442  005237  011730                  INC     DEV             ;GO TO NEXT DEV
2108  007446  105737  011730                  TSTB    DEV             ;LAST DEV?
2109  007452  001267                          BNE     LOP3            ;BR IF MORE DEVICES
2110  007454  004737  006510                  JSR     PC,BUFCLR       ;CLEAR OLD BUFFER BEFORE GETTING NEW ONE
2111  007460  004537  010300                  JSR     R5,BUFAD        ;GET NEXT BUFFER BASE
2112  007464  002000                          .WORD   2000            ;DIVISABLE BY 2000
2113  007466  005701                          TST     R1              ;DID WE GET ANOTHER BUFFER?
2114  007470  001642                          BEQ     LOOP3           ;BR IF YES AND DO NEXT ONE
2115  007472  005737  011752                  TST     DEVCT           ;TEST IS DONE, DID WE EVER FIND A
2116                                                                  ;RECOGNIZED DEVICE?
2117  007476  001004                          BNE     40$             ;BR IF YES
2118  007500  004537  010752                  JSR     R5,ERRR         ;ERROR, NO ADREDD AND ADRECD FOR ANY
2119                                                                  ;IBM DEVICE NUMBER
2120  007504  000005                          .WORD   5
2121  007506  012120                          .WORD   ERR5            ;ADDR OF ERROR MSG
2122  007510  000137  001410          40$:    JMP     ENDTST          ;TERMINATE TEST
2123
2124
2125
2126
2127
2128                                  ;       PARITY ASSIGNMENT SUBROUTINE
2129                                  ;
2130                                  ;       CALLED AS FOLLOWS:
2131                                  ;       JSR     R5,PTYGEN
```

```
                        ;       .WORD   ADDR            ;ADDRESS OF DATA TO WHICH PARITY WILL BE
                        ;                               ;ASSIGNED
                        ;       RETURN HERE
                        ;
                        ;       REGISTERS R2, R3, R4 ARE DISTROYED BY THIS SUBROUTINE
                        ;
007514  012503          PTYGEN: MOV     (R5)+,R3        ;GET ADDR OF DATA
007516  011304                  MOV     (R3),R4         ;GET THE DATA
007520  005002                  CLR     R2              ;INITIALIZE THE PARTIY SW
007522  106304          PTY2:   ASLB    R4              ;CHECK A BIT
007524  102001                  BVC     PTY3            ;BR IF ONES
007526  005102                  COM     R2              ;OTHERWISE FLIP THE PARITY SW
007530  106304          PTY3:   ASLB    R4              ;TWO BITS GET CHECKED AT ONCE
007532  001373                  BNE     PTY2            ;IF R4 = 0 WE'D BE DONE
007534  005702                  TST     R2              ;DID THE SWITCH END UP SET?
007536  100403                  BMI     PTY4            ;BR IF NO
007540  052713  000400          BIS     #400,(R3)       ;SET THE BIT IN THE TARGET DATA
007544  000402                  BR      PTY5            ;EXIT
007546  042713  000400  PTY4:   BIC     #400,(R3)       ;CLEAR THE BIT IN THE TARGET DATA
007552  000205          PTY5:   RTS     R5              ;RETURN


007554  005737  011744  SAVDEV: TST     RECORD          ;ARE WE RECORDING DEVICES FOUND?
007560  001016                  BNE     20$             ;BR IF NO AND EXIT
007562  005737  011746          TST     FRSTD           ;IS THIS THE VERY FIRST DEV RECOGNIZED?
007566  001011                  BNE     10$             ;BR IF NO
007570  052737  000001  011746  BIS     #1,FRSTD        ;SET FIRST DEV FOUND SW
007576  113737  011730  011750  MOVB    DEV,FRSTDV      ;SAVE FIRST DEVICE FOUND
007604  012737  000001  011762  MOV     #1,DEVFND       ;SET SWITCH = DEVICE FOUND
007612  005237  011752  10$:    INC     DEVCT           ;COUNT THE DEV
007616  000205          20$:    RTS     R5


007620  005737  011752  DONTSV: TST     DEVCT           ;HAVE WE FOUND ANY DEV YET?
007624  001403                  BEQ     30$             ;BR IF NO
007626  052737  000001  011744  BIS     #1,RECORD       ;TURN OFF DEV RECORDING
007634  000205          30$:    RTS     R5
```

```
                                        .SBTTL  MODULE 3.0.4   TST4  DST ACCESS TEST
                                ;
                                ;
                                ;       MODULE 3.0.4    DEVICE STATUS TABLE (DST) ACCESS TEST
                                ;
                                ;       THIS TEST CHECKS THE ABILITY TO ACCESS ALL BYTES OF THE
                                ;       DST IN ALL AREAS OF MEMORY.  THE OBJECTIVE IS TO CHECK LOADING
                                ;       OF THE DXBA FROM THE SPW DST BASE, AND COMMAND (CUCR)
                                ;
                                ;       A VALID IBM DEVICE FROM TST3 IS CHECKED.  THEN A BASE ADDR
                                ;       IS GENERATED FOR THE DST.  EACH BYTE OF THE DST IS THEN LOADED
                                ;       WITH ITS OFFSET FROM THE DST BASE.
                                ;
                                ;       A SPW ENTRY FOR THE VALID IBM DEVICE IS GENERATED WITH AP-
                                ;       PROPRIATE DST BASE AND ZERO IMMEDIATE STATUS TO ASSURE A DST
                                ;       ACCESS.  AN ISS SEQUENCE IS DONE WITH THE VALID DEV AND
                                ;       EVERY POSSIBLE IBM COMMAND THEREBY ACCESSING EVERY LOCATION
                                ;       IN THE DST.  AFTER THE ISS, THE STATUS WILL EQUAL THE COMMAND
                                ;       OR AND ERROR IS FLAGGED.
                                ;
                                ;
007636  004537  006530        TST4:   JSR     R5,INIT         ;INITIALIZE THE SOFTWARE
007642  004737  006560                JSR     PC,DXRES        ;INITIALIZE THE HARDWARE
007646  005737  011762                TST     DEVFND          ;DO WE HAVE A DEVICE?
007652  001006                        BNE     10$             ;BR IF YES
007654  004537  005134                JSR     R5,PRINT        ;TEST 3 MUST BE RUN SUCCESSFULLY FIRST
007660  012215                        .WORD   T4MSG           ;TO FIND AND IBM DEVICE #
007662  000000                        .WORD   0               ;AND ASSURE GOOD SPW ACCESS CAPABILITY
007664  000137  010274                JMP     END4            ;ABORT TEST
007670  005037  011730        10$:    CLR     DEV             ;CLEAN UP CURRENT DEV NO.
007674  113737  011750  011730        MOVB    FRSTDV,DEV      ;GET THE VALID DEV
007702  004537  007514                JSR     R5,PTYGEN       ;ASSIGN PARITY TO IT
007706  011730                        .WORD   DEV
007710  004537  010300                JSR     R5,BUFAD        ;GET A BUFFER OF 256 BYTES
007714  100400                        .WORD   100400
007716  012737  000400  011712        MOV     #400,BUFSZE     ;SET BUFFER SIZE
007724  004737  006510                JSR     PC,BUFCLR       ;CLEAR THE BUFFER
007730  005000                LOOP4:  CLR     R0              ;START WITH CMD = 0
007732  013701  011710                MOV     BUFBAS,R1       ;GET THE DST BASE
007736  110021                10$:    MOVB    R0,(R1)+        ;SET CMD IN DST
007740  005200                        INC     R0              ;NEXT CMD
007742  022700  000377                CMP     #255.,R0        ;ARE ALL STATUS = TO CMD?
007746  103373                        BHIS    10$             ;DO 256 BYTES
007750  005000                        CLR     R0
007752  013700  011730                MOV     DEV,R0          ;GET DEVICE
007756  042700  177400                BIC     #177400,R0      ;;STRIP THE PARITY BIT
007762  006300                        ASL     R0              ;MAKE IT A WORD OFFSET
007764  017701  001552                MOV     @DXOSA,R1       ;GET THE SPW BASE
007770  042701  001777                BIC     #1777,R1        ;CLEAN IT UP
007774  050100                        BIS     R1,R0           ;ADD DEV TO OFFSET
007776  005737  011652                TST     KT              ;MM IN USE?
010002  001412                        BEQ     15$             ;BR IF NO
010004  013701  011716                MOV     EACUR,R1        ;GET EXT ADDR BITS
010010  000301                        SWAB    R1              ;POSITION FOR USE IN PAR
010012  006201                        ASR     R1
```

```
 2232  010014  010137  172352                 MOV     R1,@#KIPAR5     ;USE PAR5 TO GET TO SPW TABLE
 2233  010020  042700  040000                 BIC     #40000,R0       ;SET SPW ADDR TO USE PAR5
 2234  010024  052700  120000                 BIS     #120000,R0
 2235  010030  013701  011722          15$:   MOV     BFBAPH,R1       ;SET PHY BUFFER BASE
 2236  010034  105001                         CLRB    R1              ;IMMED STATUS = 0
 2237  010036  010110                         MOV     R1,(R0)         ;SET UP SPW
 2238  010040  005037  011732                 CLR     CMD             ;START WITH CMD = 0
 2239  010044  004537  007514          LOP4:  JSR     R5,PTYGEN       ;GET COMMAND PARITY
 2240  010050  011732                         .WORD   CMD
 2241  010052  053777  011716  001460         BIS     EACUR,@DXCSA    ;SET UP EXT ADDR BITS
 2242  010060  113777  011730  001450         MOVB    DEV,@DXCAA      ;SET DEVICE TO CUAR
 2243  010066  053777  011730  001454         BIS     DEV,@DXMOA      ;PUT DEV ADDR ON BUS OUT
 2244  010074  052777  004000  001446         BIS     #ADR0,@DXMOA    ;ADDR OUT
 2245  010102  052777  060000  001440         BIS     #HLDO!SELO,@DXMOA       ;SELECT & HOLD OUT
 2246  010110  042777  004000  001432         BIC     #ADR0,@DXMOA    ;DROP ADDR OUT
 2247  010116  043777  011730  001424         BIC     DEV,@DXMOA      ;DROP DEV FROM BUS
 2248  010124  053777  011732  001416         BIS     CMD,@DXMOA      ;COMMAND TO BUS
 2249  010132  052777  002000  001410         BIS     #CMD0,@DXMOA    ;COMMAND OUT
 2250  010140  043777  011732  001402         BIC     CMD,@DXMOA      ;CLR CMD FROM BUS
 2251  010146  042777  002000  001374         BIC     #CMD0,@DXMOA    ;DROP COMMAND OUT
 2252  010154  122737  000004  011732         CMPB    #4,CMD          ;IS IT A SENSE COMMAND?
 2253  010162  001004                         BNE     5$              ;BR IF NO
 2254  010164  105777  001352                 TSTB    @DXOSA          ;STATUS FORCED TO ZERO ON SENSE CMD
 2255  010170  001411                         BEQ     10$             ;BR IF GOOD
 2256  010172  000404                         BR      7$              ;REPORT ERROR
 2257  010174  123777  011732  001340   5$:   CMPB    CMD,@DXOSA      ;IS STATUS EQUAL DEVICE?
 2258  010202  001404                         BEQ     10$             ;BR IF YS
 2259  010204  004537  010752          7$:    JSR     R5,ERRR         ;ERROR IN DST ACCESS
 2260  010210  000006                         .WORD   6               ;ERROR #
 2261  010212  012174                         .WORD   ERR6            ;MESSAGE ADDR
 2262  010214  052777  001000  001326  10$:   BIS     #SRV0,@DXMOA    ;RELIEVE STATUS
 2263  010222  042777  001000  001320         BIC     #SRV0,@DXMOA
 2264  010230  004737  006560                 JSR     PC,DXRES        ;CLEAR THE DX
 2265  010234  005237  011732                 INC     CMD             ;TRY NEXT COMMAND
 2266  010240  105737  011732                 TSTB    CMD             ;DONE IF BACK TO ZERO
 2267  010244  001277                         BNE     LOP4            ;BR TO DO NEXT CMD
 2268  010246  004737  006510                 JSR     PC,BUFCLR       ;CLEAR OLD BUFFER
 2269  010252  004537  010300          20$:   JSR     R5,BUFAD        ;GET NEXT BUFFER BASE
 2270  010256  000400                         .WORD   400
 2271  010260  005701                         TST     R1              ;DID WE GET A BUFFER?
 2272  010262  001004                         BNE     END4            ;BR IF NO,END TEST
 2273  010264  005737  011722                 TST     BFBAPH          ;IS BUFFER BASE 0'S?
 2274  010270  001770                         BEQ     20$             ;BR IF YES, THE DX DOES NOT DO A
 2275                                                                 ;DST FETCH IF SPW ENTRY IS ALL ZEROS
 2276  010272  000616                         BR      LOOP4           ;DO NEXT BUFFER
 2277  010274  000137  001410          END4:  JMP     ENDTST          ;END OF TEST 4
```

```
                                        .SBTTL  BASE ADDRESS GENERATOR MODULE
                                ;       MODULE 3.2    BASE ADDRESS GENERATOR
                                ;
                                ;       THIS MODULE SETS UP THE STARTING ADDRESS FOR DX BUFFERS
                                ;       AND HAS RESPONSIBILITY FOR ESTABLISHING EXTENDED ADDR
                                ;       BITS,BUFFER BASE, BUFFER SIZE, AND HAS CONTROL OF
                                ;       THE MEMORY MGT. REGISTERS.  KIPAR6 IS ALWAYS USED
                                ;       TO ACCESS THE DXBUFFERS IF MEM MGT. IS ENABLED.
                                ;
                                ;       THIS MODULE IS CALLED AS FOLLOWS:
                                ;
                                ;       JSR     R5,BUFAD
                                ;       .WORD   PARM
                                ;       RETURN HERE WITH R1=0 MEANS BUFFER ALLOCATED
                                ;                         R1=1 MEANS NO MEMORY LEFT
                                ;       PARM IS DEFINED AS FOLLOWS:
                                ;       BIT 15=1          INITIALIZE  TO FIRST BUFFER LOCATION
                                ;       BIT 15=0          GET NEXT BUFFER LOCATION
                                ;       BITS 14:0         BASE ADDR OF BUFFER MUST BE DIVISABLE BY
                                ;                         THIS NUMBER
010300  012500          BUFAD:  MOV     (R5)+,R0          ;GET THE PASSED PARAMETER
010302  005001                  CLR     R1                ;RESET 'OUT OF MEMORY'' INDICATOR
010304  005700                  TST     R0                ;CHECK FOR INITIALIZE SW
010306  100050                  BPL     NEXTB             ;BRANCH IF NOT TO INITIALIZE
010310  005037  011716          CLR     EACUR             ;START INIT PROCESS BY CLEARING
                                                          ;THE EA BITS
010314  012702  016000          MOV     #ENDALL,R2        ;GET VIRTUAL END OF PROGRAM ADDR
010320  042700  100000          BIC     #100000,R0        ;CLEAR HIGH BIT OF PARAM
010324  060002                  ADD     R0,R2             ;OFFSET ABOVE PROGRAM
010326  162700  000001          SUB     #1,R0             ;DEVELOP A MASK
010332  040002                  BIC     R0,R2             ;TO MAKE BASE DIVISABLE BY PROPER AMT.
010334  010237  011710          MOV     R2,BUFBAS         ;SAVE VIRTUAL BUFFER BASE
010340  010237  011722          MOV     R2,BFBAPH         ;SAVE PHYSICAL BUFFER BASE, SAME AS VIRT
                                                          ;AT THIS POINT.
010344  005737  011652          TST     KT                ;IS KT IN USE?
010350  001421                  BEQ     ENDB              ;BRANCH IF NO TO END OF MODULE
010352  012703  000006          MOV     #6,R3             ;INIT LOOP COUNTER
010356  005037  172354          CLR     @#KIPAR6          ;START TO INITIALIZE KIPAR6
010362  006202          10$:    ASR     R2      ;SHIFT ADDRESS RIGHT 6 PLACES
010364  077302                  SOB     R3,10$            ;TO ALLIGN WITH PAR
010366  010237  172354          MOV     R2,@#KIPAR6       ;SET UP PAR6 TO REFERENCE FIRST BUFFER
010372  042737  177700  011710  BIC     #177700,BUFBAS    ;CLEAR UP VIRTUAL BUFFER ADDR
010400  052737  140000  011710  BIS     #140000,BUFBAS    ;BUFFER ACCESS IS ALWAYS THRU PAR6
010406  012737  000001  177572  MOV     #1,@#SR0          ;TURN ON MM
010414  063700  011710  ENDB:   ADD     BUFBAS,R0         ;FIND TOP OF CURRENT BUFFER
010420  005200                  INC     R0        ;
010422  010037  011714          MOV     R0,BUFMAX         ;SAVE VIRT END OF BUFFER
010426  000550                  BR      ENDB1             ;DONE WITH INITIALIZATION
                                ;
                                ;       COME HERE IF NEXT BUFFER ADDR IS WANTED
                                ;
010430  060037  011710  NEXTB:  ADD     R0,BUFBAS         ;INCR TO NEXT BUFFER SPACE
010434  042737  000077  011710  BIC     #77,BUFBAS        ;CLEAR POSSIBLE LOW ORDER BITS
010442  060037  011722          ADD     R0,BFBAPH         ;STEP PHYSICAL ADDR
010446  042737  000077  011722  BIC     #77,BFBAPH        ;CLEAR POSSIBLE LOW ORDER BITS
```

```
2334  010454  103007                          BCC     2$              ;BR IF NO CARRY
2335  010456  062737  000010  011716          ADD     #10,EACUR       ;STEP EXTENDED ADDR BITS
2336  010464  023737  011716  011704          CMP     EACUR,EABITS    ;HAVE THE EXT ADR BITS GONE TOO HIGH?
2337  010472  101125                          BHI     30$             ;BR IF YES TO SET ALL DONE SW AND EXIT
2338  010474  005737  011652          2$:     TST     KT              ;IS MEM MNG IN USE?
2339  010500  001511                          BEQ     20$             ;BRANCH IF NO
2340  010502  032737  020000  011710          BIT     #20000,BUFBAS   ;WAS THERE A CARRY TO ACCESS NEXT PAR?
2341  010510  001417                          BEQ     10$             ;BR IF NO CARRY
2342  010512  062737  000200  172354          ADD     #200,@#KIPAR6   ;TAKE A 4K STEP
2343  010520  023737  172354  011650          CMP     @#KIPAR6,EASIZE ;CHECK FOR TOO HIGH
2344  010526  103402                          BLO     5$              ;BRANCH IF OK
2345  010530  005201                          INC     R1              ;SET THE ''ALL DONE'' SWITCH
2346  010532  000506                          BR      ENDB1           ;EXIT
2347  010534  042737  177700  011710  5$:     BIC     #177700,BUFBAS  ;START BUFFER BASE OVER
2348  010542  052737  140000  011710          BIS     #140000,BUFBAS  ;ACCESS THROUGH PAR6
2349  010550  013703  011710          10$:    MOV     BUFBAS,R3       ;GET THE BASE ADDR
2350  010554  060003                          ADD     R0,R3           ;CALCULATE TOP OF BUFFER
2351  010556  032703  020000                  BIT     #20000,R3       ;DIT IT CARRY ACROSS A PAR BOUNDARY
2352  010562  001322                          BNE     NEXTB           ;BR IS YES.  GO BACK AND GET A BETTER ADDR
2353  010564  010337  011714                  MOV     R3,BUFMAX       ;SAVE TOP OF BUFFER ADDR
2354  010570  013703  011722                  MOV     BFBAPH,R3       ;GET PHYSICAL BUF BASE
2355  010574  060003                          ADD     R0,R3           ;ADD BUFFER SIZE TO GET BUF TOP
2356  010576  023737  011716  011704          CMP     EACUR,EABITS    ;ARE WE IN THE TOP PHYSICAL BANK?
2357  010604  001006                          BNE     12$             ;BR IF NO
2358  010606  020337  011706                  CMP     R3,PHYMAX       ;ARE WE OVER THE TOP OF MEM?
2359  010612  103055                          BHIS    30$             ;BR IF YES, SET DONE SW
2360  010614  020327  000004                  CMP     R3,#4           ;WAS THERE A CARRY OUT OF TOP BANK?
2361  010620  101452                          BLOS    30$             ;BR IF YES, WE MUST BE DONE
2362  010622  013703  011722          12$:    MOV     BFBAPH,R3       ;GET PHYSICAL BUFFER BASE
2363  010626  060003                          ADD     R0,R3           ;FIND TOP OF PHY BUFFER
2364  010630  103677                          BCS     NEXTB           ;BR IF CARRY OVER 32K BNDRY
2365                                                                  ;AND GO BACK AND GET A BUFFER WHICH DOESN'T
2366                                                                  ;STRADDLE A 32K BOUNDARY
2367  010632  005737  011716                  TST     EACUR           ;ARE WE IN THE FIRST BANK?
2368  010636  001010                          BNE     3$              ;BR IF NO
2369  010640  020337  011644                  CMP     R3,MSIZE        ;ARE WE IN THE LOADER AREA?
2370  010644  101405                          BLOS    3$              ;BR IF OK
2371  010646  023737  011722  011646          CMP     BFBAPH,MMAX     ;IS BUFFER ABOVE LOADERS?
2372  010654  103001                          BHIS    3$              ;BRANCH IF WE'RE OK
2373  010656  000664                          BR      NEXTB           ;GO BACK AND GET A BETTER BUFFER ADDR
2374  010660  012777  014000  000654  3$:     MOV     #SPW,@DXOSA     ;SET UP SPW ADDR
2375  010666  005037  011764                  CLR     SPWMVD          ;CLEAR SPW BASE ADDR MOVED SWITCH
2376  010672  020327  014000                  CMP     R3,#SPW         ;IS TOP OF BUFFER IN SPW?
2377  010676  101411                          BLOS    15$             ;BR IF OK
2378  010700  023727  011722  016000          CMP     BFBAPH,#SPW+2000        ;IS BUFFER BASE IN TT?
2379  010706  103005                          BHIS    15$             ;BR IF NO
2380  010710  012777  016000  000624          MOV     #SPW+2000,@DXOSA        ;MOV THE SPW
2381  010716  005237  011764                  INC     SPWMVD          ;SET SPW ADDR MOVED TO ALTERNATE SWITCH
2382  010722  000412                  15$:    BR      ENDB1           ;GO TO END OF MODULE
2383  010724  013737  011710  011714  20$:    MOV     BUFBAS,BUFMAX   ;START TO GET BUFMAXIMUM
2384  010732  060037  011714                  ADD     R0,BUFMAX       ;ADD IN BUFFER SIZE
2385  010736  023737  011714  011644          CMP     BUFMAX,MSIZE    ;SEE IF WE'RE OUT OF MEMORY
2386  010744  101401                          BLOS    ENDB1           ;BR IF OK
2387  010746  005201                  30$:    INC     R1              ;SET OUT OF MEMORY SWITCH
2388  010750  000205                  ENDB1:  RTS     R5              ;RETURN FROM MODULE
```

```
                                        .SBTTL  MODULE 3.3      REPORT ERRORS
                                ;       MODULE 3.3      ERROR REPORTING
                                ;
                                ;       CALLED AS FOLLOWS:
                                ;
                                ;       JSR     R5,ERRR
                                ;       .WORD   X
                                ;       .WORD   Y
                                ;       RETURN (UNLESS LOCK ON ERROR FUNCTION OVERRIDES RETURN)
                                ;       X EQUALS ERROR IDENTIFICATION NUMBER
                                ;       Y EQUALS ADDR OF UNIQUE ERROR MESSAGE
                                ;
                                ;       THIS MODULE FORMATS AND PRINTS ERROR MESSAGES AND HANDLES
                                ;       ERROR OPTIONS SUCH AS INHIBIT ERROR MESSAGES, SHORT ERROR
                                ;       REPORT, SCOPE LOOP, HALT ON ERROR, ETC.
                                ;
010752  032777  100000  170160  ERRR:   BIT     #SW15,@SWR      ;HALT ON ERROR ON?
010760  001401                          BEQ     5$              ;BR IF NO
010762  000000                          HALT                    ;SW15 ON AND ERROR OCCURRED.

010764  013737  011666  011676  5$:     MOV     CTESTN,ERR      ;SET ERROR DETECTED SW
010772  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
010774  012746  011002                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
011000  000002                          RTI                     ;;POP NEW PC AND PS
011002                          64$:
011002  005737  011700                  TST     LOCK            ;ARE WE LOCKED ON ERROR?
011006  001407                          BEQ     10$             ;BR IF NO
011010  032777  040000  170122          BIT     #SW14,@SWR      ;SCOPE LOOP SW ON?
011016  001413                          BEQ     15$             ;BR IF NO
011020  013705  011702                  MOV     LOCKAD,R5       ;GET THE LOOP ADDR
011024  000205                          RTS     R5              ;GO TO IT
011026  032777  040000  170104  10$:    BIT     #SW14,@SWR      ;SCOPE LOOP ON?
011034  001404                          BEQ     15$             ;BR IF NO
011036  013737  011666  011700          MOV     CTESTN,LOCK     ;SET LOCKED ON ERR SW
011044  000402                          BR      20$
011046  005037  011700          15$:    CLR     LOCK            ;TURN OF LOCKED SW
011052  010537  011734          20$:    MOV     R5,ERRPC        ;SAVE ERR PC
011056  012537  011736                  MOV     (R5)+,ERRID     ;SAVE ERROR ID
011062  012537  011740                  MOV     (R5)+,ERRMS     ;SAVE ERROR MESSAGE ADDR
011066  032777  020000  170044          BIT     #SW13,@SWR      ;INHIBIT PRINT ON?
011074  001402                          BEQ     25$             ;BR IF NO
011076  000137  011520                  JMP     97$             ;TERMINATE MODULE
011102  004537  005134          25$:    JSR     R5,PRINT        ;PRINT ERRPC
011106  012762                          .WORD   ERPC            ;ADDR OF MESSAGE
011110  000000                          .WORD   0               ;NO SPECIAL PRINT CONTROLS
011112  162737  000004  011734          SUB     #4,ERRPC        ;BACKUP ADDR TO CORRECT VALUE
011120  004537  005134                  JSR     R5,PRINT        ;PRINT THE PC NUMBER ITSSELF
011124  011734                          .WORD   ERRPC
011126  100001                          .WORD   100001          ;NO CRLF, CONVERT TO ASCII
011130  004537  005134                  JSR     R5,PRINT        ;PRINT ERROR NUMBER ON SAME LINE
011134  012775                          .WORD   ERNO
011136  000001                          .WORD   1               ;INHIBIT CRLF
011140  004537  005134                  JSR     R5,PRINT        ;PRINT THE OCT
011144  011736                          .WORD   ERRID
011146  100001                          .WORD   100001
```

```
 2445  011150  013737  011740  011162          MOV     ERRMS,30$       ;GET MESSAGE POINTER
 2446  011156  004537  005134                  JSR     R5,PRINT        ;PRINT THE UNIQUE MESSAGE
 2447  011162  000000                  30$:    .WORD   0               ;FILLED FROM TWO LINES UP
 2448  011164  000000                          .WORD   0               ;NO SPECIAL CONTROLS
 2449  011166  032777  010000  167744          BIT     #SW12,@SWR      ;SHORT ERROR REPORT ON?
 2450  011174  001145                          BNE     95$             ;BR IF YES, WE'RE DONE
 2451  011176  004537  005134                  JSR     R5,PRINT        ;PRINT TEST NO.
 2452  011202  013011                          .WORD   ERTN
 2453  011204  000000                          .WORD   0
 2454  011206  004537  005134                  JSR     R5,PRINT        ;PRINT THE OCTAL TEST NO.
 2455  011212  011666                          .WORD   CTESTN
 2456  011214  100001                          .WORD   100001
 2457  011216  022737  000005  011736          CMP     #5,ERRID        ;IS IT ERROR #5?
 2458  011224  001531                          BEQ     95$             ;BR IF YES, REST OF STUFF IS MEANINGLESS
 2459  011226  004537  005134                  JSR     R5,PRINT        ;PRINT VIRT ADDR
 2460  011232  013024                          .WORD   ERADR
 2461  011234  000000                          .WORD   0
 2462  011236  004537  005134                  JSR     R5,PRINT        ;PRINT VIRT ADDR OCTAL
 2463  011242  011710                          .WORD   BUFBAS
 2464  011244  100001                          .WORD   100001
 2465  011246  005737  011652                  TST     KT              ;IS MEM MGT IN USE?
 2466  011252  001431                          BEQ     90$             ;BR IF NO WE'RE DONE
 2467  011254  004537  005134                  JSR     R5,PRINT        ;PRINT PHYSICAL ADDR
 2468  011260  013057                          .WORD   ERADPH
 2469  011262  000000                          .WORD   0
 2470  011264  004537  005134                  JSR     R5,PRINT        ;PRINT PHYSICAL ADDR OCTAL
 2471  011270  011722                          .WORD   BFBAPH
 2472  011272  100001                          .WORD   100001
 2473  011274  004537  005134                  JSR     R5,PRINT        ;PRINT EA BITS
 2474  011300  013104                          .WORD   EABIT
 2475  011302  000000                          .WORD   0
 2476  011304  013737  011716  011742          MOV     EACUR,EAERR     ;GET THE EA BITS
 2477  011312  006237  011742                  ASR     EAERR           ;POSITION BITS
 2478  011316  006237  011742                  ASR     EAERR           ;  TO BITS 1:0
 2479  011322  006237  011742                  ASR     EAERR
 2480  011326  004537  005134                  JSR     R5,PRINT        ;PRINT THE OCTAL EA BITS
 2481  011332  011742                          .WORD   EAERR
 2482  011334  100001                          .WORD   100001
 2483  011336  023727  011736  000003  90$:    CMP     ERRID,#3                ;IS IT ERROR NO. 3?
 2484  011344  001010                          BNE     93$             ;BR IF NO
 2485  011346  004537  005134                  JSR     R5,PRINT        ;PRINT DATA ADDRESS ASCII
 2486  011352  013125                          .WORD   DATAA
 2487  011354  000000                          .WORD   0
 2488  011356  004537  005134                  JSR     R5,PRINT        ;PRINT DATA ADDR OCT
 2489  011362  011720                          .WORD   DATAD
 2490  011364  100001                          .WORD   100001
 2491  011366  023727  011736  000004  93$:    CMP     ERRID,#4        ;IS IT ERROR #4?
 2492  011374  001004                          BNE     94$             ;BR IF NO
 2493  011376  113737  011730  011760          MOVB    DEV,EXP         ;GET DEV #
 2494  011404  000416                          BR      96$
 2495  011406  023727  011736  000006  94$:    CMP     ERRID,#6        ;IS IT ERROR #6?
 2496  011414  001035                          BNE     95$             ;BR IF NO
 2497  011416  122737  000004  011732          CMPB    #4,CMD          ;WAS IT A SENSE CMD?
 2498  011424  001003                          BNE     98$             ;BR IF NO
 2499  011426  005037  011760                  CLR     EXP             ;EXPECTED STATUS IS ZERO FOR SENSE
 2500  011432  000403                          BR      96$
```

```
 2501  011434  113737  011732  011760  98$:    MOVB    CMD,EXP          ;GET EXPECTED CMD
 2502  011442  117737  000074  011756  96$:    MOVB    @DXOSA,ACTU      ;GET ACTUAL STATUS
 2503  011450  004537  005134                   JSR     R5,PRINT         ;PRINT ACTUAL MESSAGE
 2504  011454  013250                           .WORD   ACTUMS
 2505  011456  000000                           .WORD   0
 2506  011460  004537  005134                   JSR     R5,PRINT         ;PRINT STATUS OCTAL
 2507  011464  011756                           .WORD   ACTU
 2508  011466  100001                           .WORD   100001
 2509  011470  004537  005134                   JSR     R5,PRINT         ;PRINT EXPECTED STATUS
 2510  011474  013262                           .WORD   EXPMS
 2511  011476  000001                           .WORD   1
 2512  011500  004537  005134                   JSR     R5,PRINT         ;PRINT EXPECTED OCTAL
 2513  011504  011760                           .WORD   EXP
 2514  011506  100001                           .WORD   100001
 2515  011510  004537  005134          95$:     JSR     R5,PRINT         ;SPACE THE PAPER
 2516  011514  013300                           .WORD   CRLF1
 2517  011516  000000                           .WORD   0
 2518  011520  022737  001566  000042  97$:     CMP     #$ENDAD,@#42     ;UNDER ACT?
 2519  011526  001001                           BNE     99$              ;BR IF NO
 2520  011530  000000                           HALT                     ;HALT ON ACT ERROR
 2521
 2522  011532  000205                  99$:     RTS     R5               ;RETURN FROM MODULE
```

```
2523                                    ;
2524                                    ;
2525                                    ;       DX ACCESS TABLES
2526                                    ;
2527                                    ;
2528  011534  000000                    DXDSA:  .WORD   0       ;DEVICE STATUS REG ADDR.
2529  011536  000000                    DXCAA:  .WORD   0       ;COMMAND AND ADDR REG ADDR.
2530  011540  000000                    DXCSA:  .WORD   0       ;CONTROL UNIT STATUS REG ADDR
2531  011542  000000                    DXOSA:  .WORD   0       ;OFFSET AND STATUS REG ADDR.
2532  011544  000000                    DXBAA:  .WORD   0       ;BUS ADDRESS REG ADDR
2533  011546  000000                    DXBCA:  .WORD   0       ;BYTE COUNT REG ADDR
2534  011550  000000                    DXMOA:  .WORD   0       ;MAINTENANCE OUT REG ADDR.
2535  011552  000000                    DXMIA:  .WORD   0       ;MAINTENANCE IN REG ADDR
2536  011554  000000                    DXCBA:  .WORD   0       ;CONTROL BITS REG ADDR
2537  011556  000000                    DXNDA:  .WORD   0       ;NPR DATA REG ADDR
2538  011560  000000                    DXESA:  .WORD   0       ;EXTRA SIGNALS REG ADDR
2539  011562  000000                    DXMOBA: .WORD   0       ;MAINTENANCE OUT BUFFERED REG ADDR
2540  011564  000000                    DXES1A: .WORD   0       ;EXTRA EXTRA SIGNALS REG ADDR.
2541                                    ;       REGS TO BE OUTPUT TO DX
2542  011566  000000                    DXDSO:  .WORD   0       ;DEVICE STATUS REG OUT
2543  011570  000000                    DXCAO:  .WORD   0       ;COMMAND AND ADDR REG OUT
2544  011572  000000                    DXCSO:  .WORD   0       ;CONTROL UNIT STATUS REG OUT
2545  011574  000000                    DXOSO:  .WORD   0       ;OFFSET AND STATUS REG OUT
2546  011576  000000                    DXBAO:  .WORD   0       ;BUS ADDR REG. OUT
2547  011600  000000                    DXMOO:  .WORD   0       ;MAINT OUT REG ADDR
2548  011602  000000                    DXBCO:  .WORD   0
2549
```

```
 2550                                   ;
 2551                                   ;
 2552                                   ;       DEFAULT PARAMETER TABLE
 2553  011604                           DPAR:
 2554  011604  000001                   DTESTN: .WORD   1       ;STARTING TEST NUMBER
 2555  011606  176200                   DUBA:   .WORD   176200  ;DEFAULT UNIBUS BASE ADDR
 2556  011610  000300                   DVECT:  .WORD   300     ;DEFAULT VECTOR ADDR
 2557  011612  000004                   DPRIOR: .WORD   4       ;DEFAULT DX PRIORITY LEVEL
 2558
```

```
 2559                                   ;
 2560                                   ;
 2561                                   ;       INPUT PARAMETER LIMITS TABLE
 2562                                   ;
 2563                                   ;
 2564   011614                          LPAR:
 2565   011614  000001                  LTESTN: 1               ;LOWEST POSSIBLE TEST NUMBER
 2566   011616  000004                  HTESTN: 4               ;HIGHEST POSSIBLE TEST NUMBER
 2567   011620  160000                  LUBA:   160000          ;LOWEST POSSIBLE UNIBUS ADDRESS
 2568   011622  177756                  HUBA:   177756          ;HIGHEST POSSIBLE UNIBUS ADDRESS
 2569   011624  000040                  LVEC:   40              ;LOWEST POSSIBLE DX VECTOR ADDRESS
 2570   011626  000776                  HVEC:   776             ;HIGHEST POSSIBLE DX VECTOR ADDRESS
 2571   011630  000000                  LPRIOR: 0               ;LOWEST POSSIBLE FOR LEVEL
 2572   011632  000006                  HPRIOR: 6               ;HIGHEST POSSIBLE FOR LEVEL
```

```
                                        ;
                                        ;
                                        ;       ACTIVE PARAMETER TABLE
                                        ;
                                        ;       THIS TABLE CONTAINS CONFIGURATION PARAMETER CURRENTLY
                                        ;       IN USE.
                                        ;
011634                                  APAR:
011634  000000                          ATESTN: .WORD   0       ;STARTING TEST NO.
011636  000000                          AUBA:   .WORD   0       ;DX UNIBUS BASE ADDR.
011640  000000                          AVECT:  .WORD   0       ;DX VECTOR ADDRESS
011642  000000                          APRIOR: .WORD   0       ;DX PRIORITY LEVEL
011644  000000                          MSIZE:  .WORD   0       ;ADDR OF LAST WORD OF MEM, NOT INCLUDING LOADERS, KT OFF
011646  000000                          MMAX:   .WORD   0       ;LAST WORD OF MEM, KT OFF
011650  000000                          EASIZE: .WORD   0       ;LOWEST INVALID SETTING OF PAR
011652  000000                          KT:     .WORD   0       ;=0 NO KT IN USE =1 KT IN USE
011654  000000                          FIRSTP: .WORD   0       ;0 IF FIRST PASS, 1 IF NOT FIRST PASS
011656  177564                          TPS:    .WORD   177564  ;TTY PRINTER STATUS REG ADDR
011660  177566                          TPB:    .WORD   177566  ;TTY PRINTER BUFFER REG ADDR
011662  177560                          TKS:    .WORD   177560  ;TTY KEYBOARD STATUS REG ADDR
011664  177562                          TDB:    .WORD   177562  ;TTY KEYBOARD BUFFER.
011666  000000                          CTESTN: .WORD   0       ;CURRENT TEST NUMBER
011670  000000                          PARMV:  .WORD   0       ;0 IF PARMS NOT VALID, 1 IF THEY ARE VALID
011672  000001                          COUNT:  .WORD   1       ;ITERATION COUNT CONSTANT
011674  000000                          CNT:    .WORD   0       ;ITERATION VARIABLE
011676  000000                          ERR:    .WORD   0       ;IF ERROR DETECTED, CONTAINS TN
011700  000000                          LOCK:   .WORD   0       ;SW=TN IF LOOP SW 14 ERR DETECTED
011702  000000                          LOCKAD: .WORD   0       ;LOCK ON ERROR ADDR
011704  000000                          EABITS: .WORD   0       ;HIGHEST VALID SETTING OF ADDR 17 16
011706  000000                          PHYMAX: .WORD   0       ;LOWEST INVALID PHYSICAL ADDR, LOW 16 BITS
011710  000000                          BUFBAS: .WORD   0       ;START OF CURRENT WORKING BUFFER, VIRTUAL
011712  000000                          BUFSZE: .WORD   0       ;SIZE OF CURRENT BUFFER IN OCT BYTES
011714  000000                          BUFMAX: .WORD   0               ;MAX VIRTUAL ADDR FOR THIS KT SETTING
011716  000000                          EACUR:  .WORD   0               ;CURRENT SETTING OF EA BITS
011720  000000                          DATAD:  .WORD   0       ;CURRENT DATA ADDR
011722  000000                          BFBAPH: .WORD   0       ;CURRENT PHYSICAL BUFFER BASE ADDR
011724  000000                          BFADPH: .WORD   0               ;CURRENT PHYSICAL DATA ADDR
011726  000000                          SPWPHY: .WORD   0       ;PHYSICAL ADDR OF SPW START
011730  000000                          DEV:    .WORD   0       ;CURRENT IBM DEVICE NO.
011732  000400                          CMD:    .WORD   400     ;CURRENT IBM COMMAND WITH PARITY BIT
011734  000000                          ERRPC:  .WORD   0       ;PC WHERE ERROR WAS DETECTED
011736  000000                          ERRID:  .WORD   0               ;ERROR NUMBER
011740  000000                          ERRMS:  .WORD   0               ;ADDR OF UNIQUE ERROR MESSAGE
011742  000000                          EAERR:  .WORD   0               ;EABITS ON ERROR
011744  000000                          RECORD: .WORD   0               ;SWITCH TO CONTROL DEV REC RECORDING
011746  000000                          FRSTD:  .WORD   0               ;FIRST DEVICE RECOGNIZED SW
011750  000000                          FRSTDV: .WORD   0               ;FIRST DEVICE RECOGNIZED NUMBER
011752  000000                          DEVCT:  .WORD   0               ;FIRST DEVICE GROUP COUNT
011754  000000                          DEVMSS: .WORD   0               ;DEVICE MESSAGE PRINTED SW
011756  000000                          ACTU:   .WORD   0       ;ACTUAL DATA
011760  000000                          EXP:    .WORD   0               ;EXPECTED DATA
011762  000000                          DEVFND: .WORD   0               ;= 1 IF DEVICE EVER RECOGNIZED
011764  000000                          SPWMVD: .WORD   0       ;- 1 IF SPW TAPLE IN ALTERNATE LOCATION
```

```
 2625                           .NLIST  BIN
 2626  011766  ERR1:    .ASCIZ  /FAST NPR DATA ERROR/
 2627  011774
 2628  012002
 2629  012010
 2630  012012  ERR2:    .ASCIZ  /NO DX INTERRUPT/
 2631  012020
 2632  012026
 2633
 2634  012032  ERR3:    .ASCIZ  /DX ADDRESSING FRROR, DATA XFER/
 2635  012040
 2636  012046
 2637  012054
 2638  012062
 2639  012070
 2640  012071  ERR4:    .ASCIZ  /SPW ACCESS, ADDR ERROR/
 2641  012076
 2642  012104
 2643  012112
 2644  012120  ERR5:    .ASCIZ  /NO IBM DEV ADR RECOGNIZED. CHECK A20 MODULE/
 2645  012126
 2646  012134
 2647  012142
 2648  012150
 2649  012156
 2650  012164
 2651  012172
 2652  012174  ERR6:    .ASCIZ  /DST ACCESS ERROR/
 2653  012202
 2654  012210
 2655  012215  T4MSG:   .ASCIZ  /TST3 MUST BE RUN FIRST FOR TST 4 TO WORK/
 2656  012222
 2657  012230
 2658  012236
 2659  012244
 2660  012252
 2661  012260
 2662  012266  HEADER: .ASCII  'CZDXJB0  DX11B MEMORY ADDRESSING TEST'<15><12>
 2663  012274
 2664  012302
 2665  012310
 2666  012316
 2667  012324
 2668  012332
 2669  012335          .ASCII  'TYPE:  <D>, DEFAULT PARAMETERS'<15><12>
 2670  012342
 2671  012350
 2672  012356
 2673  012364
 2674  012372
<12675  012374          .ASCII  '       <P>, PREVIOUS PARAMETERS'<15>
 2676  012402
 2677  012410
 2678  012416
 2679  012424
 2680  012427          .ASCII  '       <S>, SELECT PARAMETERS'<15><12>
```

```
 2681  012434
 2682  012442
 2683  012450
 2684  012456
 2685  012460          .ASCIZ  '       <N>, START AT THIS TEST NO.'<15><12>
 2686  012466
 2687  012474
 2688  012502
 2689  012510
 2690  012516
 2691  012517  INVM:   .ASCIZ  'INVALID ENTRY'
 2692  012524
 2693  012532
 2694  012535  INVPRM: .ASCIZ  'NEED MORE PARMS'
 2695  012542
 2696  012550
 2697  012555  PROMPT: .ASCIZ  'D,P,S,N? '
 2698  012562
 2699  012567  TESTN:  .ASCIZ  'STARTING TEST NO. '
 2700  012574
 2701  012602
 2702  012610
 2703  012612  BASEA:  .ASCIZ  'BASE ADDRESS '
 2704  012620
 2705  012626
 2706  012630  VECTA:  .ASCIZ  'VECTOR ADDRESS '
 2707  012636
 2708  012644
 2709  012650  PRILV:  .ASCIZ  'DX PRIORITY LEVEL '
 2710  012656
 2711  012664
 2712  012672
 2713  012673  SETSW:  .ASCIZ  'SET SWITCHES '
 2714  012700
 2715  012706
 2716  012711  BELL:   .ASCIZ  <207> 'END PASS'
 2717  012716
 2718  012723  BADP:   .ASCIZ  'INPUT PARAMETER OUTSIDE LIMITS'
 2719  012730
 2720  012736
 2721  012744
 2722  012752
 2723  012760
 2724  012762  ERPC:   .ASCIZ  'ERROR PC: '
 2725  012770
 2726  012775  ERNO:   .ASCIZ  '  ERR NO.: '
 2727  013002
 2728  013010
 2729  013011  ERTN:   .ASCIZ  'TEST NO.: '
 2730  013016
 2731  013024  ERADR:  .ASCIZ  'VIRT ADDR OF BUFFER BASE: '
 2732  013032
 2733  013040
 2734  013046
 2735  013054
 2736  013057  ERADPH: .ASCIZ  'PHY ADDR OF BUFFER: '
```

```
 2737  013064
 2738  013072
 2739  013100
 2740  013104  EABIT:   .ASCIZ  'EA BITS IN OCT: '
 2741  013112
 2742  013120
 2743  013125  DATAA:   .ASCIZ  'VIRT ADDR OF ACCESS ERROR: '
 2744  013132
 2745  013140
 2746  013146
 2747  013154
 2748  013161  DEVMS:   .ASCIZ  '1ST IBM DEV RCGNZD.  OCTAL:  '
 2749  013166
 2750  013174
 2751  013202
 2752  013210
 2753  013216
 2754  013217  DEVCMS:  .ASCIZ  'SIZE OF 1ST DEV GROUP:  '
 2755  013224
 2756  013232
 2757  013240
 2758  013246
 2759  013250  ACTUMS:  .ASCIZ  'ACTUAL:  '
 2760  013256
 2761  013262  EXPMS:   .ASCIZ  '  EXPECTED:  '
 2762  013270
 2763  013276
 2764  013300  CRLF1:   .ASCIZ  / /<15><12>
 2765                                .LIST   BIN
 2766                                .EVEN
 2767                                ;INBUF MUST BE ON AN EVEN WORD BOUNDARY
 2768  013304     012         INBUF: .BYTE   12      ;KEYBOARD INPUT BUFFER
 2769             014000              .=14000
 2770  014000                 SPW:                   ;START OF STATIONARY SPW TABLE USED ONLY TO
 2771                                               ;FIELD UNEXPECTED ERRORS.
 2772  014000  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2773  014002  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2774  014004  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2775  014006  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2776  014010  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2777  014012  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2778  014014  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2779  014016  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2780  014020  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2781  014022  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2782  014024  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2783  014026  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2784  014030  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2785  014032  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2786  014034  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2787  014036  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2788  014040  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2789  014042  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2790  014044  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2791  014046  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2792  014050  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
```

```
 2793  014052  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2794  014054  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2795  014056  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2796  014060  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2797  014062  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2798  014064  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2799  014066  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2800  014070  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2801  014072  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2802  014074  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2803  014076  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2804  014100  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2805  014102  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2806  014104  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2807  014106  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2808  014110  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2809  014112  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2810  014114  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2811  014116  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2812  014120  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2813  014122  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2814  014124  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2815  014126  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2816  014130  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2817  014132  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2818  014134  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2819  014136  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2820  014140  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2821  014142  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2822  014144  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2823  014146  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2824  014150  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2825  014152  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2826  014154  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2827  014156  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2828  014160  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2829  014162  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2830  014164  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2831  014166  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2832  014170  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2833  014172  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2834  014174  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2835  014176  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2836  014200  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2837  014202  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2838  014204  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2839  014206  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2840  014210  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2841  014212  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2842  014214  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2843  014216  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2844  014220  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2845  014222  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2846  014224  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2847  014226  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2848  014230  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
```

```
 2849  014232  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2850  014234  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2851  014236  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2852  014240  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2853  014242  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2854  014244  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2855  014246  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2856  014250  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2857  014252  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2858  014254  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2859  014256  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2860  014260  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2861  014262  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2862  014264  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2863  014266  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2864  014270  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2865  014272  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2866  014274  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2867  014276  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2868  014300  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2869  014302  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2870  014304  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2871  014306  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2872  014310  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2873  014312  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2874  014314  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2875  014316  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2876  014320  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2877  014322  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2878  014324  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2879  014326  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2880  014330  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2881  014332  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2882  014334  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2883  014336  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2884  014340  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2885  014342  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2886  014344  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2887  014346  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2888  014350  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2889  014352  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2890  014354  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2891  014356  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2892  014360  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2893  014362  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2894  014364  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2895  014366  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2896  014370  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2897  014372  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2898  014374  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2899  014376  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2900  014400  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2901  014402  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2902  014404  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2903  014406  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 2904  014410  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
```

```
2905  014412  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2906  014414  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2907  014416  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2908  014420  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2909  014422  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2910  014424  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2911  014426  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2912  014430  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2913  014432  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2914  014434  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2915  014436  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2916  014440  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2917  014442  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2918  014444  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2919  014446  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2920  014450  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2921  014452  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2922  014454  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2923  014456  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2924  014460  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2925  014462  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2926  014464  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2927  014466  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2928  014470  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2929  014472  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2930  014474  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2931  014476  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2932  014500  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2933  014502  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2934  014504  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2935  014506  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2936  014510  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2937  014512  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2938  014514  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2939  014516  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2940  014520  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2941  014522  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2942  014524  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2943  014526  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2944  014530  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2945  014532  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2946  014534  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2947  014536  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2948  014540  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2949  014542  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2950  014544  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2951  014546  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2952  014550  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2953  014552  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2954  014554  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2955  014556  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2956  014560  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2957  014562  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2958  014564  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2959  014566  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2960  014570  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
```

```
2961  014572  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2962  014574  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2963  014576  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2964  014600  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2965  014602  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2966  014604  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2967  014606  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2968  014610  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2969  014612  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2970  014614  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2971  014616  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2972  014620  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2973  014622  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2974  014624  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2975  014626  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2976  014630  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2977  014632  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2978  014634  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2979  014636  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2980  014640  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2981  014642  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2982  014644  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2983  014646  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2984  014650  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2985  014652  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2986  014654  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2987  014656  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2988  014660  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2989  014662  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2990  014664  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2991  014666  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2992  014670  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2993  014672  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2994  014674  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2995  014676  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2996  014700  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2997  014702  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2998  014704  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2999  014706  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3000  014710  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3001  014712  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3002  014714  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3003  014716  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3004  014720  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3005  014722  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3006  014724  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3007  014726  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3008  014730  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3009  014732  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3010  014734  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3011  014736  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3012  014740  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3013  014742  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3014  014744  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3015  014746  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3016  014750  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
```

```
 3017  014752  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3018  014754  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3019  014756  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3020  014760  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3021  014762  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3022  014764  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3023  014766  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3024  014770  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3025  014772  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3026  014774  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3027  014776  000002                          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
 3028  015000                          TT:                     ;START OF STATIONARY TUMBLE TABLE
 3029  016000                          ENDALL:
 3030          000001                          .END
```

```
ACTU    011756          2502*   2507    2621#
ACTUMS  013250          2504    2759#
ADRO  = 004000           608#   2084    2091    2244    2246
APAR    011634          2579#
APRIOR  011642           814*    832*    870*   2583#
ATESTN  011634           728     811*    829*    849*    885*   2580#
AUBA    011636           812*    830*    856*    901    2581#
AVECT   011640           813*    831*    863*   1835    1978    2582#
BADP    012723          1520    2718#
BASEA   012612           963    2703#
BELL    012711           774    2716#
BFADPH  011724          1984*   1988    2009*   2608#
BFBAPH  011722          1843    1984    2078    2235    2273    2310*   2332*   2333*   2354    2362    2371    2378    2471
                        2607#
BIT0  = 000001           552#
BIT00 = 000001           542#    552
BIT01 = 000002           541#    551
BIT02 = 000004           540#    550
BIT03 = 000010           539#    549
BIT04 = 000020           538#    548
BIT05 = 000040           537#    547
BIT06 = 000100           536#    546
BIT07 = 000200           535#    545
BIT08 = 000400           534#    544
BIT09 = 001000           533#    543
BIT1  = 000002           551#
BIT10 = 002000           532#
BIT11 = 004000           531#
BIT12 = 010000           530#
BIT13 = 020000           529#
BIT14 = 040000           528#
BIT15 = 100000           527#
BIT2  = 000004           550#
BIT3  = 000010           549#
BIT4  = 000020           548#
BIT5  = 000040           547#
BIT6  = 000100           546#
BIT7  = 000200           545#
BIT8  = 000400           544#
BIT9  = 001000           543#
BPTVEC= 000014           559#
BUFAD   010300          1832    1873    1976    2015    2067    2111    2210    2269    2298#
BUFBAS  011710          1855    1909    1983    2072    2215    2309*   2319*   2320*   2322    2330*   2331*   2340    2347*
                        2348*   2349    2383    2463   2602#
BUFCLR  006510          1840    1872    1909#   1982    2014    2070    2110    2213    2268
BUFMAX  011714          2011    2324*   2353*   2383*   2384*   2385   2604#
BUFSZE  011712          1838*   1856    1910    1981*   2069*   2212*  2603#
CHECKP  004766           850     857     864     871     886    1489#
CKSWR = 104406          1766#
CMD     011732          2066*   2093    2095    2238*   2240    2248    2250    2252    2257    2265*   2266    2497    2501
                        2611#
CMDO  - 002000           606#   1891    1892    2094    2096    2249    2251
CNT     011674           727*    751*   2596#
COUNT   011672           727    2595#
CR      000015           467#    944    1625    1635
CRLF  - 000200           468#   1596    1635
```

```
CRLF1   013300          1551    2516    2764#
CTESTN  011666           728*    731     748*    749     2410    2423    2455    2593#
DATAA   013125          2486    2743#
DATAD   011720          1983*   1999    2005*   2010*   2011    2489    2606#
DDISP = 177570           474#    659     712
DEV     011730          2071*   2073    2074*   2075    2077*   2081    2082    2083    2092    2097    2107*   2108    2161
                        2206*   2207*   2209    2221    2242    2243    2247    2493    2610#
DEVCMS  013217           764    2754#
DEVCT   011752           755     767    2064*   2115    2163*   2169    2619#
DEVFND  011762          2162*   2200    2623#
DEVMS   013161           758    2748#
DEVMSS  011754           725*    753     769*   2620#
DISPLA  001142           659#    712*    720*
DISPRE  000174           616#    720
DONE  = 000200           602#   1864    1941    2006
DONTSV  007620          2105    2169#
DPAR    011604          2553#
DPRIOR  011612           814     832     991    2557#
DSWR  = 177570           473#    658     711
DTESTN  011604           811     829     958    2554#
DUBA    011606           812     830     969    2555#
DVECT   011610           813     831     980    2556#
DXBAA   011544          1843*   1988*   2532#
DXBAO   011576          2546#
DXBCA   011546          1882*   2533#
DXBCO   011602          2548#
DXCAA   011536          2082*   2242*   2529#
DXCAO   011570          2543#
DXCBA   011554          1950    2085    2087    2536#
DXCSA   011540          1842*   1864*   1866*   1887*   1941*   1942*   1943*   1944*   1985*   2006*   2008*   2079*   2241*
                        2530#
DXCSO   011572          2544#
DXDSA   011534           900    2528#
DXDSO   011566          2542#
DXESA   011560          1865*   1893*   1940*   2007*   2538#
DXES1A  011564          2540#
DXGO    006366          1844    1881#   1989
DXMIA   011552          2535#
DXMOA   011550          1881*   1889*   1890*   1891*   1892*   2083*   2084*   2090*   2091*   2092*   2093*   2094*   2095*
                        2096*   2102*   2103*   2243*   2244*   2245*   2246*   2247*   2248*   2249*   2250*   2251*   2262*
                        2263*   2534#
DXMOBA  011562          2539#
DXMOO   011600          1841*   1857    1867    1869*   1881    1986*   2547#
DXNDA   011556          2537#
DXOSA   011542          1946*   1949*   2078*   2097    2224    2254    2257    2374*   2380*   2502    2531#
DXOSO   011574          2545#
DXRES   006560          1831    1940#   1953    1975    2065    2106    2199    2264
EABIT   013104          2474    2740#
EABITS  011704          1365*   1383*   1384*   1385*   1386*   2336    2356    2600#
EACUR   011716          1842    1985    2079    2229    2241    2302*   2335*   2336    2356    2367    2476    2605#
EAERR   011742          2476*   2477*   2478*   2479*   2481    2615#
EASIZE  011650          1364*   1373*   1374*   2343    2586#
EMTVEC= 000030           562#
ENDALL  016000          2304    3029#
ENDB    010414          2313    2322#
ENDB1   010750          2325    2346    2382    2386    2388#
```

```
ENDPRM  002270            808     908#
ENDTST  001410            740#   1877    2019    2122    2277
ENDT1   006350           1873#
END1    001576            777     783#
END4    010274           2205    2272    2277#
EOP     001452            753#
ERADPH  013057           2468    2736#
ERADR   013024           2460    2731#
ERNO    012775           2440    2726#
ERPC    012762           2433    2724#
ERR     011676            730*   1932*   2410*   2597#
ERRID   011736           2427*   2443    2457    2483    2491    2495    2613#
ERRMS   011740           2428*   2445    2614#
ERRPC   011734           2426*   2435*   2437    2612#
ERRR    010752           1859    1899    2002    2099    2118    2259    2406#
ERRVEC= 000004            555#    709     710*    721*   1409    1410    1414*  1419*  1431*  1436*  1445*  1455*  1456*
ERR1    011766           1861    2626#
ERR2    012012           1901    2630#
ERR3    012032           2004    2634#
ERR4    012071           2101    2640#
ERR5    012120           2121    2644#
ERR6    012174           2261    2652#
ERTN    013011           2452    2729#
EXP     011760           2493*   2499*   2501*   2513    2622#
EXPMS   013262           2510    2761#
FIRSTP  011654            724*    806     907*   2588#
FRSTD   011746           2063*   2158    2160*   2617#
FRSTDV  011750            761    2161*   2207    2618#
GETPRM  001600            726     806#
GNS     = ****** U        615    1759    1760    1761    1762    1764    1766    1767    1768    1769
GO      001346            728#    752
GO1     006206           1842#   1870
GTSWR   = 104405          1764#
HEADER  012266            822    2662#
HLDO    = 040000          605#   1889    1890    2090    2245
HPRIOR  011632           1515    2572#
HT      = 000011          465#   1594    1635
HTESTN  011616            749    1494    2566#
HUBA    011622           1501    2568#
HVEC    011626           1508    2570#
INASC   002532            878     940     954     965     976     987    1012#
INBUF   013304           1017*   1023*   1024    2768#
INIT    006530           1830    1927#   1974    2061    2198
INPRM   002272            824     846     854     861     868     883     933#
INTERR  006506           1849    1895*   1902*   1904#   1934*   1993
INTR1   006230           1836    1849#
INTR2   006766           1979    1993#
INVM    012517            840    2691#
INVPRM  012535            893    2694#
IOTVEC= 000020            560#
KIPAR0= 172340            593#   1395    1423
KIPAR1= 172342            594#
KIPAR2= 172344            595#
KIPAR3= 172346            596#
KIPAR4= 172350            597#
KIPAR5= 172352            598#   2232*
```

```
KIPAR6= 172354            599#   2315*  2318*  2342*  2343
KIPAR7= 172356            600#   1439
KIPDR0= 172300            582#
KIPDR1= 172302            583#
KIPDR2= 172304            584#
KIPDR3= 172306            585#
KIPDR4= 172310            586#
KIPDR5= 172312            587#
KIPDR6= 172314            588#
KIPDR7= 172316            589#
KT      011652            770    1353*  1372*  2227   2312   2338   2465   2587#
LF    = 000012            466#   1629   1635
LOCK    011700            729*   2415   2423*  2425*  2598#
LOCKAD  011702           1933*   2419   2599#
LOOP1   006166           1838#   1876
LOOP2   006710           1981#   2018
LOOP3   007176           2071#   2114
LOOP4   007730           2214#   2276
LOP2    006736           1985#   2012
LOP3    007232           2078#   2109
LOP4    010044           2239#   2267
LPAR    011614           2564#
LPRIOR  011630           1513    2571#
LTESTN  011614           1492    2565#
LUBA    011620           1499    2567#
LVEC    011624           1506    2569#
MMAX    011646           1354*   2371   2585#
MMVEC = 000250            571#
MSIZE   011644           1355*   1358*  1363*  2369   2385   2584#
NEXTB   010430           2301    2330#  2352   2364   2373
PARMV   011670            818     836    845*   890    906*   2594#
PHYMAX  011706           1375*   1376*  1377*  1378*  1379*  1380*  1381*  1382*  2358   2601#
PIRQ  = 177772            472#
PIRQVE= 000240            566#
PRILV   012650            985    2709#
PRINT   005134            757     760    763    766    773    821    839    875    892    937    951    962    973
                          984    1519   1543#  2202   2432   2436   2439   2442   2446   2451   2454   2459   2462
                         2467    2470   2473   2480   2485   2488   2503   2506   2509   2512   2515
PROMPT  012555            938    2697#
PR0   = 000000            489#
PR1   = 000040            490#
PR2   = 000100            491#
PR3   = 000140            492#
PR4   = 000200            493#
PR5   = 000240            494#
PR6   = 000300            495#
PR7   = 000340            496#
PS    = 177776            469#    470
PSW   = 177776            470#
PTYGEN  007514           2080    2139#  2208   2239
PTY2    007522           2142#   2146
PTY3    007530           2143    2145#
PTY4    007546           2148    2151#
PTY5    007552           2150    2152#
PWRVEC= 000024            561#    705*   706*   1775*  1776*  1785*  1791*  1803*  1804*
RDCHR   104407           1229    1767#
```

```
RDLIN = 104410          1016    1304    1768#
RDOCT = 104411          1022    1769#
RECORD  011744          2062*   2156    2171*   2616#
RESTAR  001334           726#    783    1808
RESVEC= 000010           556#
SAVDEV  007554          2089    2156#
SELO  = 020000           604#   1889    1890    2090    2245
SETSW   012673           876    2713#
SIZE    004212           898    1351#
SOSIEN= 000004           603#   1865    1893    2007
SPW     014000          1946    1949    2374    2376    2378    2380    2770#
SPWMVD  011764          1947    2375*   2381*   2624#
SPWPHY  011726          2609#
SRVO  = 001000           607#   2102    2103    2262    2263
SR0   = 177572           575#    772*   1420    1435*   1442*   2321*
SR1   = 177574           576#
SR2   = 177576           577#
SR3   = 172516           578#   1432*
STACK = 001100           460#    701
START   001164           619     694#   1070    1153
STKLMT= 177774           471#
SWR     001140           658#    699     711*    713     719*    746     1073    1112    1167*   1366    1783    1796*   2406
                        2417    2421    2429    2449
SWREG   000176           617#    719    1073    1112    1135
SW0   = 000001           524#
SW00  = 000001           514#    524
SW01  = 000002           513#    523
SW02  = 000004           512#    522
SW03  = 000010           511#    521
SW04  = 000020           510#    520
SW05  = 000040           509#    519
SW06  = 000100           508#    518
SW07  = 000200           507#    517
SW08  = 000400           506#    516
SW09  = 001000           505#    515
SW1   = 000002           523#
SW10  = 002000           504#   1366
SW11  = 004000           503#
SW12  = 010000           502#   2449
SW13  = 020000           501#   2429
SW14  = 040000           500#   2417    2421
SW15  = 100000           499#   2406
SW2   = 000004           522#
SW3   = 000010           521#
SW4   = 000020           520#
SW5   = 000040           519#
SW6   = 000100           518#
SW7   = 000200           517#
SW8   = 000400           516#
SW9   = 001000           515#
TBITVE= 000014           557#
TDB     011664          2592#
TESTN   012567           952    2699#
TKS     011662          2591#
TKVEC = 000060           564#   1050*   1051*
TPB     011660          2590#
```

```
TPS      011656           2589#
TPVEC  = 000064            565#
TRAPVE = 000034            563#    703*    704*
TRTVEC = 000014            558#
TST      001364            731#    747     750
TSTAB    001376            733     734#
TST1     006134            735    1830#
TST2     006656            736    1974#
TST3     007124            737    2061#
TST4     007636            738    2198#
TT       015000           3028#
TYPE   = 104401           1067    1079    1133    1134    1137    1148    1158    1169    1188    1236    1242    1247    1251
                          1256    1257    1259    1262    1266    1333    1335    1550    1558    1599    1696    1759#   1805
TYPOC  = 104402           1136    1555    1760#
TYPON  = 104404           1762#
TYPOS  = 104403           1761#
T4MSG    012215           2203    2655#
VECTA    012630            974    2706#
$AUTOB   001134            655#    1126    1284
$BDADR   001122            650#
$BDDAT   001126            652#
$BELL    004004           1079    1276#
$CHARC   005422           1601*   1611*   1618    1627*   1632#
$CKSWR   003106           1112#   1766
$CMTAG   001100            638#     696     697
$CM3   = 000000            668#
$CNTLC   004010           1067    1148    1277#
$CNTLG   004022           1133    1279#
$CNTLU   004015           1158    1251    1278#
$CORE    004666           1418    1445#
$CRLF    001161            669#    1169    1256    1276    1338    1600    1635
$CROUT   004716           1445    1452#
$ENDAD   001566            626     779#     816    2518
$ERFLG   001103            641#
$ERMAX   001115            647#
$ERRPC   001116            648#
$ERRTB   001164            686#
$ERTTL   001112            645#
$FILLC   001156            666#    1604    1635
$FILLS   001155            665#    1635
$GDADR   001120            649#
$GDDAT   001124            651#
$GTSWR   003176           1134#   1764
$HD    = 000003            455     456
$HIOCT   004210           1326*   1337#
$ICNT    001104            642#
$ILLUP   006116           1775    1791    1810#
$INTAG   001135            656#   1131*   1150    1170    1284
$ITEMB   001114            646#
$KTNEX   004660           1419    1444#
$KTOUT   004650           1436    1441#
$KT11    004516           1351*   1368*   1370    1417#   1421*   1444*
$LF      001162            670#   1266    1276    1338    1635
$LPADR   001106            643#
$LPERR   001110            644#
$LSTAD   004762           1354    1355    1356    1457*   1464#
```

```
$LSTBK  004764            1373    1375    1383    1458*   1465#
$MAIL = ****** U           723    1588
$MNEW   004040            1137    1282#
$MSWR   004027            1134    1280#
$NULL   001154             664#   1606    1635
$OCNT   005650            1668*   1697*   1710#
$OMODE  005652            1663*   1667*   1672    1675*   1686*   1712#
$PASS   001100             639#
$POWER  006124            1806    1813#
$PWRAD  006112            1808#
$PWRDN  005752             705    1775#   1803
$PWRMG  006106            1806#
$PWRUP  006024            1785    1791#
$QUES   001160             668#   1188    1259    1276    1335    1338    1635
$RDCHR  003450            1201#   1767
$RDDEC= ****** U          1770
$RDLIN  003540            1224#   1768
$RDOCT  004052            1299#   1769
$RDSZ = 000010            1217#
$R2A  = ****** U          1770
$SAVRE= ****** U          1770
$SAVR6  006122            1784*   1792    1793*   1794*   1812#
$SETUP= 000114             694#    702     703     705     707    1071    1076    1077    1107    1284
$SIZE   004460            1352    1369    1405#
$SIZEX  004722            1443    1453#
$STUP = 177777             694#
$SVPC = 000204             624#    629
$SWR  = 160000             455     456#    668    1809
$TKB    001146             661#   1032    1052    1063    1088    1116    1143
$TKCNT  002602            1033#   1047*   1077    1094*   1208    1210*
$TKINT  002620            1012    1047#   1068    1129
$TKQEN= 002620            1037#   1102    1213
$TKQIN  002604            1034#   1048*   1049    1100*   1101*   1102    1104*
$TKQOU  002606            1035#   1049*   1211    1212*   1213    1215*
$TKQSR  002610            1036#   1048    1104    1215
$TKS    001144             660#   1032    1053*   1084*   1086    1092*   1114    1130*   1140    1152*   1172*
$TKSRV  002670            1050    1063#
$TN   = 000001             455#
$TPB    001152             663#   1624*   1635
$TPFLG  001157             667#   1582    1635
$TPS    001150             662#   1622    1635
$TRAP   005654             703    1732#
$TRAP2  005714            1747#   1758
$TRP  = 000012            1751#   1760#   1761#   1762#   1763#   1764    1765#   1766    1767#   1768#   1769#   1770#
$TRPAD  005726            1741    1758#
$TSTNM  001102             640#
$TTYIN  003774            1226    1227    1239    1257    1271    1275#
$TYPBN= ****** U          1763
$TYPDS= ****** U          1763
$TYPE   005206            1582#   1751    1759
$TYPEC  005356            1174    1603    1610    1617    1622#   1623
$TYPEX  005424            1628    1630    1633#
$TYPOC  005452            1666#   1760
$TYPON  005466            1665    1668#   1762
$TYPOS  005426            1661#   1761
$OFILL  005651            1662*   1666*   1676    1711#
```

```
.     = 016000            611#   615#   624    625#   627#   629#   630#   637#   671    700    1032   1036#  1037
                          1038   1275#  1276   1277   1284#  1338   1635   1787   1811   2769#
```

```
CKSWR     445#
COMMEN      1#    567#
ENDCOM      1#    567#
ERROR     461#
ESCAPE      1#    567#
GETPRI      1#    567#   1413
GETSWR      1#    445#    567#
MULT        1#    567#
NEWTST      1#    567#
POP         1#    445#    567#   1023    1327    1560    1796    1797
PUSH        1#    445#    567#   1301    1543    1777    1783
REPORT      1#    567#
SCOPE     462#
SETPRI      1#    445#    567#    740    1204    1883    1927    2411
SETTRA   1751#   1760    1761    1762    1764    1766    1767    1768    1769
SETUP       1#    567#    694
SKIP        1#    567#
SLASH       1#    567#
SPACE     567#
STARS       1#    567#    622     633     671    1031    1107    1122    1193    1217    1287    1391    1567    1638    1726
         1773   1789
SWRSU       1#    567#    707#
TRMTRP   1751#
TYPBIN      1#    567#
TYPDEC      1#    567#
TYPNAM      1#    567#
TYPNUM      1#    567#
TYPOCS      1#    567#
TYPOCT      1#    567#   1135
TYPTXT      1#    567#
$$CMRE    631#
$$CMTM    631#
$$ESCA      1#    567#
$$NEWT      1#    567#
$$SET    1751#   1760    1761    1762    1764    1766    1767    1768    1769
$$SKIP      1#    567#
.EQUAT      1#    445#    457
.HEADE      1#    445#
.KT11       1#    445#    567
.SETUP      1#    445#    694
.SWRHI      1#
.$ACT1      1#    445#    620
.$APTB      1#
.$APTH      1#
.$APTY      1#
.$ASTA      1#
.$CATC      1#    445#    609
.$CMTA      1#    445#    631
.$DB2D      1#
.$DB2O      1#
.$DIV       1#
.$EOP       1#
.$ERRO      1#
.$ERRT      1#
.$MULT      1#
.$POWE      1#    445#   1771
```

```
.$RAND      1#
.$RDDE      1#
.$RDOC      1#     445#   1285
.$READ      1#     445#   1029
.$R2AZ      1#
.$SAVE      1#
.$SB2D      1#
.$SB20      1#
.$SCOP      1#
.$SIZE      1#     445#   1389
.$SUPR      1#
.$TRAP      1#     445#   1724
.$TYPB      1#
.$TYPD      1#
.$TYPE      1#     445#   1565
.$TYPO      1#     445#   1636
.$40CA      1#
.1170       1#


. ABS.  016000     000


ERRORS DETECTED:  0

CZDXJB.BIN,CZDXJB.SEQ/CRF/SOL=CZDXJB.SML,CZDXJB.P11
RUN-TIME: 15 13 .8 SECONDS
RUN-TIME RATIO: 194/29=6.6
CORE USED:  32K  (63 PAGES)
```