DX11-B

ADDRESSING TEST

MD-11-DZDXJ-A

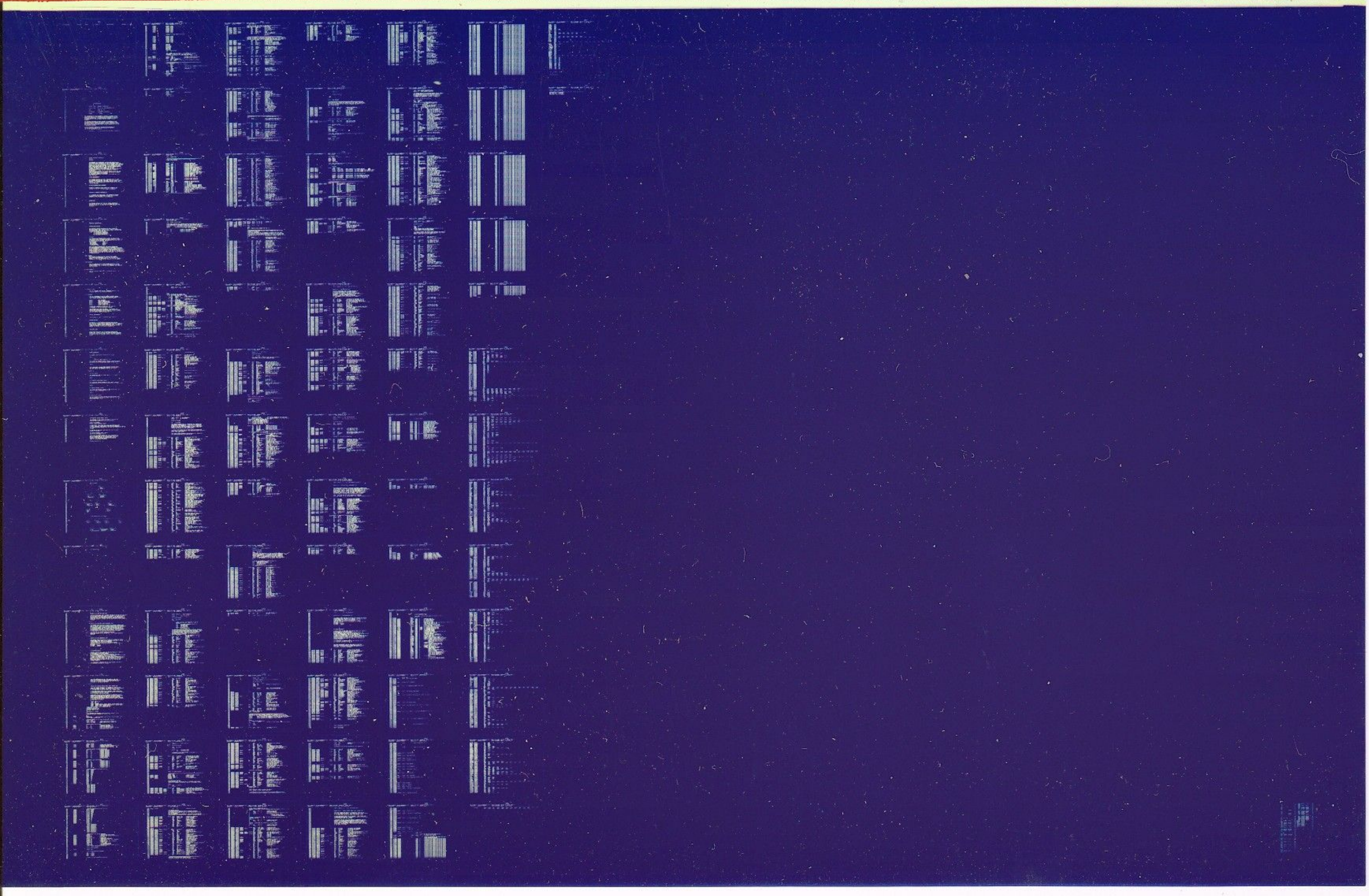
EP-DZDXJ-A-DL-A

FICHE 1 OF 1

JUN 1977

digital

MADE IN USA



B01

EOF1DZKMACSEQ	00010000	770608	PDP10 411	HDR1DZDXJASEQ	00010000	770608

```
                        .REM    %
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZDXJ-A-D

PRODUCT NAME: DX11-B OFF-LINE ADDRESSING TEST

DATE: APRIL 29, 1977

MAINTAINER: DIAGNOSTIC PROGRAMMING

AUTHOR: R. MISNER

1.0 GENERAL PROGRAM INFORMATION

1.1 ABSTRACT

THIS PROGRAM IS A SUPPLEMENT TO DZDXA AND DZDXF DX11-B MAINTENANCE MODE DIAGNOSTICS AND IS A FUNCTIONAL CHECK THE ABILITY OF THE DX TO ACCESS ALL AREAS OF MEMORY USING DIFFERENT SOURCES FOR THE NPR DATA ADDRESS. IT IS ASSUMED THAT DZDXA AND DZDXF HAVE ALREADY BEEN RUN. THE THREE NPR ADDRESS SOURCES RESULT FROM DATA WRITES TO MEMORY, STATUS POINTER WORD (SPW) FETCHES, AND DEVICE STATUS TABLE (DST) FETCHES.
DIAGNOSIS IS BASED ON THE ASSUMPTION THAT THE CPU CAN ACCESS MEMORY CORRECTLY WHILE THE DX'S ADDRESSING CAPABILITY IS SUSPECT.
ERROR MESSAGES DESCRIBE THE FAILING ADDRESS SOURCE AND THE EXPECTED LOCATION OF THE DX ACCESS. THE ACTUAL DX ADDRESS OF THE ACCESS CANNOT BE DETERMINED.
THE OPERATOR INTERFACE IS DESIGNED TO BE COMPATIBLE WITH DZDXA AND DZDXF.

1.2 SYSTEM REQUIREMENTS

THIS PROGRAM REQUIRES ANY PDP11 PROCESSOR WITH AT LEAST 8K OF MEMORY BUT NOT MORE THAN 128K; THE DX11-B AND A CONSOLE TERMINAL. THE PROGRAM WILL RUN UNDER XXDP,ACT,SLIDE, AND STAND ALONE. THIS PROGRAM HAS NOT BEEN TESTED ON CPU'S WITHOUT A SWITCH REGISTER.

1.3 RELATED DOCUMENTS AND STANDARDS

PROGRAM IS ASSEMBLED USING SYSMAC MD-11-DZQAC-C3 AND CONFORMS TO THE ACT INTERFACE AUTOCAT-11-QZAUB-B-D

1.4 DIAGNOSTIC HIERARCHY PREREQUSITES

IT IS ASSUMED THAT DZDXA,DZDXF, AND APPROPRIATE MEMORY DIAGNOSTICS ARE RUN ERROR FREE IN ORDER FOR THE ERROR MESSAGES IN THIS DIAGNOSTIC TO BE MEANINGFUL.

1.5 ASSUMPTIONS

THIS PROGRAM ASSUMES THAT THE CPU, UNIBUS ADDRESSING, AND PORTIONS OF THE DX11-B TESTED BY DZDXA AND DZDXF ARE OPERATING CORRECTLY IN ORDER FOR THE ERROR MESSAGES TO BE MEANINGFUL.

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.

```
159
160
161                                 "DX PRIORITY LEVEL"
162
163                                       THIS IS A REQUEST FOR THE DX'S INTERRUPT PRIORITY LEVEL.
164                                       IT IS NOT USED BY THIS PARTICULAR PROGRAM.
165
166
167                                 "SET SWITCHES:"
168
169                                       THIS IS A REMINDER TO SET THE CONSOLE SWITCHES BEFORE
170                                       STARTING THE PROGRAM. AFTER SETTING SWITCHES, RESPOND WITH A
171                                       <CR>. CONSOLE SWITCHES HAVE THE FOLLOWING MEANING:
172
173                                       SW<15>          HALT ON ERROR
174                                       SW<14>          LOOP ON ERROR OR TEST
175                                       SW<13>          INHIBIT ERROR PRINTOUT
176                                       SW<12>          PRINT ONLY SHORT ERROR REPORT
177                                       SW<11>          INHIBIT ITERATIONS
178                                       SW<10>          INHIBIT MEMORY MGT USE
179
180                                       TESTING WILL COMMENCE AS SOON AS A <CR> IS GIVEN
181                                       IN RESPONSE TO THE "SET SWITCHES:" MESSAGE.
182
183                                       THIS PROGRAM AUTO SIZES THE MEMORY, THE CONTROL UNIT
184                                       ADDRESS AND DEVICES PER CONTROL UNIT ARE ALSO AUTO SIZED.
185                                       THIS PARTICULAR PROGRAM DOES NOT NEED THE "LEGAL COMMANDS"
186                                       FACILITY USED BY OTHER DX DIAGNOSTICS.
187
188
189                                 2.2   SPECIAL ENVIRONMENTS
190                                       ---------------------
191
192                                       THIS PROGRAM WILL RUN IN UNATTENDED MODE UNDER ACT.
193
194
195                                 2.3   PROGRAM OPTIONS
196                                       ---------------
197
198                                       SEE SECTION 2.1 FOR PARAMETER SELECTION AND OPTIONS.
199                                       NOTE THAT ALL CONSOLE SWITCH CHANGES ARE EFFECTIVE IMMEDIATELY
200                                       EXCEPT SW<10> "INHIBIT MEMORY MANAGEMENT USES". IN ORDER TO
201                                       CHANGE THE MEANING OF SW<10> SET THE SWITCH THEN RESTART THE
202                                       PROGRAM VIA CONTROL C OR START AT ADDRESS 200.
203
204
205                                 2.4   EXECUTION TIMES
206                                       ---------------
207
208                                       SINGLE PASS EXECUTION TIME IS VARIABLE DEPENDING ON
209                                       MEMORY SIZE. A PDP-11/45 WITH 65K WORDS OF MEMORY REQUIRES
210                                       50 SECONDS. EACH ROUTINE HAS A SINGLE ITERATION REGARDLESS
211                                       OF MODE. END PASS IS SIGNALED AFTER ALL TESTS HAVE BEEN RUN
212                                       ONCE.
213
214
```

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

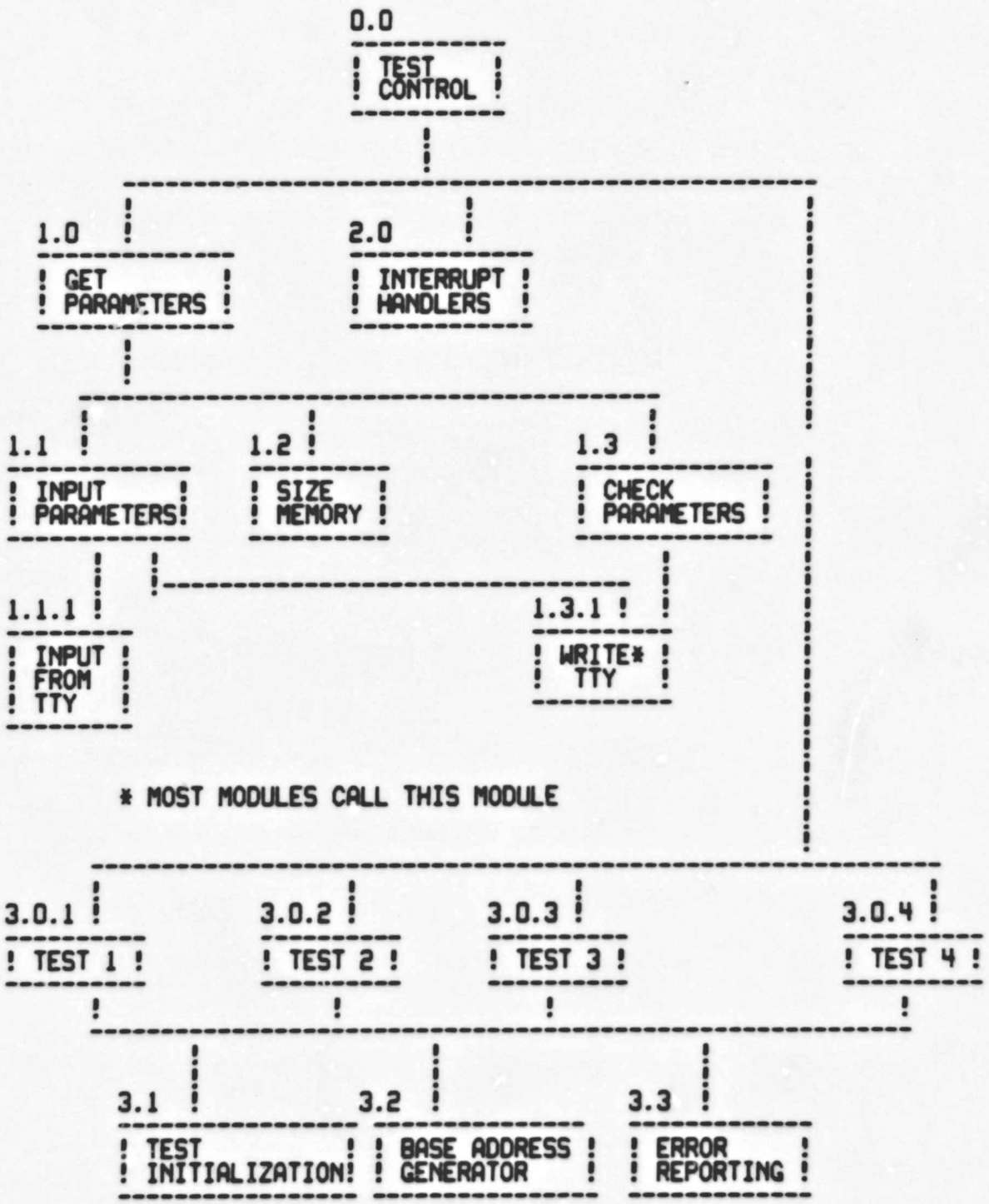
EA BITS IN OCT: XXXXXX

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



```
                                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 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 COMMMAND
                                OR AND ERROR IS FLAGGED.
                                %
                                .NLIST  CND,MC,TOC
                                .LIST   ME
                                .ENABL  ABS,AMA
                                .MCALL  .HEADER,.$CATCH,.SETUP,.$CMTAG,.$TYPE,GETSWR,CKSWR
                                .MCALL  .STRAP,.EQUAT,.$POWER,SETPRI,.$TYPOCT,.$READ,.$RDOCT
                                .MCALL  .$ACT11,.KT11,.$SIZE,PUSH,POP
                        .TITLE  MD-11-DZDXJ-A  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 FROM 4 - 776 CONTAIN A ".+2,HALT"
                                                ;#SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                                ;#LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                        000174                          .=174
000174  000000                                 DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                                 SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  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
                                                .MACRO  $$CMREG A,B
                                                $REG'A: .WORD   0               ;;CONTAINS (($REGAD)+'B)
                                                .NLIST
                                                $CM1=$CM1+1
                                                $CM2=$CM2+2
                                                .LIST
                                                .ENDM   $$CMREG
```

```
                        .MACRO  $$CMTMP A
                        $TMP'A: .WORD   0               ;;USER DEFINED
                        .NLIST
                        $CM4=$CM4+1
                        .LIST
                        .ENDM   $$CMTMP
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  001100  $CMTAG:                 ;;START OF COMMON TAGS
001100  000000                  $PASS:  .WORD   0        ;;CONTAINS PASS COUNT
001102      000                 $ISINM: .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                          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$,@#ERRVEC   ;;SET UP ERROR VECTOR
001246  012737  177570  001140          MOV     #0SWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001254  012737  177570  001142          MOV     #0DISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001262  022777  177777  177650          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001270  001012                          BNE     66$             ;;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  011650                  CLR     FIRSTP          ;CLEAR 1ST PASS SW
001330  005037  011750                  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  011666  011670          MOV     COUNT,CNT       ;SET UP ITERATION COUNT
001346  013737  011630  011662  GO:     MOV     ATESTN,CTESTN   ;GET FIRST TEST NO.
001354  005037  011674                  CLR     LOCK            ;CLEAR LOOP LOCK SW
001360  005037  011672                  CLR     ERR             ;RESET ERROR INDICATOR
001364  013702  011662          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
```

```
758  001416  000002                          RTI                     ;;POP NEW PC AND PS
759  001420                          64$:
760
761  001420  032777  040000  177512          BIT     #40000,@SWR     ;IS LOOP SWITCH ON
762  001426  001356                          BNE     TST             ;BR IF YES
763  001430  005237  011662                  INC     CTESTN          ;INCREMENT TO NEXT TEST
764  001434  023737  011662  011612          CMP     CTESTN,HTESTN   ;HAVE ALL TEST BEEN RUN
765  001442  003750                          BLE     TST             ;BR IF NO, DISPATCH TO NEXT ONE
766  001444  005337  011670                  DEC     CNT             ;DECREMENT ITERATION COUNT
767  001450  001336                          BNE     GO              ;START OVER IF NOT EXPIRED
768  001452  005737  011750          EOP:    TST     DEVMSS          ;HAVE WE PRINTED DEVICE CONFIG YET?
769  001456  001026                          BNE     10$             ;BR IF YES
770  001460  005737  011746                  TST     DEVCT           ;WERE DEVICES RECOGNIZED?
771  001464  001423                          BEQ     10$             ;BR IF NO, NOTHING TO SAY
772  001466  004537  005134                  JSR     R5,PRINT        ;PRINT DEV NO. ASCII
773  001472  013162                          .WORD   DEVMS
774  001474  000000                          .WORD   0
775  001476  004537  005134                  JSR     R5,PRINT        ;PRINT DEV NO. OCTAL
776  001502  011744                          .WORD   FRSTDV
777  001504  100001                          .WORD   100001
778  001506  004537  005134                  JSR     R5,PRINT        ;PRINT DEVICE COUNT ASCII
779  001512  013220                          .WORD   DEVCMS
780  001514  000000                          .WORD   0
781  001516  004537  005134                  JSR     R5,PRINT        ;PRINT DEVICE COUNT OCT
782  001522  011746                          .WORD   DEVCT
783  001524  100001                          .WORD   100001
784  001526  012737  000001  011750          MOV     #1,DEVMSS       ;SET DEV CONFIG PRINTED SWITCH
785  001534  005737  011646          10$:    TST     KT              ;IS KT IN USE?
786  001540  001402                          BEQ     15$             ;BR IF NO
787  001542  005037  177572                  CLR     @#SR0           ;TURN OFF MEM MGT
788  001546  004537  005134          15$:    JSR     R5,PRINT
789  001552  012712                          .WORD   BELL
790  001554  000000                          .WORD   0               ;PRINT CONTROL PARAMETER
791  001556  013700  000042                  MOV     @#42,R0         ;ARE WE UNDER ACT
792  001562  001405                          BEQ     END1            ;BR IF NO, START OVER
793  001564  000005                          RESET
794  001566  004710                  SENDAD: JSR     PC,@R0          ;GO TO MONITOR
795  001570  000240                          NOP                     ;ROOM FOR
796  001572  000240                          NOP                     ;    ACT11
797  001574  000240                          NOP                     ;    STUFF
798  001576  000656                  END1:   BR      RESTART         ;START OVER
799
```

```
                        .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  011650  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  011600  011630  MOV     DTESTN,ATESTN   ;SETUP DEFAULTS
001626  013737  011602  011632  MOV     DUBA,AUBA       ;SETUP DEFAULTS
001634  013737  011604  011634  MOV     DVEC↑,AVECT     ;SETUP DEFAULTS
001642  013737  011606  011636  MOV     DPRIOR,APRIOR   ;SETUP DEFAULTS
001650  000564                  BR      90$             ;GO SEE ABOUT MEMORY SIZING
001652  022737  001566  000042  10$:    CMP     #SENDAD,@#42    ;UNDER ACT11?
001660  001560                  BEQ     90$             ;NO OPERATOR INTERACTION
001662  005737  011664          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  012262                  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  011600  011630  MOV     DTESTN,ATESTN   ;DEFAULT SETUP
001722  013737  011602  011632  MOV     DUBA,AUBA       ;DEFAULT SETUP
001730  013737  011604  011634  MOV     DVEC↑,AVECT     ;DEFAULT SETUP
001736  013737  011606  011636  MOV     DPRIOR,APRIOR   ;DEFAULT SETUP
001744  000526                  BR      90$             ;GO SEE ABOUT MEMORY SIZING
001746  122700  000120          CMPB    #'P,R0          ;DOES OPERATOR WANT PREVIOUS
001752  001011                  BNE     30$             ;BR IF NO
001754  005737  011664          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  012520                  .WORD   INVM            ;NEVER SET UP PROPERLY
```

```
856  001772  000000                        .WORD   0               ;CONTROL WORD,CRLF + DONT CONVERT
857  001774  000726                        BR      10$             ;GO TO TRY AGAIN
858  001776  122700  000123      30$:     CMPB    #'S,R0          ;ARE PARMS TO BE SELECTED?
859  002002  001062                        BNE     40$             ;BR IF NO
860  002004  005037  011664      35$:     CLR     PARMV           ;CLEAR PARM VALID SW
861  002010  004537  002272              JSR     R5,INPRM        ;ASK INPUT MODULE TO GET TEST NO:
862  002014  000002                        .WORD   2               ;TEST NUMBER WILL BE RETURNED IN R0
863                                                              ;IN OCTAL
864  002016  010037  011630              MOV     R0,ATESTN       ;SAVE TEST NUMBER IN ACTIVE PARM TABLE
865  002022  004537  004766              JSR     R5,CHECKP       ;CHECK IF TEST NO OK
866  002026  000002                        .WORD   2               ;PARAMETER INDICATES VERIFY TEST #
867  002030  005700                        TST     R0              ;R0=0 IF CHECK OK
868  002032  001364                        BNE     35$             ;STAY HERE UNTIL OPERATOR GETS IT RIGHT
869  002034  004537  002272      36$:     JSR     R5,INPRM        ;ASK INPUT MODULE TO GET BASE ADDR
870  002040  000003                        .WORD   3               ;PARAMETER INDICATING GET UBA BASE
871  002042  010037  011632              MOV     R0,AUBA         ;SAVE UB BASE ADDR IN ACTIVE TABLE
872  002046  004537  004766              JSR     R5,CHECKP       ;CHECK FOR VALID ENTRY
873  002052  000003                        .WORD   3               ;PARM TO INDICATE CHECK UBA
874  002054  005700                        TST     R0              ;R0=0 IF CHECK OK
875  002056  001366                        BNE     36$             ;STAY HERE UNTIL HE GETS IT RIGHT
876  002060  004537  002272      37$:     JSR     R5,INPRM        ;ASK INPUT MODULE TO GET VECTOR ADDR
877  002064  000004                        .WORD   4               ;PARM INDICATING VECTOR NEEDED
878  002066  010037  011634              MOV     R0,AVECT        ;SAVE VECTOR IN ACTIVE PARM TABLE
879  002072  004537  004766              JSR     R5,CHECKP       ;CHECK FOR VALID VECTOR
880  002076  000004                        .WORD   4               ;PARM INDICATING CHECK VECTOR
881  002100  005700                        TST     R0              ;R0=0 IF VECTOR OK
882  002102  001366                        BNE     37$             ;STAY HERE UNTIL HE GETS IT RIGHT
883  002104  004537  002272      38$:     JSR     R5,INPRM        ;GO GET DX PRIORITY
884  002110  000005                        .WORD   5
885  002112  010037  011636              MOV     R0,APRIOR       ;SAVE PRIORITY
886  002116  004537  004766              JSR     R5,CHECKP       ;CHECK PRIORITY ENTRY
887  002122  000005                        .WORD   5               ;PARM INDICATING PRIORITY CHECK
888  002124  005700                        TST     R0
889  002126  001366                        BNE     38$             ;STAY HERE UNTIL HE GETS IT RIGHT
890  002130  004537  005134              JSR     R5,PRINT        ;PRINT SET SWITCHES MESSAGE
891  002134  012674                        .WORD   SETSW           ;ADDR OF MESSAGE
892  002136  000000                        .WORD   0
893  002140  004537  002532              JSR     R5,INASC        ;DUMMY INPUT REQUEST TO WAIT FOR
894  002144  000001                        .WORD   1               ;OPERATOR TO SET SWITCHES
895  002146  000425                        BR      90$             ;GO SEE ABOUT MEMORY SIZING
896  002150  122700  000116      40$:     CMPB    #'N,R0          ;WAS AN "N" INPUT
897  002154  001236                        BNE     10$             ;BR IF NONE OF OPSN WERE INPUT
898  002156  004537  002272      45$:     JSR     R5,INPRM        ;CALL FOR TEST NUMBER TO BE INPUT
899  002162  000002                        .WORD   2               ;PARM SAYS GET TEST #.
900  002164  010037  011630              MOV     R0,ATESTN       ;SAVE STARTING TEST #
901  002170  004537  004766              JSR     R5,CHECKP       ;HAVE TEST # CHECKED FOR PROPER
902  002174  000002                        .WORD   2               ;RANGE
903  002176  005700                        TST     R0              ;IS TEST # OK?
904  002200  001366                        BNE     45$             ;STAY HERE UNTIL HE GETS IT RIGHT
905  002202  005737  011664              TST     PARMV           ;WE HAVE STARTING TEST #, ARE OTHER PARMS OK?
906  002206  001005                        BNE     90$             ;BR IF OK
907  002210  004537  005134              JSR     R5,PRINT        ;PRINT
908  002214  012536                        INVPRM                  ;POINTER TO "NEED MORE PARMS" MSG
909  002216  000000                        .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
910  002220  000614                        BR      10$             ;GO BACK AND DO IT ALL AGAIN
911                                 ;
```

```
912                                                     ;
913  002222  004737  004212                  90$:   JSR     PC,SIZE         ;CALL MEMORY SIZING MODULE
914  002226  012700  000015                         MOV     #15,R0          ;START OF LOOP TO SET UP DX ACCESS ADDRESSES
915  002232  012701  011530                         MOV     #DXDSA,R1       ;GET ADDR OF START OF TABLE
916  002236  013702  011632                         MOV     AUBA,R2         ;GET DX REG BASE ADDRESS
917  002242  010221                           95$:   MOV     R2,(R1)+        ;MOVE UBA TO TABLE
918  002244  062702  000002                         ADD     #2,R2           ;STEP TO NEXT UBA
919  002250  005300                                  DEC     R0              ;DECREMENT  COUNTER
920  002252  001373                                  BNE     95$             ;FILL THE 13 LOCATIONS OF THE TABLE
921  002254  012737  000001  011664                 MOV     #1,PARMV        ;SET PARAMETER VALID SWITCH
922  002262  012737  000001  011650                 MOV     #1,FIRSTP       ;SET NOT FIRST PASS SWITCH
923  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  012556                  .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  012570                  .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  011600          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  012613                  .WORD   BASEA           ;ASK FOR BASE ADDR
002410  000000                  .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
```

```
 980  002412  004537  002532          JSR     R5,INASC        ;GET THE OCTAL
 981  002416  000002                  .WORD   2               ;INPUT
 982  002420  011100                  MOV     (R1),R0         ;GET THE NUMBER INPUT
 983  002422  001042                  BNE     90$             ;BR IF NOT DEFAULTED
 984  002424  013700  011602          MOV     DUBA,R0         ;SUBSTITUTE THE DEFAULT
 985  002430  000437                  BR      90$             ;DONE
 986  002432  120127  000004  30$:    CMPB    R1,#4           ;DOES CALLER WANT VECTOR?
 987  002436  001014                  BNE     40$             ;BR IF NO
 988  002440  004537  005134          JSR     R5,PRINT        ;PRINT
 989  002444  012631                  .WORD   VECTA           ;ASK FOR VECTOR ADDR
 990  002446  000000                  .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
 991  002450  004537  002532          JSR     R5,INASC        ;GET THE INPUT
 992  002454  000002                  .WORD   2               ;IN OCTAL
 993  002456  011100                  MOV     (R1),R0         ;GET THE NUMBER INPUT
 994  002460  001023                  BNE     90$             ;BR IF NOT DEFAULTED
 995  002462  013700  011604          MOV     DVECT,R0        ;SUBSTITUTE THE DEFAULT
 996  002466  000420                  BR      90$             ;DONE
 997  002470  120127  000005  40$:    CMPB    R1,#5           ;DOES CALLER WANT PRIORITY?
 998  002474  001014                  BNE     80$             ;BR IF INVALID CALL
 999  002476  004537  005134          JSR     R5,PRINT        ;PRINT
1000  002502  012651                  .WORD   PRILV           ;ASK FOR PRIORITY LEVEL
1001  002504  000000                  .WORD   0               ;CONTROL WORD, CRLF + DON'T CONVERT
1002  002506  004537  002532          JSR     R5,INASC        ;REQUEST INPUT
1003  002512  000002                  .WORD   2               ;IN OCTAL
1004  002514  011100                  MOV     (R1),R0         ;GET THE INPUT
1005  002516  001004                  BNE     90$             ;BR IF OK,  NOT DEFAULTED
1006  002520  013700  011606          MOV     DPRIOR,R0       ;SUBSTITUTE THE DEFAULT
1007  002524  000401                  BR      90$
1008                            ;
1009  002526  000000          80$:    HALT                    ;HALT ON INVALID CALL TO THIS MOD
1010  002530  000205          90$:    RTS     R5              ;RETURN TO CALLER
1011                                  .SBTTL  MODULE 1.1.1  INPUT FROM TTY
1012
```

```
                                ;       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 .$READ MACRO
002550  013637  013306                  MOV     @(SP)+,INBUF    ;GET THE INPUT DATA
002554  000406                          BR      20$             ;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  013306                  MOV     (SP)+,INBUF     ;;POP STACK INTO INBUF
002572  012701  013306          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
```

```
1069  002666  000207                          RTS     PC              ;;RETURN TO CALLER
1070
1071                                          ;*TK SERVICE ROUTINE
1072                                          ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
1073                                          ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
1074                                          ;*IT IN THE QUEUE.
1075                                          ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) STKINT IS CALLED AND
1076                                          ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (START)
1077                                          ;
1078  002670  117746  176252                  $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
1079  002674  042716  177600                          BIC     #↑C177,(SP)     ;;STRIP THE JUNK
1080  002700  021627  000003                          CMP     (SP),#3         ;;IS IT A CONTROL C?
1081  002704  001007                                  BNE     1$              ;;BRANCH IF NO
1082  002706  104401  004010                          TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (↑C)
1083  002712  004737  002620                          JSR     PC,$TKINT       ;;INIT THE KEYBOARD
1084  002716  005726                                  TST     (SP)+           ;;CLEAN UP STACK
1085  002720  000137  001164                          JMP     START           ;;CONTROL C RESTART
1086  002724  021627  000007                  1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
1087  002730  001004                                  BNE     2$              ;;BRANCH IF NO
1088  002732  022737  000176  001140                  CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
1089  002740  001500                                  BEQ     6$              ;;GO TO SWR CHANGE
1090
1091  002742                                  2$:
1092  002742  022737  000010  002602                  CMP     #10,$TKCNT      ;;IS THE QUEUE FULL?
1093  002750  001004                                  BNE     3$              ;;BRANCH IF NO
1094  002752  104401  004004                          TYPE    ,$BELL          ;;RING THE TTY BELL
1095  002756  005726                                  TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
1096  002760  000451                                  BR      5$              ;;EXIT
1097  002762  021627  000023                  3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
1098  002766  001021                                  BNE     32$             ;;BRANCH IF NO
1099  002770  005077  176150                          CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
1100  002774  005726                                  TST     (SP)+           ;;CLEAN CHAR OFF STACK
1101  002776  105777  176142                  31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
1102  003002  100375                                  BPL     31$             ;;LOOP UNTIL ITS THERE
1103  003004  117746  176136                          MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
1104  003010  042716  177600                          BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII
1105  003014  022627  000021                          CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
1106  003020  001366                                  BNE     31$             ;;BRANCH IF NO
1107  003022  012777  000100  176114                  MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
1108  003030  000002                                  RTI                     ;;RETURN
1109  003032  005237  002602                  32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
1110  003036  021627  000140                          CMP     (SP),#140       ;;IS IT UPPER CASE?
1111  003042  002405                                  BLT     4$              ;;BRANCH IF YES
1112  003044  021627  000175                          CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
1113  003050  003002                                  BGT     4$              ;;BRANCH IF YES
1114  003052  042716  000040                          BIC     #40,(SP)        ;;MAKE IT UPPER CASE
1115  003056  112677  177522                  4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
1116  003062  005237  002604                          INC     $TKQIN          ;;UPDATE THE POINTER
1117  003066  023727  002604  002620                  CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
1118  003074  001003                                  BNE     5$              ;;BRANCH IF NO
1119  003076  012737  002610  002604                  MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
1120  003104  000002                          5$:     RTI                     ;;RETURN
1121
1122                                          ;;**************************************************************
1123                                          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
1124                                          ;*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

003230  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?
```

```
1181  003330  001403                          BEQ     11$             ;;BRANCH IF YES
1182  003332  016677  000002  175600          MOV     2(SP),@SWR      ;;SAVE NEW SWR
1183  003340  062706  000006                  ADD     #6,SP           ;;CLEAR UP STACK
1184  003344  104401  001161          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
1185  003350  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
1186  003356  001003                          BNE     15$             ;;BRANCH IF NOT
1187  003360  012777  000100  175556          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
1188  003366  000002                  15$:    RTI                     ;;RETURN
1189  003370  004737  005356          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
1190  003374  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
1191  003400  002420                          BLT     18$             ;;BRANCH IF YES
1192  003402  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
1193  003406  003015                          BGT     18$             ;;BRANCH IF YES
1194  003410  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
1195  003414  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
1196  003420  001403                          BEQ     17$             ;;BRANCH IF YES
1197  003422  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
1198  003424  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
1199  003426  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
1200  003430  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
1201  003434  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
1202  003440  000667                          BR      7$              ;;GET THE NEXT ONE
1203  003442  104401  001160          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
1204  003446  000720                          BR      20$             ;;SIMULATE CONTROL-U
1205                                  .DSABL  LSB
1206
1207
1208                                  ;;*****************************************************************
1209                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
1210                                  ;*CALL:
1211                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
1212                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
1213                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
1214                                  ;
1215
1216  003450  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
1217  003452  016666  000004  000002  MOV     4(SP),2(SP)     ;;THE PS
1218  003460  005066  000004          CLR     4(SP)           ;;GET READY FOR A CHARACTER
1219  003464  005046                  CLR     -(SP)           ;;PUT NEW PS ON STACK
1220  003466  012746  003474          MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
1221  003472  000002                  RTI                     ;;POP NEW PC AND PS
1222  003474                  64$:
1223  003474  005737  002602  1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
1224  003500  001775                  BEQ     1$
1225  003502  005337  002602          DEC     $TKCNT          ;;DECREMENT THE COUNTER
1226  003506  117766  177074  000004  MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
1227  003514  005237  002606          INC     $TKQOUT         ;;UPDATE THE POINTER
1228  003520  023727  002606  002620  CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
1229  003526  001003                  BNE     2$              ;;BRANCH IF NO
1230  003530  012737  002610  002606  MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
1231  003536  000002          2$:     RTI                     ;;RETURN
1232                                  ;;*****************************************************************
1233                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
1234                                  ;*CALL:
1235                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
1236                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
```

```
                                        ;*                                      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
                                        $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
                                                CLR     -(SP)                   ;;CLEAR THE RUBOUT KEY
                                        1$:     MOV     #$TTYIN,R3              ;;GET ADDRESS
                                        2$:     CMP     #$TTYIN+10,R3           ;;BUFFER FULL?
                                                BLOS    4$                      ;;BR IF YES
                                                RDCHR                           ;;GO READ ONE CHARACTER FROM THE TTY
                                                MOVB    (SP)+,(R3)              ;;GET CHARACTER
                                        10$:    CMPB    #177,(R3)               ;;IS IT A RUBOUT
                                                BNE     5$                      ;;BR IF NO
                                                TST     (SP)                    ;;IS THIS THE FIRST RUBOUT?
                                                BNE     6$                      ;;BR IF NO
                                                MOVB    #'\,9$                  ;;TYPE A BACK SLASH
                                                TYPE    ,9$
                                                MOV     #-1,(SP)                ;;SET THE RUBOUT KEY
                                        6$:     DEC     R3                      ;;BACKUP BY ONE
                                                CMP     R3,#$TTYIN              ;;STACK EMPTY?
                                                BLO     4$                      ;;BR IF YES
                                                MOVB    (R3),9$                 ;;SETUP TO TYPEOUT THE DELETED CHAR.
                                                TYPE    ,9$                     ;;GO TYPE
                                                BR      2$                      ;;GO READ ANOTHER CHAR.
                                        5$:     TST     (SP)                    ;;RUBOUT KEY SET?
                                                BEQ     7$                      ;;BR IF NO
                                                MOVB    #'\,9$                  ;;TYPE A BACK SLASH
                                                TYPE    ,9$
                                                CLR     (SP)                    ;;CLEAR THE RUBOUT KEY
                                        7$:     CMPB    #25,(R3)                ;;IS CHARACTER A CTRL U?
                                                BNE     8$                      ;;BR IF NO
                                                TYPE    ,$CNTLU                 ;;TYPE A CONTROL "U"
                                                BR      1$                      ;;GO START OVER
                                        8$:     CMPB    #22,(R3)                ;;IS CHARACTER A "^R"?
                                                BNE     3$                      ;;BRANCH IF NO
                                                CLRB    (R3)                    ;;CLEAR THE CHARACTER
                                                TYPE    ,$CRLF                  ;;TYPE A "CR" & "LF"
                                                TYPE    ,$TTYIN                 ;;TYPE THE INPUT STRING
                                                BR      2$                      ;;GO PICKUP ANOTHER CHACTER
                                        4$:     TYPE    ,$QUES                  ;;TYPE A '?'
                                                BR      1$                      ;;CLEAR THE BUFFER AND LOOP
                                        3$:     MOVB    (R3),9$                 ;;ECHO THE CHARACTER
                                                TYPE    ,9$
                                                CMPB    #15,(R3)+               ;;CHECK FOR RETURN
                                                BNE     2$                      ;;LOOP IF NOT RETURN
                                                CLRB    -1(R3)                  ;;CLEAR RETURN (THE 15)
                                                TYPE    ,$LF                    ;;TYPE A LINE FEED
                                                TST     (SP)+                   ;;CLEAN RUBOUT KEY FROM THE STACK
                                                MOV     (SP)+,R3                ;;RESTORE R3
                                                MOV     (SP),-(SP)              ;;ADJUST THE STACK AND PUT ADDRESS OF THE
                                                MOV     4(SP),2(SP)             ;;      FIRST ASCII CHARACTER ON IT
                                                MOV     #$TTYIN,4(SP)
                                                RTI                             ;;RETURN
                                        9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
                                                .BYTE   0                       ;;TERMINATOR
                                        $TTYIN: .BLKB   10                      ;;RESERVE 10 BYTES FOR TTY INPUT
                                        $BELL:  .ASCIZ  <207><377><377>         ;;CODE FOR BELL
                                        $CNTLC: .ASCIZ  /^C/<15><12>            ;;CONTROL "C"
```

```
                                        $CNTLU: .ASCIZ  /↑U/<15><12>     ;;CONTROL "U"
                                        $CNTLG: .ASCIZ  /↑G/<15><12>     ;;CONTROL "G"
                                        $MSWR:  .ASCIZ  <15><12>/SWR = /
                                        $MNEW:  .ASCIZ  /  NEW = /

                                        .EVEN
                                        .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                        ;;*****************************************************************
                                        ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                        ;*CHANGE IT TO BINARY.
                                        ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
                                        ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
                                        ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
                                        ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
                                        ;*CALL:
                                        ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                        ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                        ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

004052  011646                 $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
004054  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
004062  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
004064  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
004066  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
004070  104410                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
004072  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
004074  010037  004200                  MOV     R0,5$           ;;AND SAVE IT
004100  005001                          CLR     R1              ;;CLEAR DATA WORD
004102  005002                          CLR     R2
004104  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
004106  001420                          BEQ     3$              ;;IF ZERO GET OUT
004110  122716  000060                  CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
004114  003026                          BGT     4$              ;;IS AN OCTAL DIGIT
004116  122716  000067                  CMPB    #'7,(SP)
004122  002423                          BLT     4$
004124  006301                          ASL     R1              ;;*2
004126  006102                          ROL     R2
004130  006301                          ASL     R1              ;;*4
004132  006102                          ROL     R2
004134  006301                          ASL     R1              ;;*8
004136  006102                          ROL     R2
004140  042716  177770                  BIC     #↑C7,(SP)       ;;STRIP THE ASCII JUNK
004144  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
004146  000756                          BR      2$              ;;LOOP
004150  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
004152  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
004156  010237  004210                  MOV     R2,$HIOCT
004162  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
004164  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
004166  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
004170  000002                          RTI                     ;;RETURN
004172  005726                  4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
004174  105010                          CLRB    (R0)            ;;SET A TERMINATOR
004176  104401                          TYPE                    ;;TYPE UP THRU THE BAD CHAR.
```

```
                        READ AN OCTAL NUMBER FROM THE TTY

  1349  004200  000000          5$:     .WORD   0
  1350  004202  104401  001160          TYPE    $QUES           ;;"?" "CR" & "LF"
  1351  004206  000730                  BR      1$              ;;TRY AGAIN
  1352  004210  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
  1353                                  .SBTTL  MODULE 1.2  SIZE MEMORY
  1354
```

```
                                        ;       MODULE 1.2      SIZE MEMORY
                                        ;
                                        ;       CALLED AS FOLLOWS
                                        ;
                                        ;       JSR     PC,SIZE
                                        ;
                                        ;       THIS MODULE SETS UP MEMORY SIZE INCLUDING EXTENDED
                                        ;       ADDRESSING BITS IN ACTIVE PARAMETER TABLE
                                        ;
                                        ;
004212  005037  004516                  SIZE:   CLR     $KT11           ;DON'T TRY FOR MEM MGT
004216  004737  004460                          JSR     PC,$SIZE        ;GET MEM SIZE WITH KT OFF
004222  005037  011646                          CLR     KT              ;RESET KT AVAILABLE SWITCH
004226  013737  004762  011642                  MOV     $LSTAD,MMAX     ;SAVE END OF CORE WITH KT OFF
004234  013737  004762  011640                  MOV     $LSTAD,MSIZE    ;GET END OF CORE TO WORK WITH
004242  023727  004762  017776                  CMP     $LSTAD,#17776   ;LESS THAN 8K?
004250  103415                                  BLO     35$             ;BRANCH IF YES, DON'T PRESERVE LOADERS
004252  162737  000276  011640                  SUB     #276,MSIZE      ;SAVE SPACE FOR ABS LOADER
004260  005737  000042                          TST     @#42            ;RUNNING UNDER ACT OR XXDP
004264  001407                                  BEQ     35$             ;BR IF NO
004266  023737  000042  000046                  CMP     @#42,@#46       ;RUNNING UNDER XXDP CHAIN?
004274  001403                                  BEQ     35$             ;BR IF NO
004276  162737  005500  011640                  SUB     #5500,MSIZE     ;SAVE SPACE FOR LOADERS
004304  005037  011644                  35$:    CLR     EASIZE          ;CLEAN UP EXTENDED ADDR
004310  005037  011700                          CLR     EABITS
004314  037727  174620  002000                  BIT     @SWR,#SW10      ;CHECK IF MM TO BE USED
004322  001055                                  BNE     30$             ;BR IF NO
004324  012737  000200  004516          10$:    MOV     #200,$KT11
004332  004737  004460                  20$:    JSR     PC,$SIZE
004336  005737  004516                          TST     $KT11           ;IS MM ACTIVE
004342  100045                                  BPL     30$             ;BR IF NO
004344  012737  000001  011646                  MOV     #1,KT           ;SET KT IN USE FLAG
004352  013737  004764  011644                  MOV     $LSTBK,EASIZE
004360  062737  000040  011644                  ADD     #40,EASIZE      ;INCR TO FIRST INVALID SETTING
004366  013737  004764  011702                  MOV     $LSTBK,PHYMAX   ;START GEN OF MAX PHYSICAL
004374  006337  011702                          ASL     PHYMAX          ;   ADDR
004400  006337  011702                          ASL     PHYMAX          ;     LOW
004404  006337  011702                          ASL     PHYMAX          ;       16
004410  006337  011702                          ASL     PHYMAX          ;         BITS
004414  006337  011702                          ASL     PHYMAX
004420  006337  011702                          ASL     PHYMAX
004424  052737  003776  011702                  BIS     #3776,PHYMAX    ;SET UP TO USE LAST 1K TOO
004432  013737  004764  011700                  MOV     $LSTBK,EABITS
004440  042737  171777  011700                  BIC     #171777,EABITS  ;SAVE ONLY EXT ADDR BITS
004446  000337  011700                          SWAB    EABITS          ;MOVE TO BIT POS. 3,2
004452  006337  011700                          ASL     EABITS          ;MOVE TO BIT POS. 4,3 FOR DX11
004456  000207                          30$:    RTS     PC              ;RETURN
                                        ;
                                        .SBTTL  ROUTINE TO SIZE MEMORY
                                        ;
                                        ;;*****************************************************************
                                        ;*CALL:
                                        ;*      JSR     PC,$SIZE
                                        ;*      RETURN
                                        ;*SLSTAD WILL CONTAIN:
```

```
                                ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
                                ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
                                ;*SLSTBK WILL CONTAIN THE LAST BANK AS A SAF
                                ;*SKT11 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
```

```
1467   004716  162701  004000          $CROUT: SUB     #4000,R1
1468   004722  162702  000040          $SIZEX: SUB     #40,R2          ;;DROP BACK
1469   004726  010006                          MOV     R0,SP           ;;RESTORE THE STACK
1470   004730  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
1471   004734  012637  000004                  MOV     (SP)+,@#ERRVEC
1472   004740  010137  004762                  MOV     R1,$LSTAD       ;;LAST ADDRESS
1473   004744  010237  004764                  MOV     R2,$LSTBK       ;;LAST BANK
1474   004750  012603                          MOV     (SP)+,R3        ;;RESTORE R3
1475   004752  012602                          MOV     (SP)+,R2        ;;RESTORE R2
1476   004754  012601                          MOV     (SP)+,R1        ;;RESTORE R1
1477   004756  012600                          MOV     (SP)+,R0        ;;RESTORE R0
1478   004760  000207                          RTS     PC
1479   004762  000000          $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
1480   004764  000000          $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
1481
1482                                  .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  011610          CMP     R0,LTESTN       ;CHECK LOW LIMIT
005002  002443                  BLT     90$             ;BR IF ERROR
005004  020037  011612          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  011614          CMP     R0,LUBA         ;CHECK LOW LIMIT
005026  002431                  BLT     90$             ;BR IF BAD
005030  020037  011616          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  011620          CMP     R0,LVEC         ;CHECK THE LOW LIMIT
005052  002417                  BLT     90$             ;BR IF BAD
005054  020037  011622          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  011624          CMP     R0,LPRIOR       ;CHECK LOW LIMIT
005076  002405                  BLT     90$             ;BR IF BAD
005100  020037  011626          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  012724                  .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
```

```
  1539  005130  005000                  80$:    CLR     R0              ;SET GOOD RETURN CODE
  1540  005132  000205                  95$:    RTS     R5              ;RETURN TO CALLER.
  1541
  1542                                          .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  013301                          .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 RTN
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
```

```
1599  005214  000000                    HALT                    ;;HALT HERE IF NO TERMINAL
1600  005216  000407                    BR      3$              ;;LEAVE
1601  005220  010046                1$: MOV     R0,-(SP)        ;;SAVE R0
1602  005222  017600  000002            MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
1603  005226  112046                2$: MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1604  005230  001005                    BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
1605  005232  005726                    TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
1606  005234  012600                60$: MOV     (SP)+,R0        ;;RESTORE R0
1607  005236  062716  000002        3$: ADD     #2,(SP)         ;;ADJUST RETURN PC
1608  005242  000002                    RTI                     ;;RETURN
1609  005244  122716  000011        4$: CMPB    #HT,(SP)        ;;BRANCH IF <HT>
1610  005250  001430                    BEQ     8$
1611  005252  122716  000200            CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
1612  005256  001006                    BNE     5$
1613  005260  005726                    TST     (SP)+           ;;POP  <CR><LF> EQUIV
1614  005262  104401                    TYPE                    ;;TYPE A CR AND LF
1615  005264  001161                    $CRLF
1616  005266  105037  005422            CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
1617  005272  000755                    BR      2$              ;;GET NEXT CHARACTER
1618  005274  004737  005356        5$: JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
1619  005300  123726  001156        6$: CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
1620  005304  001350                    BNE     2$              ;;IF NO GO GET NEXT CHAR.
1621  005306  013746  001154            MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
1622                                                            ;;AND THE NULL CHAR.
1623  005312  105366  000001        7$: DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
1624  005316  002770                    BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
1625  005320  004737  005356            JSR     PC,$TYPEC       ;;GO TYPE A NULL
1626  005324  105337  005422            DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
1627  005330  000770                    BR      7$              ;;LOOP
1628
1629                                ;HORIZONTAL TAB PROCESSOR
1630
1631  005332  112716  000040        8$: MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
1632  005336  004737  005356        9$: JSR     PC,$TYPEC       ;;TYPE A SPACE
1633  005342  132737  000007  005422    BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
1634  005350  001372                    BNE     9$              ;;TAB STOP
1635  005352  005726                    TST     (SP)+           ;;POP SPACE OFF STACK
1636  005354  000724                    BR      2$              ;;GET NEXT CHARACTER
1637  005356  105777  173566    $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
1638  005362  100375                    BPL     $TYPEC
1639  005364  116677  000002  173560      MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
1640  005372  122766  000015  000002    CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
1641  005400  001003                    BNE     1$              ;;BRANCH IF NO
1642  005402  105037  005422            CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
1643  005406  000406                    BR      $TYPEX          ;;EXIT
1644  005410  122766  000012  000002  1$: CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
1645  005416  001402                    BEQ     $TYPEX          ;;BRANCH IF YES
1646  005420  105227                    INCB    (PC)+           ;;COUNT THE CHARACTER
1647  005422  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
1648  005424  000207          $TYPEX: RTS     PC
1649
1650                                ;TYPOC TRAP CALL COMES TO $TYPOC TAG BELOW
1651                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
1652
1653                                ;;**************************************************************
1654                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
```

```
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;*      .BYTE   M               ;;M=1 OR 0
                                        ;*                              ;;1=TYPE LEADING ZEROS
                                        ;*                              ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
005426  017646  000000                  $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
005432  116637  000001  005651                  MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
005440  112637  005653                          MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
005444  062716  000002                          ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
005450  000406                          BR      $TYPON
005452  112737  000001  005651  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
005460  112737  000006  005653          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
005466  112737  000005  005650  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
005474  010346                          MOV     R3,-(SP)        ;;SAVE R3
005476  010446                          MOV     R4,-(SP)        ;;SAVE R4
005500  010546                          MOV     R5,-(SP)        ;;SAVE R5
005502  113704  005653                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
005506  005404                          NEG     R4
005510  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
005514  110437  005652                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
005520  113704  005651                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
005524  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
005530  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
005532  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
005534  000404                          BR      3$              ;;GO DO MSB
005536  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
005540  006105                          ROL     R5
005542  006105                          ROL     R5
005544  010503                          MOV     R5,R3
005546  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
005550  105337  005652                  DECB    $OMODE          ;;TYPE THIS DIGIT?
005554  100016                          BPL     7$              ;;BR IF NO
005556  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
005562  001002                          BNE     4$              ;;TEST FOR 0
005564  005704                          TST     R4              ;;SUPPRESS THIS 0?
005566  001403                          BEQ     5$              ;;BR IF YES
005570  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
005572  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
005576  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
005602  110337  005646                  MOVB    R3,@$           ;;SAVE FOR TYPING
```

```
 1711  005606  104401  005646                TYPE    ,B$             ;;GO TYPE THIS DIGIT
 1712  005612  105337  005650        7$:     DECB    $OCNT           ;;COUNT BY 1
 1713  005616  003347                BGT     2$              ;;BR IF MORE TO DO
 1714  005620  002402                BLT     6$              ;;BR IF DONE
 1715  005622  005204                INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 1716  005624  000744                BR      2$              ;;GO DO THE LAST DIGIT
 1717  005626  012605        6$:     MOV     (SP)+,R5        ;;RESTORE R5
 1718  005630  012604                MOV     (SP)+,R4        ;;RESTORE R4
 1719  005632  012603                MOV     (SP)+,R3        ;;RESTORE R3
 1720  005634  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 1721  005642  012616                MOV     (SP)+,(SP)
 1722  005644  000002                RTI                     ;;RETURN
 1723  005646     000        B$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 1724  005647     000                .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 1725  005650     000        $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 1726  005651     000        $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 1727  005652  000000        $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
 1728                                .SBTTL  MODULE 2.0  INTERRUPT HANDLERS
```

```
1729                                    ;
1730                                    ;       MODULE 2.0
1731                                    ;
1732                                    ;
1733                                    ;       INTERRUPT HANDLERS
1734                                    ;
1735                                    ;
1736                                    ;
1737                                    ;
1738                                    ;
1739                                    .SBTTL  TRAP DECODER
1740                                    ;
1741                                    ;;*****************************************************************
1742                                    ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
1743                                    ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
1744                                    ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
1745                                    ;*GO TO THAT ROUTINE.
1746
1747  005654  016646  000002            $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
1748  005660  042716  000020                    BIC     #20,(SP)        ;;  THE CALLER--DO NOT ALLOW
1749  005664  012746  005672                    MOV     #1$,-(SP)       ;;  T-BIT TRAPS
1750  005670  000002                            RTI                     ;;SET THE NEW STATUS
1751  005672  010046                    1$:     MOV     R0,-(SP)        ;;SAVE R0
1752  005674  016600  000002                    MOV     2(SP),R0        ;;GET TRAP ADDRESS
1753  005700  005740                            TST     -(R0)           ;;BACKUP BY 2
1754  005702  111000                            MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
1755  005704  006300                            ASL     R0              ;;POSITION FOR INDEXING
1756  005706  016000  005726                    MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
1757  005712  000200                            RTS     R0              ;;GO TO ROUTINE
1758
1759
1760                                    ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
1761
1762  005714  011646                    $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
1763  005716  016666  000004  000002            MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
1764  005724  000002                            RTI                     ;;RESTORE THE PSW
1765
1766                                    .MACRO  SETTRAP A,B,MSG
1767                                            $$SET   A,B,\<TRAP+$TRP>,\$TRP,<MSG>
1768                                    .NLIST
1769                                    $TRP=$TRP+1
1770                                    .LIST
1771                                    .ENDM   SETTRAP
1772                                    .MACRO  $$SET   A,B,C,D,COMNT
1773                                    .IF EQ $TRP-1
1774                                    .SBTTL  TRAP TABLE
1775
1776                                    ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
1777                                    ;*BY THE "TRAP" INSTRUCTION.
1778
1779                                    ;       ROUTINE
1780                                    ;       -------
1781                                    $TRPAD: .WORD   $TRAP2
1782                                    .ENDC
1783                                    .IIF NDF GNS,.NLIST
1784                                            A=      C
```

```
1785                                    .IIF NDF GNS,.LIST
1786                                            B       ;;CALL=A        TRAP+D(C)       COMNT
1787                                    .ENDM   $$SET
1788                                    .MACRO  TRMTRP
1789                                    $TERM=.-$TRPAD
1790                                    .ENDM   TRMTRP
1791                                    .SBTTL  TRAP TABLE
1792
1793                                    ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
1794                                    ;*BY THE "TRAP" INSTRUCTION.
1795
1796                                    ;       ROUTINE
1797                                    ;       -------
1798  005726  005714                    $TRPAD: .WORD   $TRAP2
1799  005730  005206                            $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
1800  005732  005452                            $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
1801  005734  005426                            $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
1802  005736  005466                            $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
1803
1804  005740  003176                            $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
1805
1806  005742  003106                            $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
1807  005744  003450                            $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
1808  005746  003540                            $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
1809  005750  004052                            $RDOCT  ;;CALL=RDOCT    TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
1810
1811                                    .SBTTL  POWER DOWN AND UP ROUTINES
1812
1813                                    ;;**************************************************************
1814                                    ;POWER DOWN ROUTINE
1815  005752  012737  006116  000024    $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
1816  005760  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;PRIO:7
1817  005766  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1818  005770  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1819  005772  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
1820  005774  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
1821  005776  010446                            MOV     R4,-(SP)        ;;PUSH R4 ON STACK
1822  006000  010546                            MOV     R5,-(SP)        ;;PUSH R5 ON STACK
1823  006002  017746  173132                    MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
1824  006006  010637  006122                    MOV     SP,$SAVR6       ;;SAVE SP
1825  006012  012737  006024  000024            MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
1826  006020  000000                            HALT
1827  006022  000776                            BR      .-2             ;;HANG UP
1828
1829                                    ;;**************************************************************
1830                                    ;POWER UP ROUTINE
1831  006024  012737  006116  000024    $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
1832  006032  013706  006122                    MOV     $SAVR6,SP       ;;GET SP
1833  006036  005037  006122                    CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
1834  006042  005237  006122            1$:     INC     $SAVR6          ;;WAIT FOR THE INC
1835  006046  001375                            BNE     1$              ;;OF  WORD
1836  006050  012677  173064                    MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
1837  006054  012605                            MOV     (SP)+,R5        ;;POP STACK INTO R5
1838  006056  012604                            MOV     (SP)+,R4        ;;POP STACK INTO R4
1839  006060  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
1840  006062  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
1841  006064  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
1842  006066  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
1843  006070  012737  005752  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
1844  006076  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
1845  006104  104401                        TYPE                    ;REPORT THE POWER FAILURE
1846  006106  006124              $PWRMG:   .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
1847  006110  012716                        MOV     (PC)+,(SP)      ;;RESTART AT RESTART
1848  006112  001334              $PWRAD:   .WORD   RESTART         ;;RESTART ADDRESS
1849  006114  000002                        RTI
1850  006116  000000              $ILLUP:   HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
1851  006120  000776                        BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
1852  006122  000000              $SAVR6:   0                       ;;PUT THE SP HERE
1853  006124  005015  047520  042527  $POWER:  .ASCIZ  <15><12>"POWER"
1854  006132  000122
1855                                        .EVEN
```

```
                                        .SBTTL  MODULE 3.0.1  TST1  FAST NPR DATA TEST
                                ;
                                ;       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.
                                ;
006134  004537  006530        TST1:   JSR     R5,INIT         ;INITIALIZE THE SOFTWARE, PS, ETC
006140  004737  006560                JSR     PC,DXRES        ;PUT THE DX IN A GOOD KNOWN STATE
006144  004537  010274                JSR     R5,BUFAD        ;GET FIRST BUFFER BASE AND SET UP KT
006150  100402                        .WORD   100402          ;CTL BIT 15=INITIALIZE, BUFFER BASE DIVISABLE
                                                                ;BY 400, SET UP BUFMAX
006152  013700  011634                MOV     AVECT,R0        ;SET UP
006156  012720  006230                MOV     #INTR1,(R0)+    ;DX INTERRUPT
006162  012710  000340                MOV     #340,(R0)       ;AND DX PSW
006166  012737  000402  011706  LOOP1: MOV     #402,BUFSZE     ;BUFFER SIZE A LITTLE BIGGER THAN OFFSET
                                                                ;TO CREATE A CARRY OUT OF HIGH ORDER BIT
006174  004737  006510        10$:    JSR     PC,BUFCLR       ;CLEAR THE BUFFER
006200  012737  100525  011574        MOV     #100525,DXM00   ;SET DATA PATTERN AND OPOUT
006206  053777  011712  003320  G01:  BIS     EACUR,@DXCSA    ;SET UP DX EXT ADDR BITS
006214  013777  011716  003316        MOV     BFBAPH,@DXBAA   ;SET UP DX BUS ADDR REG.
006222  004537  006366                JSR     R5,DXGO         ;START THE DX
006226  177376                        .WORD   -402            ;BYTE COUNT

                                ;INTERRUPT WILL RETURN HERE AFTER DATA TRANSFER IS COMPLETE

006230  005737  006506        INTR1:  TST     INTERR          ;DID WE GET HERE VIA INTER
006234  001006                        BNE     8$              ;BR IF NO, STACK IS OK
006236  012716  006244                MOV     #5$,@SP         ;SET UP RETURN ADDR
006242  000002                        RTI                     ;RELIEVE INTERRUPT STATUS
006244  012705  006252        5$:     MOV     #8$,R5          ;SET UP JSR RETURN ADDR
006250  000205                        RTS     R5              ;RELIEVE SUBROUTINE STATUS
006252  013700  011704        8$:     MOV     BUFBAS,R0       ;GET STARTING ADDR
006256  013701  011706                MOV     BUFSZE,R1       ;GET BUFFER SIZE
006262  123720  011574        10$:    CMPB    DXM00,(R0)+     ;CHECK A BYTE
006266  001404                        BEQ     15$             ;BRANCH IF NO ERROR
006270  004537  010746        12$:    JSR     R5,ERRR         ;REPORT AN ERROR
006274  000001                        .WORD   1               ;ERROR ID
006276  011762                        .WORD   ERR1            ;ADDR OF ERROR MESSAGE
006300  005301                15$:    DEC     R1              ;DECREMENT BUFFER COUNT
006302  001367                        BNE     10$             ;LOOP UNTIL ALL OF BUFFER CHECKED
006304  042777  000200  003222        BIC     #DONE,@DXCSA    ;TURN OFF FAST MODE
006312  042777  000004  003234        BIC     #SOSIEN,@DXESA  ;TURN OFF FAST MODE
006320  042777  000030  003206        BIC     #30,@DXCSA      ;CLEAR POSSIBLE CARRY IN EA BITS
006326  105737  011574                TSTB    DXM00           ;CHECK IF FIRST OF TWO PASSES
006332  100404                        BMI     30$             ;BR IF NOT FIRST PASS
006334  012737  100652  011574        MOV     #100652,DXM00   ;SET UP SECOND DATA PATTERN
006342  000721                        BR      G01             ;DO TRANSFER WITH THIS DATA
```

```
1912  006344  004737  006510          30$:    JSR     PC,BUFCLR       ;PUT BUFFER BACK TO ZEROS
1913  006350  004537  010274          ENDT1:  JSR     R5,BUFAD        ;GET NEXT BUFFER ADDRESS
1914  006354  000402                          .WORD   402             ;BASE TO BE DIVISABLE BY 400
1915  006356  005701                          TST     R1              ;CHECK FOR OUT OF MEMORY
1916  006360  001702                          BEQ     LOOP1           ;BRANCH IF THERE IS MORE
1917  006362  000137  001410                  JMP     ENDTST          ;OTHERWISE WE'RE DONE WITH TEST
1918                                  ;
1919                                  ;  SUBROUTINE TO START THE DX IN SOSIEN MODE
1920                                  ;
1921  006366  013777  011574  003150  DXGO:   MOV     DXM00,@DXMOA    ;SEND DATA AND OP OUT TO MAINT REG
1922  006374  012577  003142                  MOV     (R5)+,@DXBCA    ;SET UP BYTE COUNT FROM PASSED PARM.
1923  006400  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
1924  006402  012746  006410                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
1925  006406  000002                          RTI                     ;;POP NEW PC AND PS
1926  006410                          64$:
1927  006410  052777  000103  003116          BIS     #103,@DXCSA     ;SET  FUNCTION TO IBM WRITE
1928                                                                  ;AND ENABLE INTERRUPTS
1929  006416  052777  060000  003120          BIS     #SEL0!HLD0,@DXMOA       ;FORCE DX SELECTION
1930  006424  042777  060000  003112          BIC     #SEL0!HLD0,@DXMOA       ;DROP SELECT LINES
1931  006432  052777  002000  003104          BIS     #CMD0,@DXMOA            ;RAISE COMMAND OUT
1932  006440  042777  002000  003076          BIC     #CMD0,@DXMOA            ;DROP COMMAND OUT
1933  006446  052777  000004  003100          BIS     #SOSIEN,@DXESA          ;ENABLE FAST SERVICE OUT/SERV IN
1934                                                                          ;TO UNLEASH NPR'S
1935  006454  005037  006506                  CLR     INTERR          ;CLEAR NO INTERRUPT OCCURED FLAG
1936  006460  005001                          CLR     R1
1937  006462  005301                  10$:    DEC     R1              ;WAIT HERE FOR INTERRUPT
1938  006464  001376                          BNE     10$
1939  006466  004537  010746                  JSR     R5,ERRR         ;ERROR, INTERRUPT NEVER  CAME
1940  006472  000002                          .WORD   2               ;ERROR IDENTIFER
1941  006474  012006                          .WORD ERR2              ;ERROR MESSAGE POINTER
1942  006476  012737  000001  006506          MOV     #1,INTERR       ;SET NO INTERRUPT OCCURED FLAG
1943  006504  000205                          RTS     R5              ;CONTINUE TESTING
1944  006506  000000                  INTERR: .WORD   0       ;SET TO 1 IF DX11 FAILED TO INTERRUPT
1945
1946
1947                                  ;       SUBROUTINE TO CLEAR THE CURRENT BUFFER
1948
1949  006510  013700  011704          BUFCLR: MOV     BUFBAS,R0       ;GET BUFFER START
1950  006514  013701  011706                  MOV     BUFSZE,R1       ;GET BUFFER SIZE
1951  006520  105020                  10$:    CLRB    (R0)+           ;CLEAR A BYTE
1952  006522  005301                          DEC     R1              ;COUNT IT
1953  006524  001375                          BNE     10$             ;LOOP TO CLEAR BUFFER
1954  006526  000207                          RTS     PC              ;RETURN TO CALLER
1955
```

```
                                        .SBTTL  MODULE 3.1  TEST INITIALIZATION
                                ;
                                ;       MODULE 3.1      TEST SOFTWARE INITIALIZATION
                                ;
                                ;
                                ;       CALLED AS FOLLOWS:
                                ;
                                ;       JSR     R5,INIT
                                ;       RETURN COMES HERE
                                ;
                                ;
                                INIT:
006530                                  MOV     #340,-(SP)      ;;PUT NEW PS ON STACK
006530  012746  000340                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
006534  012746  006542                  RTI                     ;;POP NEW PC AND PS
006540  000002                  64$:
006542                                  CLR     ERR             ;RESET ERROR OCCURRED INDICATION
006542  005037  011672                  MOV     R5,LOCKAD       ;SET UP LOCK ON ERROR RETURN ADDR
006546  010537  011676                  CLR     INTERR          ;CLEAR MISSED INTR FLAG
006552  005037  006506                  RTS     R5
006556  000205
                                ;
                                ;       HARDWARE INITIALIZATION SUBROUTINE
                                ;
                                ;
006560  052777  000010  002766  DXRES:  BIS     #10,@DXESA      ;DISABLE DX TIME OUT, DEBUG ONLY
006566  042777  000200  002740          BIC     #DONE,@DXCSA    ;CLEAR DONE AND LOCKO
006574  042777  000006  002732          BIC     #6,@DXCSA       ;SET FUNCTION TO RESET
006602  052777  000001  002724          BIS     #1,@DXCSA       ;ISSUE THE DX RESET
006610  042777  001230  002716          BIC     #1230,@DXCSA    ;CLEAR POTENTIAL DISASTER OF ONLINE OR INTEN
                                                                ;CLEAR EA BITS ALSO
006616  012777  014000  002712          MOV     #SPW,@DXOSA     ;SET UP CU OFFSET TO SPW TABLE
006624  005737  011760                  TST     SPWMVD          ;IS SPW TABLE AT HOME BASE?
006630  001403                          BEQ     5$              ;BR IF YES
006632  012777  016000  002676          MOV     #SPW+2000,@DXOSA        ;SET UP ALTERNATE SPW LOCATION
006640  032777  100000  002702  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  010274                  JSR     R5,BUFAD        ;GET STARTING BUFFER ADDRESS
006672  101000                          .WORD   101000          ;PARM MEANS INIT AND SET UP1000 BYTE BUFFER
006674  013700  011634                  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  011706  LOOP2:  MOV     #1000,BUFSZE    ;ESTABLISH BUFFER SIZE
006716  004737  006510                  JSR     PC,BUFCLR       ;CLEAR THE BUFFER
006722  013737  011704  011714          MOV     BUFBAS,DATAD    ;INITIALIZE DATA ADDR
006730  013737  011716  011720          MOV     BFBAPH,BFADPH   ;GET PHYSICAL BASE ADDR
006736  053777  011712  002570  LOP2:   BIS     EACUR,@DXCSA    ;SET EA BITS IN THE DX11
006744  012737  100377  011574          MOV     #100377,DXMOD   ;DATA PATTERN TO BE WRITTEN BY DX
                                ;
006752  013777  011720  002560          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$,R5          ;SET UP RETURN ADDR
007006  000205                          RTS     R5              ;RELIEVE SUBROUTINE STATUS
007010  027727  002700  177777  8$:     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  010746                  JSR     R5,ERRR         ;REPORT THE ERROR
007024  000003                          .WORD   3               ;ERROR NUMBER THREE
007026  012026                          .WORD   ERR3            ;ADDR OF ERROR MESSAGE
007030  005077  002660          10$:    CLR     @DATAD          ;RESTORE THE WORD UNDER TEST
007034  042777  000200  002472          BIC     #DONE,@DXCSA    ;TURN OFF INTERRUPTS
007042  042777  000004  002504          BIC     #SOSIEN,@DXESA  ;CLEAR FAST MODE
007050  042777  000030  002456          BIC     #30,@DXCSA      ;CLEAR POSSIBLE CARRY IN EA BITS
007056  062737  000002  011720          ADD     #2,BFADPH       ;BUMP THE PHYSICAL ADDRESS
007064  062737  000002  011714          ADD     #2,DATAD        ;BUMP DATA POINTER
007072  023737  011714  011710          CMP     DATAD,BUFMAX    ;DONE WITH BUFFER?
```

```
2052  007100  101716                BLOS    LOP2            ;BRANCH IF NO
2053
2054  007102  004737  006510        JSR     PC,BUFCLR       ;RESTORE BUFFER
2055  007106  004537  010274        JSR     R5,BUFAD        ;GET NEXT BUFFER
2056  007112  001000                .WORD   1000            ;SIZE =1000
2057  007114  005701                TST     R1              ;DID WE GET ONE?
2058  007116  001674                BEQ     LOOP2           ;BRANCH IF YES
2059  007120  000137  001410        JMP     ENDTST          ;OTHERWISE TERMINATE TEST
```

```
                        .SBTTL          TST3  SPW ADDRESSING

                ;
                ;       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
                ;       UNTIL 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.
                ;
007124  004537  006530          TST3:   JSR     R5,INIT         ;INITIALIZE THE SOFTWARE
007130  005037  011740                  CLR     RECORD          ;SET UP TO RECORD IBM DEV RECOG.
007134  005037  011742                  CLR     FRSID   ;SET TO RECORD FIRST DEV RECOGNIZED
007140  005037  011746                  CLR     DEVCT           ;CLEAR DEVICE COUNTER
007144  004737  006560                  JSR     PC,DXRES        ;RESET THE DX
007150  012737  000400  011726          MOV     #400,CMD        ;INITIALIZE A COMMAND OF ZERO
007156  004537  010274                  JSR     R5,BUFAD        ;GET THE FIRST BUFFER ADDR
007162  102000                  .WORD   102000          ;SPW TO BE ON 2000 BOUNDARY
007164  012737  002000  011706          MOV     #2000,BUFSZE    ;SET UP BUFFER SIZE
007172  004737  006510                  JSR     PC,BUFCLR       ;CLEAR BUFFER
007176  005037  011724          LOOP3:  CLR     DEV             ;START AT DEVICE 0
007202  013700  011704                  MOV     BUFBAS,R0       ;GET SPW TABLE BASE
007206  013720  011724          10$:    MOV     DEV,(R0)+       ;MOV IMMED. STATUS TO SPW EQUAL TO DEV
007212  005237  011724                  INC     DEV             ;INC TO NEXT STATUS = NEXT DEV
007216  022737  000377  011724          CMP     #255.,DEV       ;IS DST FULL?
```

```
2116  007224  103370                  BHIS    10$             ;DO MORE IF IT IS NOT
2117  007226  005037  011724          CLR     DEV             ;START AT DEV 0
2118  007232  013777  011716  002276  LOP3:   MOV     BFBAPH,@DXOSA   ;SET UP OFFSET FOR SPW
2119  007240  053777  011712  002266          BIS     EACUR,@DXCSA    ;SET UP EXT ADDR BITS
2120  007246  004537  007514          JSR     R5,PTYGEN       ;ASSIGN PARITY TO THE DEVICE BYTE
2121  007252  011724                  .WORD   DEV             ;ADDR OF DATA TO ASSIGN PARITY TO
2122  007254  113777  011724  002250          MOVB    DEV,@DXCAR      ;SET DEV TO CUAR
2123  007262  053777  011724  002254          BIS     DEV,@DXMOA      ;PUT DEV ADDR ON BUSOUT
2124  007270  052777  004000  002246          BIS     #ADRO,@DXMOA    ;RAISE ADDR OUT
2125  007276  032777  000001  002244          BIT     #1,@DXCBA       ;IS  ADRECD ON?
2126  007304  001452                  BEQ     20$             ;BR IF ADDR NOT RECOGNIZED
2127  007306  032777  000002  002234          BIT     #2,@DXCBA       ;IS ADRECC ON?
2128  007314  001446                  BEQ     20$             ;BR IF ADDR NOT RECOGNIZED
2129  007316  004537  007554          JSR     R5,SAVDEV       ;GO SAVE RECOGNIZED DEV DATA
2130  007322  052777  060000  002214          BIS     #HLDO!SELO,@DXMOA       ;RAISE SEL OUT, HOLD OUT
2131  007330  042777  004000  002206          BIC     #ADRO,@DXMOA    ;DROP ADRO
2132  007336  043777  011724  002200          BIC     DEV,@DXMOA      ;REMOVE DEVICE FROM BUSO
2133  007344  053777  011726  002172          BIS     CMD,@DXMOA      ;PUT CMD =0 ON BUSOUT
2134  007352  052777  002000  002164          BIS     #CMDO,@DXMOA    ;RAISE CMD OUT
2135  007360  043777  011726  002156          BIC     CMD,@DXMOA      ;REMOVE CMD FROM BUSO
2136  007366  042777  002000  002150          BIC     #CMDO,@DXMOA    ;DROP CMD OUT
2137  007374  123777  011724  002134          CMPB    DEV,@DXOSA      ;IS STATUS EQUAL TO DEV?
2138  007402  001404                  BEQ     10$             ;BR IF OK
2139  007404  004537  010746          JSR     R5,ERRR         ;REPORT ERROR
2140  007410  000004                  .WORD   4               ;ERROR ID
2141  007412  012065                  .WORD   ERR4            ;ERROR MESSAGE ADDR
2142  007414  052777  001000  002122  10$:    BIS     #SRVO,@DXMOA    ;RAISE SERVICE OUT TO RELIEVE STATUS
2143  007422  042777  001000  002114          BIC     #SRVO,@DXMOA    ;DROP SERVICE OUT
2144  007430  000402                  BR      30$
2145  007432  004537  007620          20$:    JSR     R5,DONTSV       ;GO HANDLE RECOGNIZED DEV DATA
2146  007436  004737  006560          30$:    JSR     PC,DXRES        ;RESTORE DX
2147  007442  005237  011724          INC     DEV             ;GO TO NEXT DEV
2148  007446  105737  011724          TSTB    DEV             ;LAST DEV?
2149  007452  001267                  BNE     LOP3            ;BR IF MORE DEVICES
2150  007454  004737  006510          JSR     PC,BUFCLR       ;CLEAR OLD BUFFER BEFORE GETTING NEW ONE
2151  007460  004537  010274          JSR     R5,BUFAD        ;GET NEXT BUFFER BASE
2152  007464  002000                  .WORD   2000            ;DIVISABLE BY 2000
2153  007466  005701                  TST     R1              ;DID WE GET ANOTHER BUFFER?
2154  007470  001642                  BEQ     LOOP3           ;BR IF YES AND DO NEXT ONE
2155  007472  005737  011746          TST     DEVCT           ;TEST IS DONE, DID WE EVER FIND A
2156                                                          ;RECOGNIZED DEVICE?
2157  007476  001004                  BNE     40$             ;BR IF YES
2158  007500  004537  010746          JSR     R5,ERRR         ;ERROR, NO ADREDD AND ADRECD FOR ANY
2159                                                          ;IBM DEVICE NUMBER
2160  007504  000005                  .WORD   5
2161  007506  012114                  .WORD   ERR5            ;ADDR OF ERROR MSG
2162  007510  000137  001410          40$:    JMP     ENDTST          ;TERMINATE TEST
2163
2164
2165
2166
2167
2168                          ;       PARITY ASSIGNMENT SUBROUTINE
2169                          ;
2170                          ;       CALLED AS FOLLOWS:
2171                          ;       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  011740          SAVDEV: TST     RECORD          ;ARE WE RECORDING DEVICES FOUND?
007560  001016                          BNE     20$             ;BR IF NO AND EXIT
007562  005737  011742                  TST     FRSTD           ;IS THIS THE VERY FIRST DEV RECOGNIZED?
007566  001011                          BNE     10$             ;BR IF NO
007570  052737  000001  011742          BIS     #1,FRSTD        ;SET FIRST DEV FOUND SW
007576  113737  011724  011744          MOVB    DEV,FRSTDV      ;SAVE FIRST DEVICE FOUND
007604  012737  000001  011756          MOV     #1,DEVFND       ;SET SWITCH = DEVICE FOUND
007612  005237  011746          10$:    INC     DEVCT           ;COUNT THE DEV
007616  000205                  20$:    RTS     R5


007620  005737  011746          DONTSV: TST     DEVCT           ;HAVE WE FOUND ANY DEV YET?
007624  001403                          BEQ     30$             ;BR IF NO
007626  052737  000001  011740          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  011756          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  012211                  .WORD   T4MSG           ;TO FIND AND IBM DEVICE #
007662  000000                  .WORD   0               ;AND ASSURE GOOD SPW ACCESS CAPABILITY
007664  000137  010270          JMP     END4            ;ABORT TEST
007670  005037  011724  10$:    CLR     DEV             ;CLEAN UP CURRENT DEV NO.
007674  113737  011744  011724  MOVB    FRSTDV,DEV      ;GET THE VALID DEV
007702  004537  007514          JSR     R5,PTYGEN       ;ASSIGN PARITY TO IT
007706  011724                  .WORD   DEV
007710  004537  010274          JSR     R5,BUFAD        ;GET A BUFFER OF 256 BYTES
007714  100400                  .WORD   100400
007716  012737  000400  011706  MOV     #400,BUFSZE     ;SET BUFFER SIZE
007724  004737  006510          JSR     PC,BUFCLR       ;CLEAR THE BUFFER
007730  005000          LOOP4:  CLR     R0              ;START WITH CMD = 0
007732  013701  011704          MOV     BUFBAS,R1       ;GET THE DST BASE
007736  110021          10$:    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  113700  011724          MOVB    DEV,R0          ;GET DEVICE
007756  006300                  ASL     R0              ;MAKE IT A WORD OFFSET
007760  017701  001552          MOV     @DXOSA,R1       ;GET THE SPW BASE
007764  042701  001777          BIC     #1777,R1        ;CLEAN IT UP
007770  050100                  BIS     R1,R0           ;ADD DEV TO OFFSET
007772  005737  011646          TST     KT              ;MM IN USE?
007776  001412                  BEQ     15$             ;BR IF NO
010000  013701  011712          MOV     EACUR,R1        ;GET EXT ADDR BITS
010004  000301                  SWAB    R1              ;POSITION FOR USE IN PAR
010006  006201                  ASR     R1
010010  010137  172352          MOV     R1,@#KIPAR5     ;USE PAR5 TO GET TO SPW TABLE
```

```
2272  010014  042700  040000            BIC     #40000,R0       ;SET SPW ADDR TO USE PAR5
2273  010020  052700  120000            BIS     #120000,R0
2274  010024  013701  011716    15$:    MOV     @FBAPH,R1       ;SET PHY BUFFER BASE
2275  010030  105001                     CLRB    R1              ;IMMED STATUS = 0
2276  010032  010110                     MOV     R1,(R0)         ;SET UP SPW
2277  010034  005037  011726            CLR     CMD             ;START WITH CMD = 0
2278  010040  004537  007514    LOP4:   JSR     R5,PTYGEN       ;GET COMMAND PARITY
2279  010044  011726                     .WORD   CMD
2280  010046  053777  011712  001460    BIS     EACUR,@DXCSA    ;SET UP EXT ADDR BITS
2281  010054  113777  011724  001450    MOVB    DEV,@DXCAA      ;SET DEVICE TO CUAR
2282  010062  053777  011724  001454    BIS     DEV,@DXMOA      ;PUT DEV ADDR ON BUS OUT
2283  010070  052777  004000  001446    BIS     #ADR0,@DXMOA    ;ADDR OUT
2284  010076  052777  060000  001440    BIS     #HLD0!SEL0,@DXMOA       ;SELECT & HOLD OUT
2285  010104  042777  004000  001432    BIC     #ADR0,@DXMOA    ;DROP ADDR OUT
2286  010112  043777  011724  001424    BIC     DEV,@DXMOA      ;DROP DEV FROM BUS
2287  010120  053777  011726  001416    BIS     CMD,@DXMOA      ;COMMAND TO BUS
2288  010126  052777  002000  001410    BIS     #CMD0,@DXMOA    ;COMMAND OUT
2289  010134  043777  011726  001402    BIC     CMD,@DXMOA      ;CLR CMD FROM BUS
2290  010142  042777  002000  001374    BIC     #CMD0,@DXMOA    ;DROP COMMAND OUT
2291  010150  122737  000004  011726    CMPB    #4,CMD          ;IS IT A SENSE COMMAND?
2292  010156  001004                     BNE     5$              ;BR IF NO
2293  010160  105777  001352            TSTB    @DXOSA          ;STATUS FORCED TO ZERO ON SENSE CMD
2294  010164  001411                     BEQ     10$             ;BR IF GOOD
2295  010166  000404                     BR      7$              ;REPORT ERROR
2296  010170  123777  011726  001340    CMPB    CMD,@DXOSA      ;IS STATUS EQUAL DEVICE?
2297  010176  001404                     BEQ     10$             ;BR IF YS
2298  010200  004537  010746    7$:     JSR     R5,ERRR         ;ERROR IN DST ACCESS
2299  010204  000006                     .WORD   6               ;ERROR #
2300  010206  012170                     .WORD   ERR6            ;MESSAGE ADDR
2301  010210  052777  001000  001326    BIS     #SRV0,@DXMOA    ;RELIEVE STATUS
2302  010216  042777  001000  001320    BIC     #SRV0,@DXMOA
2303  010224  004737  006560            JSR     PC,DXRES        ;CLEAR THE DX
2304  010230  005237  011726            INC     CMD             ;TRY NEXT COMMAND
2305  010234  105737  011726            TSTB    CMD             ;DONE IF BACK TO ZERO
2306  010240  001277                     BNE     LOP4            ;BR TO DO NEXT CMD
2307  010242  004737  006510            JSR     PC,BUFCLR       ;CLEAR OLD BUFFER
2308  010246  004537  010274    20$:    JSR     R5,BUFAD        ;GET NEXT BUFFER BASE
2309  010252  000400                     .WORD   400
2310  010254  005701                     TST     R1              ;DID WE GET A BUFFER?
2311  010256  001004                     BNE     END4            ;BR IF NO,END TEST
2312  010260  005737  011716            TST     @FBAPH          ;IS BUFFER BASE 0'S?
2313  010264  001770                     BEQ     20$             ;BR IF YES, THE DX DOES NOT DO A
2314                                                             ;DST FETCH IF SPW ENTRY IS ALL ZEROS
2315  010266  000620                     BR      LOOP4           ;DO NEXT BUFFER
2316  010270  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
                                ;
010274  012500          BUFAD:  MOV     (R5)+,R0        ;GET THE PASSED PARAMETER
010276  005001                  CLR     R1              ;RESET "OUT OF MEMORY" INDICATOR
010300  005700                  TST     R0              ;CHECK FOR INITIALIZE SW
010302  100050                  BPL     NEXTB           ;BRANCH IF NOT TO INITIALIZE
010304  005037  011712          CLR     EACUR           ;START INIT PROCESS BY CLEARING
                                                        ;THE EA BITS
010310  012702  016000          MOV     #ENDALL,R2      ;GET VIRTUAL END OF PROGRAM ADDR
010314  042700  100000          BIC     #100000,R0      ;CLEAR HIGH BIT OF PARAM
010320  060002                  ADD     R0,R2           ;OFFSET ABOVE PROGRAM
010322  162700  000001          SUB     #1,R0           ;DEVELOP A MASK
010326  040002                  BIC     R0,R2           ;TO MAKE BASE DIVISABLE BY PROPER AMT.
010330  010237  011704          MOV     R2,BUFBAS       ;SAVE VIRTUAL BUFFER BASE
010334  010237  011716          MOV     R2,BFBAPH       ;SAVE PHYSICAL BUFFER BASE, SAME AS VIRT
                                                        ;AT THIS POINT.
010340  005737  011646          TST     KT              ;IS KT IN USE?
010344  001421                  BEQ     ENDB            ;BRANCH IF NO TO END OF MODULE
010346  012703  000006          MOV     #6,R3           ;INIT LOOP COUNTER
010352  005037  172354          CLR     @#KIPAR6        ;START TO INITIALIZE KIPAR6
010356  006202          10$:    ASR     R2      ;SHIFT ADDRESS RIGHT 6 PLACES
010360  077302                  SOB     R3,10$          ;TO ALLIGN WITH PAR
010362  010237  172354          MOV     R2,@#KIPAR6     ;SET UP PAR6 TO REFERENCE FIRST BUFFER
010366  042737  177700  011704  BIC     #177700,BUFBAS  ;CLEAR UP VIRTUAL BUFFER ADDR
010374  052737  140000  011704  BIS     #140000,BUFBAS  ;BUFFER ACCESS IS ALWAYS THRU PAR6
010402  012737  000001  177572  MOV     #1,@#SR0        ;TURN ON MM
010410  063700  011704  ENDB:   ADD     BUFBAS,R0       ;FIND TOP OF CURRENT BUFFER
010414  005200                  INC     R0      ;
010416  010037  011710          MOV     R0,BUFMAX       ;SAVE VIRT END OF BUFFER
010422  000550                  BR      ENDB1           ;DONE WITH INITIALIZATION
                                ;
                                ;
                                ;       COME HERE IF NEXT BUFFER ADDR IS WANTED
                                ;
010424  060037  011704  NEXTB:  ADD     R0,BUFBAS       ;INCR TO NEXT BUFFER SPACE
010430  042737  000077  011704  BIC     #77,BUFBAS      ;CLEAR POSSIBLE LOW ORDER BITS
010436  060037  011716          ADD     R0,BFBAPH       ;STEP PHYSICAL ADDR
010442  042737  000077  011716  BIC     #77,BFBAPH      ;CLEAR POSSIBLE LOW ORDER BITS
```

```
2373  010450  103007                        BCC     2$              ;BR IF NO CARRY
2374  010452  062737  000010  011712        ADD     #10,EACUR       ;STEP EXTENDED ADDR BITS
2375  010460  023737  011712  011700        CMP     EACUR,EABITS    ;HAVE THE EXT ADR BITS GONE TOO HIGH?
2376  010466  101125                        BHI     30$             ;BR IF YES TO SET ALL DONE SW AND EXIT
2377  010470  005737  011646        2$:     TST     KT              ;IS MEM MNG IN USE?
2378  010474  001511                        BEQ     20$             ;BRANCH IF NO
2379  010476  032737  020000  011704        BIT     #20000,BUFBAS   ;WAS THERE A CARRY TO ACCESS NEXT PAR?
2380  010504  001417                        BEQ     10$             ;BR IF NO CARRY
2381  010506  062737  000200  172354        ADD     #200,@#KIPAR6   ;TAKE A 4K STEP
2382  010514  023737  172354  011644        CMP     @#KIPAR6,EASIZE ;CHECK FOR TOO HIGH
2383  010522  103402                        BLO     5$              ;BRANCH IF OK
2384  010524  005201                        INC     R1              ;SET THE "ALL DONE" SWITCH
2385  010526  000506                        BR      ENDB1           ;EXIT
2386  010530  042737  177700  011704  5$:   BIC     #177700,BUFBAS  ;START BUFFER BASE OVER
2387  010536  052737  140000  011704        BIS     #140000,BUFBAS  ;ACCESS THROUGH PAR6
2388  010544  013703  011704        10$:    MOV     BUFBAS,R3       ;GET THE BASE ADDR
2389  010550  060003                        ADD     R0,R3           ;CALCULATE TOP OF BUFFER
2390  010552  032703  020000                BIT     #20000,R3       ;DIT IT CARRY ACROSS A PAR BOUNDARY
2391  010556  001322                        BNE     NEXTB           ;BR IS YES.  GO BACK AND GET A BETTER ADDR
2392  010560  010337  011710                MOV     R3,BUFMAX       ;SAVE TOP OF BUFFER ADDR
2393  010564  013703  011716                MOV     BFBAPH,R3       ;GET PHYSICAL BUF BASE
2394  010570  060003                        ADD     R0,R3           ;ADD BUFFER SIZE TO GET BUF TOP
2395  010572  023737  011712  011700        CMP     EACUR,EABITS    ;ARE WE IN THE TOP PHYSICAL BANK?
2396  010600  001006                        BNE     12$             ;BR IF NO
2397  010602  020337  011702                CMP     R3,PHYMAX       ;ARE WE OVER THE TOP OF MEM?
2398  010606  103055                        BHIS    30$             ;BR IF YES, SET DONE SW
2399  010610  020327  000004                CMP     R3,#4           ;WAS THERE A CARRY OUT OF TOP BANK?
2400  010614  101452                        BLOS    30$             ;BR IF YES, WE MUST BE DONE
2401  010616  005737  011712        12$:    TST     EACUR           ;ARE WE IN THE FIRST BANK?
2402  010622  001014                        BNE     3$              ;BR IF NO
2403  010624  013703  011716                MOV     BFBAPH,R3       ;GET PHYSICAL BUFFER BASE
2404  010630  060003                        ADD     R0,R3           ;FIND TOP OF PHY BUFFER
2405  010632  103674                        BCS     NEXTB           ;BR IF CARRY OVER 32K BNDRY
2406                                                                ;AND GO BACK AND GET A BUFFER WHICH DOESN'T
2407                                                                ;STRADDEL A 32K BOUNDARY
2408  010634  020337  011640                CMP     R3,MSIZE        ;ARE WE IN THE LOADER AREA?
2409  010640  101405                        BLOS    3$              ;BR IF OK
2410  010642  023737  011716  011642        CMP     BFBAPH,MMAX     ;IS BUFFER ABOVE LOADERS?
2411  010650  103001                        BHIS    3$              ;BRANCH IF WE'RE OK
2412  010652  000664                        BR      NEXTB           ;GO BACK AND GET A BETTER BUFFER ADDR
2413  010654  012777  014000  000654  3$:   MOV     #SPW,@DX0SA     ;SET UP SPW ADDR
2414  010662  005037  011760                CLR     SPWMVD          ;CLEAR SPW BASE ADDR MOVED SWITCH
2415  010666  020327  014000                CMP     R3,#SPW         ;IS TOP OF BUFFER IN SPW?
2416  010672  101411                        BLOS    15$             ;BR IF OK
2417  010674  023727  011716  016000        CMP     BFBAPH,#SPW+2000        ;IS BUFFER BASE IN TT?
2418  010702  103005                        BHIS    15$             ;BR IF NO
2419  010704  012777  016000  000624        MOV     #SPW+2000,@DX0SA        ;MOV THE SPW
2420  010712  005237  011760                INC     SPWMVD          ;SET SPW ADDR MOVED TO ALTERNATE SWITCH
2421  010716  000412                        BR      ENDB1           ;GO TO END OF MODULE
2422  010720  013737  011704  011710  20$:   MOV     BUFBAS,BUFMAX   ;START TO GET BUFMAXIMUM
2423  010726  060037  011710                ADD     R0,BUFMAX       ;ADD IN BUFFER SIZE
2424  010732  023737  011710  011640        CMP     BUFMAX,MSIZE    ;SEE IF WE'RE OUT OF MEMORY
2425  010740  101401                        BLOS    ENDB1           ;BR IF OK
2426  010742  005201                30$:    INC     R1              ;SET OUT OF MEMORY SWITCH
2427  010744  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.
                                        ;
                                        ;
010746  032777  100000  170164  ERRR:   BIT     #SW15,@SWR      ;HALT ON ERROR ON?
010754  001401                          BEQ     5$              ;BR IF NO
010756  000000                          HALT                    ;SW15 ON AND ERROR OCCURRED.

010760  013737  011662  011672  5$:     MOV     CTESTN,ERR      ;SET ERROR DETECTED SW
010766  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
010770  012746  010776                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
010774  000002                          RTI                     ;;POP NEW PC AND PS
010776                          64$:
010776  005737  011674                  TST     LOCK            ;ARE WE LOCKED ON ERROR?
011002  001407                          BEQ     10$             ;BR IF NO
011004  032777  040000  170126          BIT     #SW14,@SWR      ;SCOPE LOOP SW ON?
011012  001413                          BEQ     15$             ;BR IF NO
011014  013705  011676                  MOV     LOCKAD,R5       ;GET THE LOOP ADDR
011020  000205                          RTS     R5              ;GO TO IT
011022  032777  040000  170110  10$:    BIT     #SW14,@SWR      ;SCOPE LOOP ON?
011030  001404                          BEQ     15$             ;BR IF NO
011032  013737  011662  011674          MOV     CTESTN,LOCK     ;SET LOCKED ON ERR SW
011040  000402                          BR      20$
011042  005037  011674          15$:    CLR     LOCK            ;TURN OF LOCKED SW
011046  010537  011730          20$:    MOV     R5,ERRPC        ;SAVE ERR PC
011052  012537  011732                  MOV     (R5)+,ERRID     ;SAVE ERROR ID
011056  012537  011734                  MOV     (R5)+,ERRMS     ;SAVE ERROR MESSAGE ADDR
011062  032777  020000  170050          BIT     #SW13,@SWR      ;INHIBIT PRINT ON?
011070  001402                          BEQ     25$             ;BR IF NO
011072  000137  011514                  JMP     97$             ;TERMINATE MODULE
011076  004537  005134          25$:    JSR     R5,PRINT        ;PRINT ERRPC
011102  012763                          .WORD   ERRPC           ;ADDR OF MESSAGE
011104  000000                          .WORD   0               ;NO SPECIAL PRINT CONTROLS
011106  162737  000004  011730          SUB     #4,ERRPC        ;BACKUP ADDR TO CORRECT VALUE
011114  004537  005134                  JSR     R5,PRINT        ;PRINT THE PC NUMBER ITSSELF
011120  011730                          .WORD   ERRPC
011122  100001                          .WORD   100001          ;NO CRLF, CONVERT TO ASCII
011124  004537  005134                  JSR     R5,PRINT        ;PRINT ERROR NUMBER ON SAME LINE
011130  012776                          .WORD   ERRNO
011132  000001                          .WORD   1               ;INHIBIT CRLF
011134  004537  005134                  JSR     R5,PRINT        ;PRINT THE OCT
011140  011732                          .WORD   ERRID
011142  100001                          .WORD   100001
```

```
2484  011144  013737  011734  011156          MOV     ERRMS,30$       ;GET MESSAGE POINTER
2485  011152  004537  005134                  JSR     R5,PRINT        ;PRINT THE UNIQUE MESSAGE
2486  011156  000000                  30$:    .WORD   0               ;FILLED FROM TWO LINES UP
2487  011160  000000                          .WORD   0               ;NO SPECIAL CONTROLS
2488  011162  032777  010000  167750          BIT     #SW12,@SWR      ;SHORT ERROR REPORT ON?
2489  011170  001145                          BNE     95$             ;BR IF YES, WE'RE DONE
2490  011172  004537  005134                  JSR     R5,PRINT        ;PRINT TEST NO.
2491  011176  013012                          .WORD   ERTN
2492  011200  000000                          .WORD   0
2493  011202  004537  005134                  JSR     R5,PRINT        ;PRINT THE OCTAL TEST NO.
2494  011206  011662                          .WORD   CTESTN
2495  011210  100001                          .WORD   100001
2496  011212  022737  000005  011732          CMP     #5,ERRID        ;IS IT ERROR #5?
2497  011220  001531                          BEQ     95$             ;BR IF YES, REST OF STUFF IS MEANINGLESS
2498  011222  004537  005134                  JSR     R5,PRINT        ;PRINT VIRT ADDR
2499  011226  013025                          .WORD   ERADR
2500  011230  000000                          .WORD   0
2501  011232  004537  005134                  JSR     R5,PRINT        ;PRINT VIRT ADDR OCTAL
2502  011236  011704                          .WORD   BUFBAS
2503  011240  100001                          .WORD   100001
2504  011242  005737  011646                  TST     KT              ;IS MEM MGT IN USE?
2505  011246  001431                          BEQ     90$             ;BR IF NO WE'RE DONE
2506  011250  004537  005134                  JSR     R5,PRINT        ;PRINT PHYSICAL ADDR
2507  011254  013060                          .WORD   ERADPH
2508  011256  000000                          .WORD   0
2509  011260  004537  005134                  JSR     R5,PRINT        ;PRINT PHYSICAL ADDR OCTAL
2510  011264  011716                          .WORD   BFBAPH
2511  011266  100001                          .WORD   100001
2512  011270  004537  005134                  JSR     R5,PRINT        ;PRINT EA BITS
2513  011274  013105                          .WORD   EABIT
2514  011276  000000                          .WORD   0
2515  011300  013737  011712  011736          MOV     EACUR,EAERR     ;GET THE EA BITS
2516  011306  006237  011736                  ASR     EAERR           ;POSITION BITS
2517  011312  006237  011736                  ASR     EAERR           ;  TO BITS 1:0
2518  011316  006237  011736                  ASR     EAERR
2519  011322  004537  005134                  JSR     R5,PRINT        ;PRINT THE OCTAL EA BITS
2520  011326  011736                          .WORD   EAERR
2521  011330  100001                          .WORD   100001
2522  011332  023727  011732  000003  90$:    CMP     ERRID,#3                ;IS IT ERROR NO. 3?
2523  011340  001010                          BNE     93$             ;BR IF NO
2524  011342  004537  005134                  JSR     R5,PRINT        ;PRINT DATA ADDRESS ASCII
2525  011346  013126                          .WORD   DATAA
2526  011350  000000                          .WORD   0
2527  011352  004537  005134                  JSR     R5,PRINT        ;PRINT DATA ADDR OCT
2528  011356  011714                          .WORD   DATAD
2529  011360  100001                          .WORD   100001
2530  011362  023727  011732  000004  93$:    CMP     ERRID,#4        ;IS IT ERROR #4?
2531  011370  001004                          BNE     94$             ;BR IF NO
2532  011372  113737  011724  011754          MOVB    DEV,EXP         ;GET DEV #
2533  011400  000416                          BR      96$
2534  011402  023727  011732  000006  94$:    CMP     ERRID,#6        ;IS IT ERROR #6?
2535  011410  001035                          BNE     95$             ;BR IF NO
2536  011412  122737  000004  011726          CMPB    #4,CMD          ;WAS IT A SENSE CMD?
2537  011420  001003                          BNE     98$             ;BR IF NO
2538  011422  005037  011754                  CLR     EXP             ;EXPECTED STATUS IS ZERO FOR SENSE
2539  011426  000403                          BR      96$
```

```
2540  011430  113737  011726  011754  98$:   MOVB    CMD,EXP          ;GET EXPECTED CMD
2541  011436  117737  000074  011752  96$:   MOVB    @DXOSA,ACTU      ;GET ACTUAL STATUS
2542  011444  004537  005134                  JSR     R5,PRINT         ;PRINT ACTUAL MESSAGE
2543  011450  013251                          .WORD   ACTUMS
2544  011452  000000                          .WORD   0
2545  011454  004537  005134                  JSR     R5,PRINT         ;PRINT STATUS OCTAL
2546  011460  011752                          .WORD   ACTU
2547  011462  100001                          .WORD   100001
2548  011464  004537  005134                  JSR     R5,PRINT         ;PRINT EXPECTED STATUS
2549  011470  013263                          .WORD   EXPMS
2550  011472  000001                          .WORD   1
2551  011474  004537  005134                  JSR     R5,PRINT         ;PRINT EXPECTED OCTAL
2552  011500  011754                          .WORD   EXP
2553  011502  100001                          .WORD   100001
2554  011504  004537  005134          95$:    JSR     R5,PRINT         ;SPACE THE PAPER
2555  011510  013301                          .WORD   CRLF1
2556  011512  000000                          .WORD   0
2557  011514  022737  001566  000042  97$:    CMP     #SENDAD,@#42     ;UNDER ACT?
2558  011522  001001                          BNE     99$              ;BR IF NO
2559  011524  000000                          HALT                     ;HALT ON ACT ERROR
2560
2561  011526  000205                  99$:    RTS     R5               ;RETURN FROM MODULE
```

```
2562                                    ;
2563                                    ;
2564                                    ;       DX ACCESS TABLES
2565                                    ;
2566                                    ;
2567   011530  000000                   DXDSA:  .WORD   0       ;DEVICE STATUS REG ADDR.
2568   011532  000000                   DXCAA:  .WORD   0       ;COMMAND AND ADDR REG ADDR.
2569   011534  000000                   DXCSA:  .WORD   0       ;CONTROL UNIT STATUS REG ADDR
2570   011536  000000                   DXOSA:  .WORD   0       ;OFFSET AND STATUS REG ADDR.
2571   011540  000000                   DXBAA:  .WORD   0       ;BUS ADDRESS REG ADDR
2572   011542  000000                   DXBCA:  .WORD   0       ;BYTE COUNT REG ADDR
2573   011544  000000                   DXMOA:  .WORD   0       ;MAINTENANCE OUT REG ADDR.
2574   011546  000000                   DXMIA:  .WORD   0       ;MAINTENANCE IN REG ADDR
2575   011550  000000                   DXCBA:  .WORD   0       ;CONTROL BITS REG ADDR
2576   011552  000000                   DXNDA:  .WORD   0       ;NPR DATA REG ADDR
2577   011554  000000                   DXESA:  .WORD   0       ;EXTRA SIGNALS REG ADDR
2578   011556  000000                   DXMOBA: .WORD   0       ;MAINTENANCE OUT BUFFERED REG ADDR
2579   011560  000000                   DXES1A: .WORD   0       ;EXTRA EXTRA SIGNALS REG ADDR.
2580                                    ;       REGS TO BE OUTPUT TO DX
2581   011562  000000                   DXDSO:  .WORD   0       ;DEVICE STATUS REG OUT
2582   011564  000000                   DXCAO:  .WORD   0       ;COMMAND AND ADDR REG OUT
2583   011566  000000                   DXCSO:  .WORD   0       ;CONTROL UNIT STATUS REG OUT
2584   011570  000000                   DXOSO:  .WORD   0       ;OFFSET AND STATUS REG OUT
2585   011572  000000                   DXBAO:  .WORD   0       ;BUS ADDR REG. OUT
2586   011574  000000                   DXMOO:  .WORD   0       ;MAINT OUT REG ADDR
2587   011576  000000                   DXBCO:  .WORD   0
2588
```

```
                                        ;
                                        ;
                                        ;       DEFAULT PARAMETER TABLE
011600                                  DPAR:
011600  000001                          DTESTN: .WORD   1       ;STARTING TEST NUMBER
011602  176200                          DUBA:   .WORD   176200  ;DEFAULT UNIBUS BASE ADDR
011604  000300                          DVECT:  .WORD   300     ;DEFAULT VECTOR ADDR
011606  000004                          DPRIOR: .WORD   4       ;DEFAULT DX PRIORITY LEVEL
```

```
2598                                    ;
2599                                    ;
2600                                    ;       INPUT PARAMETER LIMITS TABLE
2601                                    ;
2602                                    ;
2603   011610                           LPAR:
2604   011610   000001                  LTESTN: 1               ;LOWEST POSSIBLE TEST NUMBER
2605   011612   000004                  HTESTN: 4               ;HIGHEST POSSIBLE TEST NUMBER
2606   011614   160000                  LUBA:   160000          ;LOWEST POSSIBLE UNIBUS ADDRESS
2607   011616   177756                  HUBA:   177756          ;HIGHEST POSSIBLE UNIBUS ADDRESS
2608   011620   000040                  LVEC:   40              ;LOWEST POSSIBLE DX VECTOR ADDRESS
2609   011622   000776                  HVEC:   776             ;HIGHEST POSSIBLE DX VECTOR ADDRESS
2610   011624   000000                  LPRIOR: 0               ;LOWEST POSSIBLE FOR LEVEL
2611   011626   000006                  HPRIOR: 6               ;HIGHEST POSSIBLE FOR LEVEL
```

```
                                ;
                                ;
                                ;       ACTIVE PARAMETER TABLE
                                ;
                                ;       THIS TABLE CONTAINS CONFIGURATION PARAMETER CURRENTLY
                                ;       IN USE.
011630                          APAR:
011630  000000                  ATESTN: .WORD   0       ;STARTING TEST NO.
011632  000000                  AUBA:   .WORD   0       ;DX UNIBUS BASE ADDR.
011634  000000                  AVECT:  .WORD   0       ;DX VECTOR ADDRESS
011636  000000                  APRIOR: .WORD   0       ;DX PRIORITY LEVEL
011640  000000                  MSIZE:  .WORD   0       ;ADDR OF LAST WORD OF MEM, NOT INCLUDING LOADER, KT OFF
011642  000000                  MMAX:   .WORD   0       ;LAST WORD OF MEM, KT OFF
011644  000000                  EASIZE: .WORD   0       ;LOWEST INVALID SETTING OF PAR
011646  000000                  KT:     .WORD   0       ;=0 NO KT IN USE =1 KT IN USE
011650  000000                  FIRSTP: .WORD   0       ;0 IF FIRST PASS, 1 IF NOT FIRST PASS
011652  177564                  TPS:    .WORD   177564  ;TTY PRINTER STATUS REG ADDR
011654  177566                  TPB:    .WORD   177566  ;TTY PRINTER BUFFER REG ADDR
011656  177560                  TKS:    .WORD   177560  ;TTY KEYBOARD STATUS REG ADDR
011660  177562                  TDB:    .WORD   177562  ;TTY KEYBOARD BUFFER.
011662  000000                  CTESTN: .WORD   0       ;CURRENT TEST NUMBER
011664  000000                  PARMV:  .WORD   0       ;0 IF PARMS NOT VALID, 1 IF THEY ARE VALID
011666  000001                  COUNT:  .WORD   1       ;ITERATION COUNT CONSTANT
011670  000000                  CNT:    .WORD   0       ;ITERATION VARIABLE
011672  000000                  ERR:    .WORD   0       ;IF ERROR DETECTED, CONTAINS TN
011674  000000                  LOCK:   .WORD   0       ;SW=TN IF LOOP SW 14 ERR DETECTED
011676  000000                  LOCKAD: .WORD   0       ;LOCK ON ERROR ADDR
011700  000000                  EABITS: .WORD   0       ;HIGHEST VALID SETTING OF ADDR 17 16
011702  000000                  PHYMAX: .WORD   0       ;LOWEST INVALID PHYSICAL ADDR, LOW 16 BITS
011704  000000                  BUFBAS: .WORD   0       ;START OF CURRENT WORKING BUFFER, VIRTUAL
011706  000000                  BUFSZE: .WORD   0       ;SIZE OF CURRENT BUFFER IN OCT BYTES
011710  000000                  BUFMAX: .WORD   0               ;MAX VIRTUAL ADDR FOR THIS KT SETTING
011712  000000                  EACUR:  .WORD   0               ;CURRENT SETTING OF EA BITS
011714  000000                  DATAD:  .WORD   0       ;CURRENT DATA ADDR
011716  000000                  BFBAPH: .WORD   0       ;CURRENT PHYSICAL BUFFER BASE ADDR
011720  000000                  BFADPH: .WORD   0               ;CURRENT PHYSICAL DATA ADDR
011722  000000                  SPWPHY: .WORD   0       ;PHYSICAL ADDR OF SPW START
011724  000000                  DEV:    .WORD   0       ;CURRENT IBM DEVICE NO.
011726  000400                  CMD:    .WORD   400     ;CURRENT IBM COMMAND WITH PARITY BIT
011730  000000                  ERRPC:  .WORD   0       ;PC WHERE ERROR WAS DETECTED
011732  000000                  ERRID:  .WORD   0               ;ERROR NUMBER
011734  000000                  ERRMS:  .WORD   0               ;ADDR OF UNIQUE ERROR MESSAGE
011736  000000                  EAERR:  .WORD   0               ;EABITS ON ERROR
011740  000000                  RECORD: .WORD   0               ;SWITCH TO CONTROL DEV REC RECORDING
011742  000000                  FRSTD:  .WORD   0               ;FIRST DEVICE RECOGNIZED SW
011744  000000                  FRSTDV: .WORD   0               ;FIRST DEVICE RECOGNIZED NUMBER
011746  000000                  DEVCT:  .WORD   0               ;FIRST DEVICE GROUP COUNT
011750  000000                  DEVMSS: .WORD   0               ;DEVICE MESSAGE PRINTED SW
011752  000000                  ACTU:   .WORD   0       ;ACTUAL DATA
011754  000000                  EXP:    .WORD   0               ;EXPECTED DATA
011756  000000                  DEVFND: .WORD   0               ;= 1 IF DEVICE EVER RECOGNIZED
011760  000000                  SPWMVD: .WORD   0       ;= 1 IF SPW TAPLE IN ALTERNATE LOCATION
```

```
2664                            .NLIST  BIN
2665   011762   ERR1:   .ASCIZ  /FAST NPR DATA ERROR/
2666   011770
2667   011776
2668   012004
2669   012006   ERR2:   .ASCIZ  /NO DX INTERRUPT/
2670   012014
2671   012022
2672
2673   012026   ERR3:   .ASCIZ  /DX ADDRESSING ERROR, DATA XFER/
2674   012034
2675   012042
2676   012050
2677   012056
2678   012064
2679   012065   ERR4:   .ASCIZ  /SPW ACCESS, ADDR ERROR/
2680   012072
2681   012100
2682   012106
2683   012114   ERR5:   .ASCIZ  /NO IBM DEV ADR RECOGNIZED. CHECK A20 MODULE/
2684   012122
2685   012130
2686   012136
2687   012144
2688   012152
2689   012160
2690   012166
2691   012170   ERR6:   .ASCIZ  /DST ACCESS ERROR/
2692   012176
2693   012204
2694   012211   T4MSG:  .ASCIZ  /TST3 MUST BE RUN FIRST FOR TST 4 TO WORK/
2695   012216
2696   012224
2697   012232
2698   012240
2699   012246
2700   012254
2701   012262   HEADER: .ASCII  'MD-11-DZDXJA  DX11B MEMORY ADDRESSING TEST'<15><12>
2702   012270
2703   012276
2704   012304
2705   012312
2706   012320
2707   012326
2708   012334
2709   012336           .ASCII  'TYPE:  <D>, DEFAULT PARAMETERS'<15><12>
2710   012344
2711   012352
2712   012360
2713   012366
2714   012374
2715   012375           .ASCII  '       <P>, PREVIOUS PARAMETERS'<15><12>
2716   012402
2717   012410
2718   012416
2719   012424
```

```
2720  012430          .ASCII  '      <S>, SELECT PARAMETERS'<15><12>
2721  012436
2722  012444
2723  012452
2724  012460
2725  012461          .ASCIZ  '      <N>, START AT THIS TEST NO.'<15><12>
2726  012466
2727  012474
2728  012502
2729  012510
2730  012516
2731  012520  INVM:   .ASCIZ  'INVALID ENTRY'
2732  012526
2733  012534
2734  012536  INVPRM: .ASCIZ  'NEED MORE PARMS'
2735  012544
2736  012552
2737  012556  PROMPT: .ASCIZ  'D,P,S,N? '
2738  012564
2739  012570  TESTN:  .ASCIZ  'STARTING TEST NO. '
2740  012576
2741  012604
2742  012612
2743  012613  BASEA:  .ASCIZ  'BASE ADDRESS '
2744  012620
2745  012626
2746  012631  VECTA:  .ASCIZ  'VECTOR ADDRESS '
2747  012636
2748  012644
2749  012651  PRILV:  .ASCIZ  'DX PRIORITY LEVEL '
2750  012656
2751  012664
2752  012672
2753  012674  SETSW:  .ASCIZ  'SET SWITCHES '
2754  012702
2755  012710
2756  012712  BELL:   .ASCIZ  <207> 'END PASS'
2757  012720
2758  012724  BADP:   .ASCIZ  'INPUT PARAMETER OUTSIDE LIMITS'
2759  012732
2760  012740
2761  012746
2762  012754
2763  012762
2764  012763  ERPC:   .ASCIZ  'ERROR PC: '
2765  012770
2766  012776  ERNO:   .ASCIZ  '  ERR NO.: '
2767  013004
2768  013012  ERTN:   .ASCIZ  'TEST NO.: '
2769  013020
2770  013025  ERADR:  .ASCIZ  'VIRT ADDR OF BUFFER BASE: '
2771  013032
2772  013040
2773  013046
2774  013054
2775  013060  ERADPH: .ASCIZ  'PHY ADDR OF BUFFER: '
```

```
013066
013074
013102
013105  EABIT:  .ASCIZ  'EA BITS IN OCT: '
013112
013120
013126  DATAA:  .ASCIZ  'VIRT ADDR OF ACCESS ERROR: '
013134
013142
013150
013156
013162  DEVMS:  .ASCIZ  '1ST IBM DEV RCGNZD.  OCTAL:  '
013170
013176
013204
013212
013220  DEVCMS: .ASCIZ  'SIZE OF 1ST DEV GROUP:  '
013226
013234
013242
013250
013251  ACTUMS: .ASCIZ  'ACTUAL:  '
013256
013263  EXPMS:  .ASCIZ  '  EXPECTED:  '
013270
013276
013301  CRLF1:  .ASCIZ  / /<15><12>
                        .LIST   BIN
        013306          .EVEN
                        ;INBUF MUST BE ON AN EVEN WORD BOUNDARY
013306     012  INBUF:  .BYTE   12      ;KEYBOARD INPUT BUFFER
        014000          .=14000
014000          SPW:                    ;START OF STATIONARY SPW TABLE USED ONLY TO
                                        ;FIELD UNEXPECTED ERRORS.
014000  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014002  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014004  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014006  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014010  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014012  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014014  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014016  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014020  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014022  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014024  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014026  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014030  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014032  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014034  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014036  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014040  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014042  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014044  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014046  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014050  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014052  000002         .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
```

```
2832  014054  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2833  014056  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2834  014060  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2835  014062  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2836  014064  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2837  014066  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2838  014070  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2839  014072  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2840  014074  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2841  014076  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2842  014100  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2843  014102  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2844  014104  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2845  014106  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2846  014110  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2847  014112  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2848  014114  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2849  014116  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2850  014120  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2851  014122  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2852  014124  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2853  014126  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2854  014130  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2855  014132  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2856  014134  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2857  014136  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2858  014140  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2859  014142  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2860  014144  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2861  014146  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2862  014150  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2863  014152  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2864  014154  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2865  014156  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2866  014160  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2867  014162  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2868  014164  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2869  014166  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2870  014170  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2871  014172  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2872  014174  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2873  014176  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2874  014200  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2875  014202  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2876  014204  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2877  014206  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2878  014210  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2879  014212  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2880  014214  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2881  014216  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2882  014220  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2883  014222  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2884  014224  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2885  014226  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2886  014230  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2887  014232  000002          .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
```

```
2888  014234  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2889  014236  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2890  014240  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2891  014242  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2892  014244  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2893  014246  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2894  014250  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2895  014252  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2896  014254  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2897  014256  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2898  014260  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2899  014262  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2900  014264  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2901  014266  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2902  014270  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2903  014272  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2904  014274  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2905  014276  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2906  014300  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2907  014302  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2908  014304  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2909  014306  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2910  014310  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2911  014312  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2912  014314  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2913  014316  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2914  014320  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2915  014322  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2916  014324  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2917  014326  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2918  014330  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2919  014332  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2920  014334  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2921  014336  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2922  014340  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2923  014342  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2924  014344  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2925  014346  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2926  014350  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2927  014352  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2928  014354  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2929  014356  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2930  014360  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2931  014362  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2932  014364  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2933  014366  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2934  014370  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2935  014372  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2936  014374  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2937  014376  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2938  014400  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2939  014402  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2940  014404  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2941  014406  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2942  014410  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2943  014412  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
```

```
2944  014414  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2945  014416  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2946  014420  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2947  014422  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2948  014424  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2949  014426  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2950  014430  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2951  014432  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2952  014434  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2953  014436  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2954  014440  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2955  014442  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2956  014444  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2957  014446  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2958  014450  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2959  014452  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2960  014454  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2961  014456  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2962  014460  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2963  014462  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2964  014464  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2965  014466  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2966  014470  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2967  014472  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2968  014474  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2969  014476  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2970  014500  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2971  014502  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2972  014504  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2973  014506  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2974  014510  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2975  014512  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2976  014514  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2977  014516  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2978  014520  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2979  014522  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2980  014524  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2981  014526  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2982  014530  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2983  014532  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2984  014534  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2985  014536  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2986  014540  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2987  014542  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2988  014544  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2989  014546  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2990  014550  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2991  014552  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2992  014554  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2993  014556  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2994  014560  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2995  014562  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2996  014564  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2997  014566  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2998  014570  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
2999  014572  000002                .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
```

```
014574 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014576 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014600 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014602 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014604 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014606 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014610 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014612 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014614 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014616 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014620 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014622 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014624 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014626 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014630 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014632 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014634 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014636 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014640 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014642 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014644 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014646 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014650 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014652 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014654 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014656 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014660 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014662 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014664 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014666 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014670 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014672 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014674 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014676 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014700 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014702 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014704 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014706 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014710 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014712 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014714 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014716 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014720 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014722 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014724 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014726 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014730 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014732 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014734 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014736 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014740 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014742 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014744 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014746 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014750 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
014752 000002          .WORD   2        ;STATUS OF UNIT CHECK WITH NO DST ACCESS
```

```
3056  014754  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3057  014756  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3058  014760  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3059  014762  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3060  014764  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3061  014766  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3062  014770  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3063  014772  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3064  014774  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3065  014776  000002                    .WORD   2       ;STATUS OF UNIT CHECK WITH NO DST ACCESS
3066  015000                    TT:                     ;START OF STATIONARY TUMBLE TABLE
3067  016000                    ENDALL:
3068          000001                    .END
```

```
ACTU    011752          2541*   2546    2660#
ACTUMS  013251          2543    2797#
ADR0  = 004000           610#   2124    2131    2283    2285
APAR    011630          2618#
APRIOR  011636           829*    847*    885*   2622#
ATESTN  011630           743     826*    844*    864*    900*   2619#
AUBA    011632           827*    845*    871*    916    2620#
AVECT   011634           828*    846*    878*   1875    2018   2621#
BADP    012724          1535    2758#
BASEA   012613           978    2743#
BELL    012712           789    2756#
BFADPH  011720          2024*   2028    2049*   2647#
BFBAPH  011716          1883    2024    2118    2274    2312   2349*  2371*  2372*  2393   2403   2410   2417   2510
                        2646#
BIT0  = 000001           554#
BIT00 = 000001           544#    554
BIT01 = 000002           543#    553
BIT02 = 000004           542#    552
BIT03 = 000010           541#    551
BIT04 = 000020           540#    550
BIT05 = 000040           539#    549
BIT06 = 000100           538#    548
BIT07 = 000200           537#    547
BIT08 = 000400           536#    546
BIT09 = 001000           535#    545
BIT1  = 000002           553#
BIT10 = 002000           534#
BIT11 = 004000           533#
BIT12 = 010000           532#
BIT13 = 020000           531#
BIT14 = 040000           530#
BIT15 = 100000           529#
BIT2  = 000004           552#
BIT3  = 000010           551#
BIT4  = 000020           550#
BIT5  = 000040           549#
BIT6  = 000100           548#
BIT7  = 000200           547#
BIT8  = 000400           546#
BIT9  = 001000           545#
BPTVEC= 000014           561#
BUFAD   010274          1872    1913    2016    2055    2107   2151   2250   2308   2337#
BUFBAS  011704          1895    1949    2023    2112    2255   2348*  2358*  2359*  2361   2369*  2370*  2379   2386*
                        2387*   2388    2422    2502    2641#
BUFCLR  006510          1880    1912    1949#   2022    2054   2110   2150   2253   2307
BUFMAX  011710          2051    2363*   2392*   2422*   2423*  2424   2643#
BUFSZE  011706          1878*   1896    1950    2021*   2109*  2252*  2642#
CHECKP  004766           865     872     879     886     901   1504#
CKSWR = 104406          1806#
CMD     011726          2106*   2133    2135    2277*   2279   2287   2289   2291   2296   2304*  2305   2536   2540
                        2650#
CMD0  = 002000           608#   1931    1932    2134    2136   2288   2290
CNT     011670           742*    766*   2635#
COUNT   011666           742    2634#
CR    = 000015           469#    959    1640    1650
CRLF  = 000200           470#   1611    1650
```

```
CRLF1   013301            1566    2555    2802#
CTESTN  011662             743*    746     763*    764     2449    2462    2494    2632#
DATAA   013126            2525    2782#
DATAD   011714            2023*   2039    2045*   2050*   2051    2528    2645#
DDISP = 177570             476#    674     727
DEV     011724            2111*   2113    2114*   2115    2117*   2121    2122    2123    2132    2137    2147*   2148    2201
                          2246*   2247*   2249    2261    2281    2282    2286    2532    2649#
DEVCMS  013220             779    2792#
DEVCT   011746             770     782    2104*   2155    2203*   2209    2658#
DEVFND  011756            2202*   2240    2662#
DEVMS   013162             773    2787#
DEVMSS  011750             740*    768     784*   2659#
DISPLA  001142             674#    727*    735*
DISPRE  000174             618#    735
DONE  = 000200             604#   1904    1981    2046
DONTSV  007620            2145    2209#
DPAR    011600            2592#
DPRIOR  011606             829     847    1006    2596#
DSWR  = 177570             475#    673     726
DTESTN  011600             826     844     973    2593#
DUBA    011602             827     845     984    2594#
DVECT   011604             828     846     995    2595#
DXBAA   011540            1883*   2028*   2571#
DXBAO   011572            2585#
DXBCA   011542            1922*   2572#
DXBCO   011576            2587#
DXCAA   011532            2122*   2281*   2568#
DXCAO   011564            2582#
DXCBA   011550            1990    2125    2127    2575#
DXCSA   011534            1882*   1904*   1906*   1927*   1981*   1982*   1983*   1984*   2025*   2046*   2048*   2119*   2280*
                          2569#
DXCSO   011566            2583#
DXDSA   011530             915    2567#
DXDSO   011562            2581#
DXESA   011554            1905*   1933*   1980*   2047*   2577#
DXES1A  011560            2579#
DXGO    006366            1884    1921#   2029
DXMIA   011546            2574#
DXMOA   011544            1921*   1929*   1930*   1931*   1932*   2123*   2124*   2130*   2131*   2132*   2133*   2134*   2135*
                          2136*   2142*   2143*   2282*   2283*   2284*   2285*   2286*   2287*   2288*   2289*   2290*   2301*
                          2302*   2573#
DXMOBA  011556            2578#
DXMOO   011574            1881*   1897    1907    1909*   1921    2026*   2586#
DXNDA   011552            2576#
DXOSA   011536            1986*   1989*   2118*   2137    2263    2293    2296    2413*   2419*   2541    2570#
DXOSO   011570            2584#
DXRES   006560            1871    1980#   1993    2015    2105    2146    2239    2303
EABIT   013105            2513    2779#
EABITS  011700            1380*   1398*   1399*   1400*   1401*   2375    2395    2639#
EACUR   011712            1882    2025    2119    2268    2280    2341*   2374*   2375    2395    2401    2515    2644#
EAERR   011736            2515*   2516*   2517*   2518*   2520    2654#
EASIZE  011644            1379*   1388*   1389*   2382    2625#
EMTVEC= 000030             564#
ENDALL  016000            2343    3067#
ENDB    010410            2352    2361#
ENDB1   010744            2364    2385    2421    2425    2427#
```

```
ENDPRM  002270            823    923#
ENDTST  001410            755#  1917    2059    2162    2316
ENDT1   006350           1913#
END1    001576            792    798#
END4    010270           2245   2311    2316#
EOP     001452            768#
ERADPH  013060           2507   2775#
ERADR   013025           2499   2770#
ERNO    012776           2479   2766#
ERPC    012763           2472   2764#
ERR     011672            745*  1972*   2449*   2636#
ERRID   011732           2466*  2482    2496    2522    2530    2534    2652#
ERRMS   011734           2467*  2484    2653#
ERRPC   011730           2465*  2474*   2476    2651#
ERRR    010746           1899   1939    2042    2139    2158    2298    2445#
ERRVEC= 000004            557#   724     725*    736*   1424    1425    1429*  1434*  1446*  1451*  1460*  1470*  1471*
ERR1    011762           1901   2665#
ERR2    012006           1941   2669#
ERR3    012026           2044   2673#
ERR4    012065           2141   2679#
ERR5    012114           2161   2683#
ERR6    012170           2300   2691#
ERTN    013012           2491   2768#
EXP     011754           2532*  2538*   2540*   2552    2661#
EXPMS   013263           2549   2799#
FIRSTP  011650            739*   821     922*   2627#
FRSTD   011742           2103*  2198    2200*   2656#
FRSTDV  011744            776   2201*   2247    2657#
GETPRM  001600            741    821#
GNS   = ****** U          617   1799    1800    1801    1802    1804    1806    1807    1808    1809
GO      001346            743#   767
GO1     006206           1882#  1910
GTSWR = 104405           1804#
HEADER  012262            837   2701#
HLDO  = 040000            607#  1929    1930    2130    2284
HPRIOR  011626           1530   2611#
HT    = 000011            467#  1609    1650
HTESTN  011612            764   1509    2605#
HUBA    011616           1516   2607#
HVEC    011622           1523   2609#
INASC   002532            893    955     969     980     991    1002    1027#
INBUF   013306           1032*  1038*   1039    2806#
INIT    006530           1870   1967#   2014    2101    2238
INPRM   002272            839    861     869     876     883     898     948#
INTERR  006506           1889   1935*   1942*   1944#   1974*   2033
INTR1   006230           1876   1889#
INTR2   006766           2019   2033#
INVM    012520            855   2731#
INVPRM  012536            908   2734#
IOTVEC= 000020            562#
KIPAR0= 172340            595#   1410    1438
KIPAR1= 172342            596#
KIPAR2= 172344            597#
KIPAR3= 172346            598#
KIPAR4= 172350            599#
KIPAR5= 172352            600#  2271#
```

```
KIPAR6= 172354            601#  2354*  2357*  2381*  2382
KIPAR7= 172356            602#  1454
KIPDR0= 172300            584#
KIPDR1= 172302            585#
KIPDR2= 172304            586#
KIPDR3= 172306            587#
KIPDR4= 172310            588#
KIPDR5= 172312            589#
KIPDR6= 172314            590#
KIPDP7= 172316            591#
KT      011646            785   1368*  1387*  2266   2351   2377   2504   2626#
LF    = 000012            468#  1644   1650
LOCK    011674            744*  2454   2462*  2464*  2637#
LOCKAD  011676           1973*  2458   2638#
LOOP1   006166           1878#  1916
LOOP2   006710           2021#  2058
LOOP3   007176           2111#  2154
LOOP4   007730           2254#  2315
LOP2    006736           2025#  2052
LOP3    007232           2118#  2149
LOP4    010040           2278#  2306
LPAR    011610           2603#
LPRIOR  011624           1528   2610#
LTESTN  011610           1507   2604#
LUBA    011614           1514   2606#
LVEC    011620           1521   2608#
MMAX    011642           1369*  2410   2624#
MMVEC = 000250            573#
MSIZE   011640           1370*  1373*  1378*  2408   2424   2623#
NEXTB   010424           2340   2369#  2391   2405   2412
PARMV   011664            833    851   860*   905    921*   2633#
PHYMAX  011702           1390*  1391*  1392*  1393*  1394*  1395*  1396*  1397*  2397  2640#
PIRQ  = 177772            474#
PIRQVE= 000240            568#
PRILV   012551           1000   2749#
PRINT   005134            772    775   778    781    788    836    854    890    907   952   966   977   988
                          999   1534   1558#  2242   2471   2475   2478   2481   2485  2490  2493  2498  2501
                         2506   2509   2512   2519   2524   2527   2542   2545   2548  2551  2554
PROMPT  012556            953   2737#
PR0   = 000000            491#
PR1   = 000040            492#
PR2   = 000100            493#
PR3   = 000140            494#
PR4   = 000200            495#
PR5   = 000240            496#
PR6   = 000300            497#
PR7   = 000340            498#
PS    = 177776            471#   472
PSW   = 177776            472#
PTYGEN  007514           2120   2179#  2248   2278
PTY2    007522           2182#  2186
PTY3    007530           2183   2185#
PTY4    007546           2188   2191#
PTY5    007552           2190   2192#
PWRVEC= 000024            563#   720*   721*  1815*  1816*  1825*  1831*  1843*  1844*
RDCHR = 104407           1244   1807#
```

```
RDLIN = 104410          1031    1319    1808#
RDOCT = 104411          1037    1809#
RECORD  011740          2102*   2196    2211*   2655#
RESTAR  001334           741#    798    1848
RESVEC= 000010           558#
SAVDEV  007554          2129    2196#
SEL0  = 020000           606#   1929    1930    2130    2284
SETSW   012674           891    2753#
SIZE    004212           913    1366#
SOSIEN= 000004           605#   1905    1933    2047
SPW     014000          1986    1989    2413    2415    2417    2419    2808#
SPWMVD  011760          1987    2414*   2420*   2663#
SPWPHY  011722          2648#
SRV0  = 001000           609#   2142    2143    2301    2302
SR0   = 177572           577#    787*   1435    1450*   1457*   2360*
SR1   = 177574           578#
SR2   = 177576           579#
SR3   = 172516           580#   1447*
STACK = 001100           462#    716
START   001164           621     709#   1085    1168
STKLMT= 177774           473#
SWR     001140           673#    714     726*    728     734*    761     1088    1127    1182*   1381    1823    1836*   2445
                        2456    2460    2468    2488
SWREG   000176           619#    734    1088    1127    1150
SW0   = 000001           526#
SW00  = 000001           516#    526
SW01  = 000002           515#    525
SW02  = 000004           514#    524
SW03  = 000010           513#    523
SW04  = 000020           512#    522
SW05  = 000040           511#    521
SW06  = 000100           510#    520
SW07  = 000200           509#    519
SW08  = 000400           508#    518
SW09  = 001000           507#    517
SW1   = 000002           525#
SW10  = 002000           506#   1381
SW11  = 004000           505#
SW12  = 010000           504#   2488
SW13  = 020000           503#   2468
SW14  = 040000           502#   2456    2460
SW15  = 100000           501#   2445
SW2   = 000004           524#
SW3   = 000010           523#
SW4   = 000020           522#
SW5   = 000040           521#
SW6   = 000100           520#
SW7   = 000200           519#
SW8   = 000400           518#
SW9   = 001000           517#
TBITVE= 000014           559#
TDB     011660          2631#
TESTN   012570           967    2739#
TKS     011656          2630#
TKVEC = 000060           566#   1065*   1066*
TPB     011654          2629#
```

```
TPS     011652          2628#
TPVEC = 000064           567#
TRAPVE= 000034           565#    718*    719*
TRTVEC= 000014           560#
TST     001364           746#    762     765
TSTAB   001376           748     749#
TST1    006134           750    1870#
TST2    006656           751    2014#
TST3    007124           752    2101#
TST4    007636           753    2238#
TT      015000          3066#
TYPE  = 104401          1082    1094    1148    1149    1152    1163    1173    1184    1203    1251    1257    1262    1266
                        1271    1272    1274    1277    1281    1348    1350    1565    1573    1614    1711    1799#   1845
TYPOC = 104402          1151    1570    1800#
TYPON = 104404          1802#
TYPOS = 104403          1801#
T4MSG   012211          2243    2694#
VECTA   012631           989    2746#
SAUTOB  001134           670#    1141    1299
SBDAOR  001122           665#
SBDDAT  001126           667#
SBELL   004004          1094    1291#
SCHARC  005422          1616*   1626*   1633    1642*   1647#
SCKSWR  003106          1127#   1806
SCMTAG  001100           653#    711     712
SCM3  = 000000           683#
SCNTLC  004010          1082    1163    1292#
SCNTLG  004022          1148    1294#
SCNTLU  004015          1173    1266    1293#
SCORE   004666          1433    1460#
SCRLF   001161           684#    1184    1271    1291    1353    1615    1650
SCROUT  004716          1460    1467#
SENDAD  001566           628     794#    831     2557
SERFLG  001103           656#
SERMAX  001115           662#
SERRPC  001116           663#
SERRTB  001164           701#
SERTTL  001112           660#
SFILLC  001156           681#    1619    1650
SFILLS  001155           680#    1650
SGDAOR  001120           664#
SGDDAT  001124           666#
SGTSWR  003176          1149#   1804
SHD   = 000003           457     458
SHIOCT  004210          1341*   1352#
SICNT   001104           657#
SILLUP  006116          1815    1831    1850#
SINTAG  001135           671#    1146*   1165    1185    1299
SITEMB  001114           661#
SKTNEX  004660          1434    1459#
SKTOUT  004650          1451    1456#
SKT11   004516          1366*   1383*   1385    1432#   1436*   1459*
SLF     001162           685#    1281    1291    1353    1650
SLPADR  001106           658#
SLPERR  001110           659#
SLSTAD  004762          1369    1370    1371    1472*   1479#
```

```
SLSTBK  004764          1388    1390    1398    1473*   1480#
$MAIL = ****** U         738    1603
$MNEW   004040          1152    1297#
$MSWR   004027          1149    1295#
$NULL   001154           679#   1621    1650
$OCNT   005650          1683*   1712*   1725#
$OMODE  005652          1678*   1682*   1687    1690*   1701*   1727#
$PASS   001100           654#
$POWER  006124          1846    1853#
$PWRAD  006112          1848#
$PWRON  005752           720    1815#   1843
$PWRMG  006106          1846#
$PWRUP  006024          1825    1831#
$QUES   001160           683#   1203    1274    1291    1350    1353    1650
$ROCHR  003450          1216#   1807
$RODEC= ****** U        1810
$ROLIN  003540          1239#   1808
$RODCT  004052          1314#   1809
$RDSZ = 000010          1232#
$R2A  = ****** U        1810
$SAVRE= ****** U        1810
$SAVR6  006122          1824*   1832    1833*   1834*   1852#
$SETUP= 000114           709#    717     718     720     722    1086    1091    1092    1122    1299
$SIZE   004460          1367    1384    1420#
$SIZEX  004722          1458    1468#
$STUP = 177777           709#
$SVPC = 000204           626#    631
$SWR  = 160000           457     458#    683     1849
$TKB    001146           676#   1047    1067    1078    1103    1131    1158
$TKCNT  002602          1048#   1062*   1092    1109*   1223    1225*
$TKINT  002620          1027    1062#   1083    1144
$TKQEN= 002620          1052#   1117    1228
$TKQIN  002604          1049#   1063*   1064    1115*   1116*   1117    1119*
$TKQOU  002606          1050#   1064*   1226    1227*   1228    1230*
$TKQSR  002610          1051#   1063    1119    1230
$TKS    001144           675#   1047    1068*   1099*   1101    1107*   1129    1145*   1155    1167*   1187*
$TKSRV  002670          1065    1078#
$TN   = 000001           457#
$TPB    001152           678#   1639*   1650
$TPFLG  001157           682#   1597    1650
$TPS    001150           677#   1637    1650
$TRAP   005654           718    1747#
$TRAP2  005714          1762#   1798
$TRP  = 000012          1791#   1800#   1801#   1802#   1803#   1804    1805#   1806    1807#   1808#   1809#   1810#
$TRPAD  005726          1756    1798#
$TSTNM  001102           655#
$TTYIN  003774          1241    1242    1254    1272    1286    1290#
$TYPBN= ****** U        1803
$TYPDS= ****** U        1803
$TYPE   005206          1597#   1791    1799
$TYPEC  005356          1189    1618    1625    1632    1637#   1638
$TYPEX  005424          1643    1645    1648#
$TYPOC  005452          1681#   1800
$TYPON  005466          1680    1683#   1802
$TYPOS  005426          1676#   1801
$OFILL  005651          1677*   1681*   1691    1726#
```

```
.     = 016000          613#   617#   626    627#   629#   631#   632#   652#   686    715    1047   1051#  1052
                        1053   1290#  1291   1292   1299#  1353   1650   1827   1851   2804#  2807#
```

```
CKSWR     444#
COMMEN    569#
ENDCOM    569#
ERROR     463#
ESCAPE    569#
GETPRI    569#  1428
GETSWR    444#   569#
MULT      569#
NEWTST    569#
POP       446#   569#  1038  1342  1575  1836  1837
PUSH      446#   569#  1316  1558  1817  1823
REPORT    569#
SCOPE     464#
SETPRI    445#   569#   755  1219  1923  1967  2450
SETTRA   1766#  1791   1800  1801  1802  1804  1806  1807  1808  1809
SETUP     569#   709
SKIP      569#
SLASH     569#
SPACE     569#
STARS     569#   624   648   686  1046  1122  1137  1208  1232  1302  1406  1582  1653  1741  1813
         1829
SWRSU     569#   722#
TRMTRP   1788#
TYPBIN    569#
TYPDEC    569#
TYPMAN    569#
TYPNUM    569#
TYPOCS    569#
TYPOCT    569#  1150
TYPTXT    569#
$$CMRE    633#
$$CMTM    640#
$$ESCA    569#
$$NEWT    569#
$$SET    1772#  1791   1800  1801  1802  1804  1806  1807  1808  1809
$$SKIP    569#
.EQUAT    445#   459
.HEADE    444#   447
.KT11     446#   569
.SETUP    444#   709
.$ACT1    446#   622
.$CATC    444#   611
.$CMTA    444#   633
.$POWE    445#  1811
.$RDOC    445#  1300
.$READ    445#  1044
.$SIZE    446#  1404
.$TRAP    445#  1739
.$TYPE    444#  1580
.$TYPO    445#  1651

. ABS.  016000     000

 ERRORS DETECTED:  0
```

DEFAULT GLOBALS GENERATED: 0

DZDXJA,DZDXJA/CRF/SOL=DZDXJA.P11[400,3274],SYSMAC.SML[400,1066]
RUN-TIME: 13 7 .7 SECONDS
RUN-TIME RATIO: 749/21=34.2
CORE USED: 20K (40 PAGES)