PDP11

UNIBUS SYSTEMS EXERCISER

CZKUAD0

DIAGNOSTIC

AH-8856D-MC

COPYRIGHT © 75–78

FICHE 1 OF 1

JUL 1978

digital

MADE IN USA

Identification

Product Code: AC-8855D-MC

Product Name: CZKUAD0 Unibus Systems Exerciser Diagnostic

Date :   1-APRIL-78

Maintainer: Diagnostic Group

Author: Manuel Soares

MODIFIED BY: BILL SCHLITZKUS

Table of Contents

1.0 ABSTRACT

This program was created to test PDP-11's CPU interface circuitry. It
uses the Unibus Exerciser(s) (UBE) to insure proper operation by
simulating peripherals which wou.d require the 11-CPU to produce the
necessary signals. It should be noted that the UBE is a powerful tool
and if it is not programmed correctly could cause various problems on
the Bus.

2.0 REQUIREMENTS

2.1 Hardware

This program assumes the following in proper working condition:
1. The Unibus, 2. Memory (8K minimum), and 3. UBE(s) (4 maximum).
If a fourth UBE is being used, its time delay should be set at 100us
to prevent latency problems in one of the tests.

With two or more UBE(s), all should have W1 jumpers except the one
furthest electrically from the CPU. If there are more than 4 UBE(s)
on the Unibus the program is not responsible for any problems which
might occur , since it is programmed to handle a maximum of only 4.

2.2 Software

After loading the program the starting address must be 200, so that
the first time through, the available UBE(s) are determined. In
addition if one or more UBE(s) are added or removed, the program again
must be started at 200.   Otherwise, to avoid duplicating some
printouts, the program can be restarted at address 220.

****
A SOFTWARE HALT CAN BE CAUSED BY DEPRESSING CONTROL-H ON THE CONSOLE.
IF THE PROGRAM IS HALTED THIS WAY, AND THE PROGRAM IS RESTARTED,
DEPRESS ANY CONSOLE KEY TO REMOVE THE SOFTWARE HALT CONDITION.
****

3.0 PROGRAM DESCRIPTION

This program was assembled with MACY11 using PDP-11 Maindec Sysmac
package .

3.1 Switch Options

The use of thes program on processors having a software switch register necessitates operator interaction: the operator must set up location 176 with the switch register values desired .

```
    Switch          Use
    ------          --,

     15        Halt on Error
     14        Loop on test
     13        Inhibit error typeouts
     11        Inhibit iterations
     10        Bell on error
      9        Loop on error
      8        Loop on test in SWR<5:0>
    NOTE:    If you wish to inhibit all typing except ''end of pass''
    you must put down switch 7, after loading 200.
      6          WHEN SET, INHIBIT TEST 14
```

3.2 Test 1 through Test 16

TEST 1 - No Bus grants issued with processor at higher priority than bus request. This test is to insure that any request is not honored as long as the processor is at the same or higher priority.

TEST 2 - Issuing of non-processor grants and arbitration tests. This test will request on NPR through BR4 levels with the processor status initially at level 7 and make sure the device exercises an NPG to do a fun 1-dati, then the requests will be repeated while sequentially lowering the processor status from 7 to 0 to allow arbitration of all requests and the issuing of NPG.

TEST 3 - Issuing of Bus grant 7 and arbitration tests. This test will arbitrate for a BG7. The requests will be on levels BR7 thru BR4, doing fun 1-dati transfers, and the processor status lowered sequentially from 7 to 0.

TEST 4 - Issuing of Bus grant 6 and arbitration tests. This test will arbitrate for a BG6, the requests will be on levels BR6 thru BR4, doing fun 1-dati transfers, and the processor status lowered sequentially from 6 to 0.

TEST 5 - Issuing of Bus grant 5 and arbitration tests. This test will arbitrate for a BG5, the requests will be on levels BR5 thru BR4, doing fun 1-dati transfers, and the processor status lowered sequentially from 5 to 0.

TEST 6 - Issuing of Bus grant 4 and arbitration tests.   This test
          will arbitrate for a BG4, the requests will be on level BR4,
          doing func 1-dati transfers, and the processor status I
          lowered sequentially from 4 to 0.

TEST 7 - CPU test for no sack time out. This test will check that
          the CPU times out and drops a grant if no sack signal is
          received. If the CPU time out is inoperative, the Bus
          exerciser will time out and send the sack signal to prevent
          a Bus hang and set an error flag in CR2.

TEST 10 - CPU test for receiving sack. This test is to insure that
          the CPU can receive the sack signal; The time delay will be
          set on device 1 and several dati transfers made.   If there
          is not bus late error, the CPU received sack correctly. It
          is assumed that dev 1 time delay is set for 10us.

TEST 11 - Passing of grants and interrupt test. This test will set
          off   all   available devices simultaneously whose only
          functions will be to interrupt, the requests will all be at
          level 7 so that the device closest to tha CPU should receive
          BG7 first and interrupt first, the next closest should
          interrupt next and so on.

TEST 12 - Address lines (14 - 17) check. This test will check Bus
          address lines 14 thru 17 by doing a fun 1-dati-npr to those
          addresses. If the addresses don't exist the interrupt
          routine will ignore any no ssyn error.

TEST 13 - CPU test for ACLO/DCLO sequence.   This test checks the
          assertion of ACLO and DCLO and that the CPU traps to the
          correct service routine. If this program is running under
          ACT11 this test will be skipped.

TEST 14 - Parity error test. This test will cause parity error and
          checks that the CPU traps to the correct vector.
          THIS TEST IS SKIPPED ON MACHINES THAT DON'T HAVE THE SXT INSTRUCTION
          (EG., 11/05 AND 11/20).
          THIS TEST SHOULD BE DESELECTED IF THE MEMORY PARITY OPTION
          IS NOT PRESENT OR NOT ENABLED.

          SW06=1   INHIBIT TEST 14

TEST 15 - Multitransfers I. This test will cause any Bus exercisers,
          up to 4, to create a lot of traffic on the Bus and check
          that the CPU can handle it;   all devices are set off
          simultaneously.

TEST 16 - Multitransfers II. This test will have the available
          exercisers doing various transfers and/or interrupts at
          different request levels to further check CPU handling
          capabilities.

TEST 17 - DUMMY END OF PROGRAM. This portion of the program
is   just   to   see   if   " H" has   been typed on the console to
cause a program halt. If there is no " H" the program
continues by jumping to $EOP (end-of-pass routine).
IF THE PROGRAM IS HALTED THIS WAY, AND THE PROGRAM IS RESTARTED,
DEPRESS ANY CONSOLE KEY TO REMOVE THE SOFTWARE HALT CONDITION.

3.3 Sysmac Routines

The 'END OF PASS ROUTINE' thru 'Power Down and Up Routines', as listed
in the program listing, are the Sysmac package macros. They are
called out in the source program, some with arguments and some
without, and are expanded in the listing. Some macros are necessary
for the operation of others, so for a complete explanation of all
available Sysmac Macros see PDP-11 Maindec Sysmac Package (DZQAC-B-D).

4.0 ERROR REPORTING

The minimum amount of information given when an error occurs is the PC
of the error call and the Test number in which it occurred. Other
pertinent data will by typed out depending on the test being run at
that time.

     1

```
    2      167400                      $SWR=167400
    3      000300                      $SWRMK=300
    4                          .TITLE   UNIBUS EXERCISER
    5                          ;*COPYRIGHT (C) MARCH, 75
    6                          ;*DIGITAL EQUIPMENT CORP.
    7                          ;*MAYNARD, MASS. 01754
    8                          ;*
    9                          ;*PROGRAM BY M.SOARES
   10                          ;*
   11                          ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
   12                          ;*PACKAGE (MAINDEC-11-DZQAC-B1),AUG 29,1975.
   13                          ;*
   14      000001             $TN=1
   15
   16                          .SBTTL   OPERATIONAL SWITCH SETTINGS
   17                          ;*
   18                          ;*      SWITCH                 USE
   19                          ;*      ------        --------------------
   20                          ;*        15          HALT ON ERROR
   21                          ;*        14          LOOP ON TEST
   22                          ;*        13          INHIBIT ERROR TYPEOUTS
   23                          ;*        11          INHIBIT ITERATIONS
   24                          ;*        10          BELL ON ERROR
   25                          ;*         9          LOOP ON ERROR
   26                          ;*         8          LOOP ON TEST IN SWR<5:0>
   27                          ;*         6          WHEN SET, INHIBIT TEST 14
   28
   29                          .SBTTL   BASIC DEFINITIONS
   30
   31                          ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
   32      001100             STACK=  1100
   33                          .EQUIV   EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
   34                          .EQUIV   IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
   35      177776             PS=     177776          ;;PROCESSOR STATUS WORD
   36                          .EQUIV   PS,PSW
   37      177774             STKLMT= 177774          ;;STACK LIMIT REGISTER
   38      177772             PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
   39      177570             DSWR=   177570          ;;HARDWARE SWITCH REGISTER
   40      177570             DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
   41
   42                          ;*GENERAL PURPOSE REGISTER DEFINITIONS
   43      000000             R0=     %0              ;;GENERAL REGISTER
   44      000001             R1=     %1              ;;GENERAL REGISTER
   45      000002             R2=     %2              ;;GENERAL REGISTER
   46      000003             R3=     %3              ;;GENERAL REGISTER
   47      000004             R4=     %4              ;;GENERAL REGISTER
   48      000005             R5=     %5              ;;GENERAL REGISTER
   49      000006             R6=     %6              ;;GENERAL REGISTER
   50      000007             R7=     %7              ;;GENERAL REGISTER
   51      000006             SP-     %6              ;;STACK POINTER
   52      000007             PC=     %7              ;;PROGRAM COUNTER
   53
   54                          ;*PRIORITY LEVEL DEFINITIONS
   55      000000             PR0=    0               ;;PRIORITY LEVEL 0
   56      000040             PR1=    40              ;;PRIORITY LEVEL 1
   57      000100             PR2=    100             ;;PRIORITY LEVEL 2
```

```
   58      000140              PR3=     140              ;;PRIORITY LEVEL 3
   59      000200              PR4=     200              ;;PRIORITY LEVEL 4
   60      000240              PR5=     240              ;;PRIORITY LEVEL 5
   61      000300              PR6=     300              ;;PRIORITY LEVEL 6
   62      000340              PR7=     340              ;;PRIORITY LEVEL 7
   63
   64                          ;*"SWITCH REGISTER" SWITCH DEFINITIONS
   65      100000              SW15=    100000
   66      040000              SW14=    40000
   67      020000              SW13=    20000
   68      010000              SW12=    10000
   69      004000              SW11=    4000
   70      002000              SW10=    2000
   71      001000              SW09=    1000
   72      000400              SW08=    400
   73      000200              SW07=    200
   74      000100              SW06=    100
   75      000040              SW05=    40
   76      000020              SW04=    20
   77      000010              SW03=    10
   78      000004              SW02=    4
   79      000002              SW01=    2
   80      000001              SW00=    1
   81                          .EQUIV   SW09,SW9
   82                          .EQUIV   SW08,SW8
   83                          .EQUIV   SW07,SW7
   84                          .EQUIV   SW06,SW6
   85                          .EQUIV   SW05,SW5
   86                          .EQUIV   SW04,SW4
   87                          .EQUIV   SW03,SW3
   88                          .EQUIV   SW02,SW2
   89                          .EQUIV   SW01,SW1
   90                          .EQUIV   SW00,SW0
   91
   92                          ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
   93      100000              BIT15=   100000
   94      040000              BIT14=   40000
   95      020000              BIT13=   20000
   96      010000              BIT12=   10000
   97      004000              BIT11=   4000
   98      002000              BIT10=   2000
   99      001000              BIT09=   1000
  100      000400              BIT08=   400
  101      000200              BIT07=   200
  102      000100              BIT06=   100
  103      000040              BIT05=   40
  104      000020              BIT04=   20
  105      000010              BIT03=   10
  106      000004              BIT02=   4
  107      000002              BIT01=   2
  108      000001              BIT00=   1
  109                          .EQUIV   BIT09,BIT9
  110                          .EQUIV   BIT08,BIT8
  111                          .EQUIV   BIT07,BIT7
  112                          .EQUIV   BIT06,BIT6
  113                          .EQUIV   BIT05,BIT5
```

```
114                                  .EQUIV  BIT04,BIT4
115                                  .EQUIV  BIT03,BIT3
116                                  .EQUIV  BIT02,BIT2
117                                  .EQUIV  BIT01,BIT1
118                                  .EQUIV  BIT00,BIT0
119
120                                  ;*BASIC "CPU" TRAP VECTOR ADDRESSES
121        000004                    ERRVEC= 4                 ;;TIME OUT AND OTHER ERRORS
122        000010                    RESVEC= 10                ;;RESERVED AND ILLEGAL INSTRUCTIONS
123        000014                    TBITVEC=14                ;;"T" BIT
124        000014                    TRTVEC= 14                ;;TRACE TRAP
125        000014                    BPTVEC= 14                ;;BREAKPOINT TRAP (BPT)
126        000020                    IOTVEC= 20                ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
127        000024                    PWRVEC= 24                ;;POWER FAIL
128        000030                    EMTVEC= 30                ;;EMULATOR TRAP (EMT) **ERROR**
129        000034                    TRAPVEC=34                ;;"TRAP" TRAP
130        000060                    TKVEC=  60                ;;TTY KEYBOARD VECTOR
131        000064                    TPVEC=  64                ;;TTY PRINTER VECTOR
132        000240                    PIRQVEC=240               ;;PROGRAM INTERRUPT REQUEST VECTOR
133
134                                  .SBTTL  MEMORY MANAGEMENT DEFINITIONS
135
136                                  ;*KT11 VECTOR ADDRESS
137
138        000250                    MMVEC=  250
139
140                                  ;*KT11 STATUS REGISTER ADDRESSES
141
142        177572                    SR0=    177572
143        177574                    SR1=    177574
144        177576                    SR2=    177576
145        172516                    SR3=    172516
146
147                                  ;*USER "I" PAGE DESCRIPTOR REGISTERS
148
149        177600                    UIPDR0= 177600
150        177602                    UIPDR1= 177602
151        177604                    UIPDR2= 177604
152        177606                    UIPDR3= 177606
153        177610                    UIPDR4= 177610
154        177612                    UIPDR5= 177612
155        177614                    UIPDR6= 177614
156        177616                    UIPDR7= 177616
157
158                                  ;*USER "I" PAGE ADDRESS REGISTERS
159
160        177640                    UIPAR0= 177640
161        177642                    UIPAR1= 177642
162        177644                    UIPAR2= 177644
163        177646                    UIPAR3= 177646
164        177650                    UIPAR4= 177650
165        177652                    UIPAR5= 177652
166        177654                    UIPAR6= 177654
167        177656                    UIPAR7= 177656
168
169                                  ;*KERNAL "I" PAGE DESCRIPTOR REGISTERS
```

```
 170
 171            172300                      KIPDR0= 172300
 172            172302                      KIPDR1= 172302
 173            172304                      KIPDR2= 172304
 174            172306                      KIPDR3= 172306
 175            172310                      KIPDR4= 172310
 176            172312                      KIPDR5= 172312
 177            172314                      KIPDR6= 172314
 178            172316                      KIPDR7= 172316
 179
 180                                        ;*KERNAL ''I'' PAGE ADDRESS REGISTERS
 181
 182            172340                      KIPAR0= 172340
 183            172342                      KIPAR1= 172342
 184            172344                      KIPAR2= 172344
 185            172346                      KIPAR3= 172346
 186            172350                      KIPAR4= 172350
 187            172352                      KIPAR5= 172352
 188            172354                      KIPAR6= 172354
 189            172356                      KIPAR7= 172356
 190
 191
 192                                        .SBTTL  TRAP CATCHER
 193
 194            000000                           .=0
 195                                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
 196                                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
 197                                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
 198            000174                           .=174
 199  000174   000000                       DISPREG: .WORD  0              ;;SOFTWARE DISPLAY REGISTER
 200  000176   000000                       SWREG:   .WORD  0              ;;SOFTWARE SWITCH REGISTER
 201            000200                           .=200
 202  000200   005037   001174              CLR     $TMP0         ;MAKE SURE TMP0=0
 203  000204   000137   001756              JMP     @#START       ;JUMP TO START
 204

      ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
      ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
      ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
              ;*WHEN LOADING THE PROGRAM FOR THE FIRST TIME, OR ANY TIME
              ;*AFTER ALTERING THE # OF EXERCISERS ON THE BUS,
              ;*YOU MUST START AT LOCATION 200 AND
              ;*RESTART AT LOCATION 220 ONLY IF YOU DO NOT WISH
              ;*TO SIZE MEMORY AND TYPE OUT DEV ADDRESSES AGAIN
      ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *
      ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *
      ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *

 218            000220                           .=220
 219  000220   012737   000777   001174     MOV     #777,$TMP0    ;TMP0 IS INDICATOR FOR RESTART
 220  000226   000137   001756              JMP     @#START       ;JUMP TO START
 221
 222                                        ;*************************************************************
 223
 224                                        .SBTTL  ACT11 HOOKS
 225                                        ;HOOKS REQUIRED BY ACT11
```

```
226              000232                          $SVPC=.                      ;SAVE PC
227              000046                          .=46
228   000046     015636                          $ENDAD                       ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
229              000052                          .=52
230   000052     040000                          .WORD   40000                ;;2)SET LOC.52 TO 40000
231              000232                          .=$SVPC                      ;; RESTORE PC
232
```

`

```
233                                  ;****************************************************************
234
235                                  .SBTTL   COMMON TAGS
236
237                                  ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
238                                  ;*USED IN THE PROGRAM.
239
240          001100                           .=1100
241  001100                  $CMTAG:                          ;;START OF COMMON TAGS
242  001100  000000          $PASS:   .WORD  0                ;;CONTAINS PASS COUNT
243  001102     000          $TSTNM:  .BYTE  0                ;;CONTAINS 1.IE TEST NUMBER
244  001103     000          $ERFLG:  .BYTE  0                ;;CONTAINS ERROR FLAG
245  001104  000000          $ICNT:   .WORD  0                ;;CONTAINS SUBTEST ITERATION COUNT
246  001106  000000          $LPADR:  .WORD  0                ;;CONTAINS SCOPE LOOP ADDRESS
247  001110  000000          $LPERR:  .WORD  0                ;;CONTAINS SCOPE RETURN FOR ERRORS
248  001112  000000          $ERTTL:  .WORD  0                ;;CONTAINS TOTAL ERRORS DETECTED
249  001114     000          $ITEMB:  .BYTE  0                ;;CONTAINS ITEM CONTROL BYTE
250  001115     001          $ERMAX:  .BYTE  1                ;;CONTAINS MAX. ERRORS PER TEST
251  001115  000000          $ERRPC:  .WORD  0                ;;CONTAINS PC OF LAST ERROR INSTRUCTION
252  001120  000000          $GDADR:  .WORD  0                ;;CONTAINS ADDRESS OF 'GOOD' DATA
253  001122  000000          $BDADR:  .WORD  0                ;;CONTAINS ADDRESS OF 'BAD' DATA
254  001124  000000          $GDDAT:  .WORD  0                ;;CONTAINS 'GOOD' DATA
255  001126  000000          $BDDAT:  .WORD  0                ;;CONTAINS 'BAD' DATA
256  001130  000000                   .WORD  0                ;;RESERVED--NOT TO BE USED
257  001132  000000                   .WORD  0
258  001134  000000                   .WORD  0
259  001136  177570          SWR:     .WORD  DSWR             ;;ADDRESS OF SWITCH REGISTER
260  001140  177570          DISPLAY: .WORD  DDISP            ;;ADDRESS OF DISPLAY REGISTER
261  001142  177560          $TKS:           177560           ;;TTY KBD STATUS
262  001144  177562          $TKB:           177562           ;;TTY KBD BUFFER
263  001146  177564          $TPS:           177564           ;;TTY PRINTER STATUS REG. ADDRESS
264  001150  177566          $TPB:           177566           ;;TTY PRINTER BUFFER REG. ADDRESS
265  001152     000          $NULL:   .BYTE  0                ;;CONTAINS NULL CHARACTER FOR FILLS
266  001153     002          $FILLS:  .BYTE  2                ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
267  001154     012          $FILLC:  .BYTE  12               ;;INSERT FILL CHARS. AFTER A "LINE FEED"
268  001155     000          $TPFLG:  .BYTE  0                ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
269  001156  000000          $REGAD:  .WORD  0                ;;CONTAINS THE ADDRESS FROM
270                                                            ;;WHICH  ($REG0) WAS OBTAINED
271  001160  000000          $REG0:   .WORD  0                ;;CONTAINS (($REGAD)+0)
272  001162  000000          $REG1:   .WORD  0                ;;CONTAINS (($REGAD)+2)
273  001164  000000          $REG2:   .WORD  0                ;;CONTAINS (($REGAD)+4)
274  001166  000000          $REG3:   .WORD  0                ;;CONTAINS (($REGAD)+6)
275  001170  000000          $REG4:   .WORD  0                ;;CONTAINS (($REGAD)+10)
276  001172  000000          $REG5:   .WORD  0                ;;CONTAINS (($REGAD)+12)
277  001174  000000          $TMP0:   .WORD  0                ;;USER DEFINED
278  001176  000000          $TMP1:   .WORD  0                ;;USER DEFINED
279  001200  000000          $TMP2:   .WORD  0                ;;USER DEFINED
280  001202  000000          $TMP3:   .WORD  0                ;;USER DEFINED
281  001204  000000          $TMP4:   .WORD  0                ;;USER DEFINED
282  001206  000000          $TMP5:   .WORD  0                ;;USER DEFINED
283  001210  000000          $TIMES:  0                       ;;MAX. NUMBER OF ITERATIONS
284  001212  000000          $ESCAPE:0                        ;;ESCAPE ON ERROR ADDRESS
285  001214  177607  000377  $BELL:   .ASCIZ  <207><377><377> ;;CODE FOR BELL
286  001220     077          $QUES:   .ASCII  /?/             ;;QUESTION MARK
287  001221     015          $CRLF:   .ASCII  <15>            ;;CARRIAGE RETURN
288  001222  000012          $LF:     .ASCIZ  <12>            ;;LINE FEED
```

```
289                                 ;********************************************************
290
291                                 .SBTTL   ERROR POINTER TABLE
292
293                                 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
294                                 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
295                                 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
296                                 ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
297                                 ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
298
299                                 ;*      EM              ;;POINTS TO THE ERROR MESSAGE
300                                 ;*      DH              ;;POINTS TO THE DATA HEADER
301                                 ;*      DT              ;;POINTS TO THE DATA
302                                 ;*      DF              ;;POINTS TO THE DATA FORMAT
303
304
305     001224                      $ERRTB:
306                                 ;********************************************************
307                                 ;********************************************************
308                                 ;ITEM 1
309     001224  011404                      EM1             ;CPU TRAPPED THRU LOC 4 -TIME OUT
310     001226  011452                      DH1             ; ADDR    $ERRPC #ERR/TST#
311     001230  015262                      DT1             ;$REG2,$ERRPC,$TSTNM,0
312     001232  000000                      0
313                                 ;ITEM 2
314     001234  011503                      EM2             ;CPU ISSUED A BUS GRANT WITH PSW = 7
315                                                         ;DEV 1 SHOULD NOT HAVE BECOME BUS MASTER
316     001236  011616                      DH2             ;BE1DB BE1CC BE1BA BE1CR1 PSW $ERRPC #ERR/TST#
317     001240  015272                      DT2             ;$REG0,$REG1,$REG2,$REG3,$REG4,$ERRPC,$TSTNM,0
318     001242  000000                      0
319                                 ;ITEM  3
320     001244  011706                      EM3             ;CPU DID NOT ISSUE A BUS NPG
321     001246  011740                      DH3             ;BE1CR1 BE1CC FM-PS TO-PS $ERRPC #ERR/TST#
322     001250  015312                      DT3             ;$REG0,$REG1,$REG2,$REG3,$ERRPC,$TSTNM,0
323     001252  000000                      0
324                                 ;ITEM 4
325     001254  012021                      EM4             ;CPU DID NOT ISSUE BUS GRANT 7
326     001256  011740                      DH3
327     001260  015312                      DT3
328     001262  000000                      0
329                                 ;ITEM 5
330     001264  012057                      EM5             ;CPU DID NOT ISSUE BUS GRANT 6
331     001266  011740                      DH3
332     001270  015312                      DT3
333     001272  000000                      0
334                                 ;ITEM 6
335     001274  012115                      EM6             ;CPU DID NOT ISSUE BUS GRANT 5
336     001276  011740                      DH3
337     001300  015312                      DT3
338     001302  000000                      0
339                                 ;ITEM 7
340     001304  012153                      EM7             ;CPU DID NOT ISSUE BUS GRANT 4
341     001306  011740                      DH3
342     001310  015312                      DT3
343     001312  000000                      0
344                                 ;ITEM 10
```

```
    345  C01314  012211                    EM10          ;ONE OR MORE DEVICES DID NOT INTERRUPT
    346  001316  012257                    DH10          ;THIS IS THE ORDER IN WHICH THEY INTERRUPTED
    347                                                   ;  1ST  2ND  3RD  4TH $ERRPC #ERR/TST#
    348  001320  015330                    DT10          ;$REG1,$REG2,$REG3,$REG4,$ERRPC,$TSTNM,0
    349  001322  000000                    0
    350                              ;ITEM 11
    351  001324  012415                    EM11          ;BUS ADDRESS LINES <A17:A14> DID NOT FUNCTION PROPERLY
    352  001326  012503                    DH11          ;BE1CR1 BE1CR2 BE1BA $ERRPC #ERR/TST#
    353  001330  015346                    DT11          ;$REG1,$REG2,$REG3,$ERRPC,$TSTNM,0
    354  001332  000000                    0
    355                              ;ITEM 12
    356  001334  012554                    EM12          ;CPU NO SACK TIME OUT LOGIC FAILED(TO NEGATE BUS GRANT)
    357  001336  012642                    DH12          ;BE1CR1 BE1CR2 $ERRPC #ERR/TST#
    358  001340  015362                    DT12          ;$REG0,$REG1,$ERRPC,$TSTNM,0
    359  001342  000000                    0
    360                              ;ITEM 13
    361  001344  012703                    EM13          ;CPU DID NOT PROPERLY EXECUTE AN ACLO/DCLO SEQUENCE
    362  001346  012766                    DH13          ;$ERRPC #ERR/TST#
    363  001350  015374                    DT13          ;$ERRPC,$TSTNM,0
    364  001352  000000                    0
    365                              ;ITEM 14
    366  001354  013007                    EM14          ;CPU DID NOT TRAP FROM BUS PARITY ERR PA/PB = 0/1
    367  001356  012766                    DH13
    368  001360  015374                    DT13
    369  001362  000000                    0
    370                              ;ITEM 15
    371  001364  013072                    EM15          ;DEV 1 DID DATIP WITH ROL ON DATOB TO MEMORY
    372                                                   ;THE TRANSFER TO THE FOLLOWING LOC WAS INCORRECT
    373  001366  013235                    DH15          ;MEMORY   ACTUAL  CORRECT
    374                                                   ; LOC     DATA    DATA  $ERRPC #ERR/TST# $ICNT #
    375  0U1370  015402                    DT15          ;$REG0,$REG1,$REG3,$ERRPC,$TSTNM,$ICNT,0
    376  001372  000000                    0
    377                              ;ITEM 16
    378  001374  013347                    EM16          ;DEV 3'S DATO TO MEMORY DID NOT EQUAL PATTERN IN R3
    379  001376  013235                    DH15
    380  001400  015402                    DT15
    381  001402  000000                    0
    382                              ;ITEM 17
    383  001404  013435                    EM17          ;DEV 4'S DATO TO MEMORY DID NOT EQUAL PATERN IN R4
    384  001406  013235                    DH15
    385  001410  015402                    DT15
    386  001412  000000                    0
    387                              ;ITEM 20
    388  001414  013523                    EM20          ;DEV 1 DID FUN 1-NPR-DATIP;INCORRECT PATTERN IN MEMORY
    389  001416  013235                    DH15
    390  001420  015402                    DT15
    391  001422  000000                    0
    392                              ;ITEM 21
    393  001424  013617                    EM21          ;DEV 2 DID FUN 2-NPR-DATOB;INCORRECT PATTERN IN MEMORY
    394  001426  013235                    DH15
    395  001430  015402                    DT15
    396  001432  000000                    0
    397                              ;ITEM 22
    398  001434  013713                    EM22          ;BIT 7 OF CR2 SET-CPU DID NOT TIME OUT WITH SACK INHIBITED
    399  001436  014005                    DH22          ;DEV #  PC  $ERRPC #ERR/TST#
    400  001440  015420                    DT22          ;$TMP4,$REG5,$ERRPC,$TSTNM,0
```

```
   401   001442  000000                         0
   402                                  ;ITEM 23
   403   001444  014047                         EM23              ;BIT 11 OF CR2 SET-NO SSYN ON INTR SIGNAL
   404   001446  014005                         DH22
   405   001450  015420                         DT22
   406   001452  000000                         0
   407                                  ;ITEM 24
   408   001454  014120                         EM24              ;BIT 5 OF CR2 SET-RECEIVED WRONG GRANT
   409   001456  014005                         DH22
   410   001460  015420                         DTc2
   411   001462  000000                         0
   412                                  ;ITEM 25
   413   001464  014166                         EM25              ;BIT 6 OF CR2 SET-BUS LATE
   414   001466  014005                         DH22
   415   001470  015420                         DT22
   416   001472  000000                         0
   417                                  ;ITEM 26
   418   001474  014220                         EM26              ;BIT 8 OF CR2 SET-DEV DID NOT RECEIVE SSYN
   419   001476  014005                         DH22
   420   001500  015420                         DT22
   421   001502  000000                         0
   422                                  ;ITEM 27
   423   001504  014262                         EM27              ;BIT 9 OF CR2 SET-WRONG ADDR ON BUS
   424   001506  014005                         DH22
   425   001510  015420                         DT22
   426   001512  000000                         0
   427                                  ;ITEM 30
   428   001514  014331                         EM30              ;BIT 10 OF CR2 SET-DEV RECEIVED OTHER THAN ONE GRANT
   429   001516  014005                         DH22
   430   001520  015420                         DT22
   431   001522  000000                         0
   432                                  ;ITEM31
   433   001524  014420                         EM31              ;BKGRND RTN INSTRUCTIONS OF NEGB'S WERE NOT DONE
   434                                                            ;CORRECTLY TO $REG1 DURING MULTITRANFERS II
   435   001526  014560                         DH31              ;ACTUAL CORRECT
   436                                                            ;DATA    DATA $ERRPC #ERR/TST# $ICNT #
   437   001530  015432                         DT31              ;$REG1,146463,$ERRPC,$TSTNM,$ICNT,^
   438   001532  000000                         0
   439                                  ;ITEM 32
   440   001534  014653                         EM32              ;DEV 3 DID DATI BUT HAS INCORRECT
   441                                                            ;VALUES IN DATA REGISTER
   442   001536  014560                         DH31
   443   001540  015432                         DT31
   444   001542  000000                         0
   445                                  ;ITEM 33
   446   001544  014737                         EM33              ;DEV 4 DID NOT INTR THE CORRECT # OF TIMES
   447   001546  014560                         DH31
   448   001550  015432                         DT31
   449   001552  000000                         0
   450                                  ;ITEM 34
   451   001554  015011                         EM34              ;LAST DATI XFER BY DEV 1 WAS INCORRECT-
   452                                                            ;EITHER DEV DID NOT WORK OR WRONG DATA WASSET UP
   453   001556  014560                         DH31
   454   001560  015432                         DT31
   455   001562  000000                         0
   456                                  ;ITEM 35
```

```
457  001564  015165                      EM35               ;CPU TRAPPED THRU LOC 0 TO CATCH
458                                                          ;IMPROPERLY LOADED VECTORS
459  001566  011452                      DH1            ; ADDR  $ERRPC #ERR/TST#
460  001570  015262                      DT1            ;$REG2,$ERRPC,$TSTNM,0
461  001572  000000                      0
462                              ;************************************************************
463                              ;************************************************************
464                              ;************************************************************
465  001574  007740                      ALLERR :7740         ;ALL ERR BITS OF CR2
466  001576  170014                      SIMLGO :170014       ;ADDR TO SET OFF ALL DEVICES SIMOLTANEOUSLY
467          000114                      PBVEC  =114          ;TRAP VEC FOR PARITY ERROR
468          000116                      PBPSW  =116          ;PSW ADDR FOR TRAP ON PARITY ERR
469  001600  000000                      BE1DB  :0            ;DATA REG ADDR FOR DEVICE 1
470  001602  000000                      BE1CC  :0            ;CYCLE COUNT REG ADDR FOR DEV 1
471  001604  000000                      BE1BA  :0            ;ADDR REG ADDR FOR DEV 1
472  001606  000000                      BE1CR1 :0            ;CONTROL REG 1 ADDR FOR DEV 1
473  001610  000000                      BE1CLR :0            ;CLEAR ERRS REG ADDR FOR DEV 1
474  001612  000000                      BE1CR2 :0            ;CONTROL REG 2 ADDR FOR DEV 1
475  001614  000000                      BE2DB  :0            ;DATA REG ADDR FOR DEV 2
476  001616  000000                      BE2CC  :0            ;CYCLE COUNT REG ADDR FOR DEV 2
477  001620  000000                      BE2BA  :0            ;ADDR REG ADDR FOR DEV 2
478  001622  000000                      BE2CR1 :0            ;CONTROL REG 1 ADDR FOR DEV 2
479  001624  000000                      BE2CLR :0            ;CLEAR ERRS REG ADDR FOR DEV 2
480  001626  000000                      BE2CR2 :0            ;CONTROL REG 2 ADDR FOR DEV 2
481  001630  000000                      BE3DB  :0            ;DATA REG ADDR FOR DEV 3
482  001632  000000                      BE3CC  :0            ;CYCLE COUNT REG ADDR FOR DEV 3
483  001634  000000                      BE3BA  :0            ;ADDR REG ADDR FOR DEV 3
484  001636  000000                      BE3CR1 :0            ;CONTROL REG 1 ADDR FOR DEV 3
485  001640  000000                      BE3CLR :0            ;CLEAR ERRS REG ADDR FOR DEV 3
486  001642  000000                      BE3CR2 :0            ;CONTROL REG 2 ADDR FOR DEV 3
487  001644  000000                      BE4DB  :0            ;DATA REG ADDR FOR DEV 4
488  001646  000000                      BE4CC  :0            ;CYCLE COUNT REG ADDR FOR DEV 4
489  001650  000000                      BE4BA  :0            ;ADDR REG ADDR FOR DEV 4
490  001652  000000                      BE4CR1 :0            ;CONTROL REG 1 ADDR FOR DEV 4
491  001654  000000                      BE4CLR :0            ;CLEAR ERRS REG ADDR FOR DEV 4
492  001656  000000                      BE4CR2 :0            ;CONTROL REG 2 ADDR FOR DEV 4
493  001660  000000                      BE1VEC :0            ;TRAP VEC ADDR FOR DEV 1
494  001662  000000                      BE1PSW :0            ;PSW ADDR FOR TRAP THRU BE1VEC
495  001664  000000                      BE2VEC :0            ;TRAP VEC ADDR FOR DEV 2
496  001666  000000                      BE2PSW :0            ;PSW ADDR FOR TRAP THRU BE2VEC
497  001670  000000                      BE3VEC :0            ;TRAP VEC ADDR FOR DEV 3
498  001672  000000                      BE3PSW :0            ;PSW ADDR FOR TRAP THRU BE3VEC
499  001674  000000                      BE4VEC :0            ;TRAP VEC ADDR FOR DEV 4
500  001676  000000                      BE4PSW :0            ;PSW ADDR FOR TRAP THRU BE4VEC
501  001700  000000                      DEVCNT :0            ;CONTAINS # OF DEVICES ON BUS
502  001702  000000  000000  000000      DEVS   :0,0,0,0      ;WILL CONTAIN ADDR(S) OF INTR'G DEVS
503  001710  000000
504  001712  000000                      DATA1  :0            ;MAX ADDR TO WHICH DATA XFERRED BY DEV 1
505  001714  000000                      DATA2  :0            ;MAX ADDR TO WHICH DATA XFERRED BY DEV 2
506  001716  000000                      DATA3  :0            ;MAX ADDR TO WHICH DATA XFERRED BY DEV 3
507  001720  000000                      DATA4  :0            ;MAX ADDR TO WHICH DATA XFERRED BY DEV 4
508  001722  000000                      ENDMEM :0            ;TAG ENDING DEFINED LABELS
509                              ;************************************************************
510                              ;************************************************************
511  001724                      CLRRTN:
512  001724  012703  001600          MOV      #BE1DB,R3       ;R3 IS POINTER TO BUFFER AREAS
```

```
 513   001730  005023              1$:      CLR     (R3)+           ;CLEAR BUFFER THEN INCREMENT ADDR
 514   001732  022703  001722               CMP     #ENDMEM, R3     ;IF POINTER AT LAST BUFFER, EXIT
 515   001736  100374               BPL     1$              ;IF PLUS, GO BACK AND CLEAR NEXT ADDR
 516   001740  012703  001160               MOV     #$REG0,R3       ;NOW START TO CLEAR TEMP REGISTERS
 517   001744  005023              2$:      CLR     (R3)+           ;CLEAR CURRENT ADDR
 518   001746  022703  001206               CMP     #$TMP5,R3       ;CHECK FOR LAST TEMP REG ADDR
 519   001752  101374               BHI     2$              ;IF NOT, CLEAR NEXT TEMP REG
 520   001754  000207               RTS     PC              ;EXIT
 521                                 ;************************************************************
 522                                 ;************************************************************
 523   001756                      START:
 524   001756  012706  001100               MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 525   001762  005026               CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 526   001764  022706  001126               CMP     #$BDDAT,R6      ;;DONE?
 527   001770  001374               BNE     .-6             ;;LOOP BACK IF NO
 528   001772  012706  001100               MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 529   001776  012737  015656  000020        MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 530   002004  012737  000340  000022        MOV     #340,@#IOTVEC+2 ;;LEVEL 7
 531   002012  012737  016134  000030        MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 532   002020  012737  000340  000032        MOV     #340,@#EMTVEC+2 ;;LEVEL 7
 533   002026  012737  020136  000034        MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 534   002034  012737  000340  000036        MOV     #340,@#TRAPVEC+2;LEVEL 7
 535   002042  012737  020202  000024        MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
 536   002050  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;LEVEL 7
 537   002056  013737  015502  015474        MOV     SENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
 538   002064  005037  001210               CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
 539   002070  005037  001212               CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 540   002074  112737  000001  001115        MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
 541   002102  012737  002102  001106        MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
 542   002110  012737  002110  001110        MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
 543   002116  013746  000004               MOV     @#4,-(SP)       ;;SAVE ERROR VECTOR
 544   002122  013746  000006               MOV     @#6,-(SP)
 545   002126  012737  002142  000004        MOV     #64$,4          ;;SET UP TIME OUT VECTOR
 546   002134  005777  176776               TST     @SWR            ;;TRY TO REFERENCE HARDWARE SWR
 547   002140  000407               BR      65$             ;;BRANCH IF NO TIMEOUT TRAP OCCURS
 548   002142  012737  000176  001136 64$:   MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
 549   002150  012737  000174  001140        MOV     #DISPREG,DISPLAY ;;POINT TO SOFTWARE DISPLAY REG
 550   002156  022626               CMP     (SP)+,(SP)+     ;;RESTORE STACK
 551   002160  012637  000006      65$:      MOV     (SP)+,@#6       ;;RESTORE ERROR VECTOR
 552   002164  012637  000004               MOV     (SP)+,@#4
 553   002170  032777  000200  176740        BIT     #BIT07,@SWR     ;IS SWITCH 7 UP?
 554   002176  001402               BEQ     3$              ;IF NOT, SKIP TYPEOUT
 555   002200  104400  011174               TYPE    ,@NO
 556   002204                      3$:
 557   002204  022737  000777  001174        CMP     #777,$TMP0      ;IS THIS RESTART FROM LOC 220?
 558   002212  001002               BNE     5$              ;IF NOT,SKIP THE JMP INSTR
 559   002214  000137  003506               JMP     @#TST1          ;ELSE JUMP TO TEST 1
 560
 561   002220                      5$:
 562   002220  012737  010562  000000        MOV     #THRU0,0        ;SET UP FOR TRAP THRU LOC 0
 563   002226  012737  000340  000002        MOV     #PR7,2          ;SET UP PSW FOR TRAP THRU 0
 564   002234  032777  000200  176674        BIT     #BIT07,@SWR     ;IS SWITCH 7 UP?
 565   002242  001452               BEQ     33$             ;IF NOT, SKIP TYPEOUT
 566   002244  104400  002252               TYPE    ,.+4            ;;TYPE ASCIZ STRING
 567   002250  000422               BR      66$             ;;GET OVER THE ASCIZ
 568                                 ;;.ASCIZ      <15><12>/IF BUS HANGS WHILE SIZING MEMORY/
```

```
 569   002316                              66$:
 570   002316  104400  002324                        TYPE     ..+4              ::TYPE ASCIZ STRING
 571   002322  000422                                BR       67$               ::GET OVER THE ASCIZ
 572                                                  ::.ASCIZ      <15><12>/IT IS DUE TO NO CPU SSYN TIME OUT/
 573   002370                              67$:
 574   002370                              33$:
 575   002370  004737  001724                        JSR      PC,CLRRTN         ;CLEAR BUFFER AREAS
 576   002374  004737  016616                        JSR      PC,$SIZE          ;FIND AVAILABLE MEMORY
 577   002400  012737  000340  000006                MOV      #PR7,ERRVEC+2     ;PS=7 FOR TRAP THRU LOC 4
 578   002406  012700  170000                         MOV      #170000,R0        ;SET UP POINTER FOR 1ST POSSIBLE DEV ADDR
 579   002412  012702  000510                         MOV      #510,R2           ;SET UP POINTER FOR 1ST POSSIBLE VEC ADDR
 580   002416  012701  001600                         MOV      #BE1DB,R1         ;SET UP POINTER FOR DEVICE ADDR LOCATION
 581   002422  012703  001660                         MOV      #BE1VEC,R3        ;SET UP POINTER FOR INTR ADDR LOCATION
 582   002426                              LODDEV:
 583   002426  022700  170060                        CMP      #170060,R0        ;IS R0 > LAST POSSIBLE DEV ADDR?
 584   002432  002002                                BGE      10$               ;IF NOT,GO TO 10$
 585   002434  000137  002736                        JMP      BGIN              ;ELSE GO TO BGIN
 586   002440                              10$:
 587   002440  012737  002544  000004                MOV      #NODEV,ERRVEC     ;SET UP TRAP VECTOR FOR TIME OUT
 588   002446  005710                                TST      (R0)              ;SEE IF ACTUAL DEVICE ADDRESS EXISTS
 589   002450  012737  002662  000004                MOV      #TYMOUT,ERRVEC    ;CHANGE TRAP VECTOR FOR ERROR CONDITION
 590   002456  005237  001700                        INC      DEVCNT            ;COUNT DEVICES
 591   002462  010021                      MOVREG:    MOV      R0,(R1)+          ;MOVE ACTUAL DEVICE ADDR TO DEVICE NAME
 592   002464  010037  001172                        MOV      R0,$REG5          ;REG5 CONTAINS LAST DEVICE ADDR
 593   002470  062700  000002                        ADD      #2,R0             ;INCREMENT POINTER BY 2
 594   002474  105237  001174                        INCB     $TMP0             ;COUNT # OF REGISTERS PER DEVICE
 595   002500  122737  000005  001174                CMPB     #5,$TMP0          ;AFTER 5 REGISTERS
 596   002506  001365                                BNE      MOVREG            ;ARE RECORDED
 597   002510  105037  001174                        CLRB     $TMP0             ;CLEAR THE COUNTING REGISTER
 598   002514  062700  000004                        ADD      #4,R0             ;ADD 4 TO THE POINTER THEN
 599   002520  010021                                MOV      R0,(R1)+          ;RECORD THE LAST REGISTER ADDRESS
 600   002522  062700  000002                        ADD      #2,R0             ;INCREMENT POINTER BY 2
 601   002526                              MOVVEC:                              ;NOW START RECORDING
 602   002526  010223                                MOV      R2,(R3)+          ;THE INTR VECTORS
 603   002530  062702  000002                        ADD      #2,R2             ;INCREMENT POINTER BY 2
 604   002534  010223                                MOV      R2,(R3)+          ;THE INTR VECTORS
 605   002536  062702  000002                        ADD      #2,R2             ;INCREMENT POINTER BY 2
 606   002542  000731                                BR       LODDEV            ;AND GO SEE IF THER'S ANOTHER DEVICE
 607                                       ;***************************************************************
 608                                       ;***************************************************************
 609   002544                              NODEV:                               ;TIME OUT ROUTINE FOR DEVICE CHECK
 610   002544  022700  170060                        CMP      #170060,R0        ;IF ALL POSSIBLE ADDR'S HAVE NOT BEEN CHECKED
 611   002550  003035                                BGT      ADD20             ;OUT-GO BACK AND CHECK FOR MORE,
 612   002552  012716  002736                        MOV      #BGIN,(SP)        ;ELSE CHANGE STACK POINTER
 613   002556  022737  000000  001700                CMP      #0,DEVCNT         ;CHECK FOR NO EXERCISERS
 614   002564  001035                                BNE      EXNO              ;IF ONE OR MORE EXERCISERS, EXIT
 615   002566  104400  002574                        TYPE     ..+4              ::TYPE ASCIZ STRING
 616   002572  000423                                BR       64$               ::GET OVER THE ASCIZ
 617                                                  ::.ASCIZ      <15><12>/THERE ARE NO EXERCISERS ON THE BUS/
 618   002642                              64$:
 619   002642  000000                                HALT
 620   002644  062700  000020            ADD20:      ADD      #20,R0            ;ADD 20 TO POINTER
 621   002650  062702  000004                        ADD      #4,R2             ;POINTER=NEXT DEV'S VEC LOCATIONS
 622   002654  012716  002426                        MOV      #LODDEV,(SP)      ;GO BACK TO LODDEV
 623   002660                              EXNO:
 624   002660  000002                                RTI                        ;EXIT
```

```
 625                                 ;*******************************************************************
 626                                 ;*******************************************************************
 627   002662                        TYMOUT:                          ;TIME OUT ROUTINE
 628
 629   002662  011637  001164                MOV     (SP),$REG2       ;THE MOVE IS FOR TYPEOUT REASONS
 630   002666  162737  000002  001164        SUB     #2,$REG2         ;SUBTRACT 2 TO FIND ACTUAL ADDR
 631   002674  104001                        ERROR   1                ;ERR MESSG FOR ILLEGAL TIME OUT
 632   002676  000002                        RTI
 633                                 ;*******************************************************************
 634                                 ;*******************************************************************
 635                                 ;*******************************************************************
 636   002700                        CLRREG:
 637   002700  005000                        CLR     R0               ;R0=0
 638   002702  005002                        CLR     R2               ;R2=0
 639   002704  012701  001600                MOV     #BE1DB,R1        ;R1 WILL POINT TO ADDR OF 1ST DEVICE
 640   002710  005200                1$:     INC     R0               ;R0 IS REGISTER COUNTER
 641   002712  005031                        CLR     @(R1)+           ;CLEAR CONTENTS OF REG
 642   002714  022700  000006                CMP     #6,R0            ;IF COUNT IS NOT 6
 643   002720  001373                        BNE     1$               ;GO BACK AND INCREMENT COUNT
 644   002722  005000                        CLR     R0               ;ELSE CLEAR R0
 645   002724  005202                        INC     R2               ;ADD 1 TO R2(DEVICE COUNTER)
 646   002726  020237  001700                CMP     R2,DEVCNT        ;SEE IF IT = PREVIOUS COUNT
 647   002732  101766                        BLOS    1$               ;IF NOT, CLEAR NEXT DEV REGS
 648   002734  000207                        RTS     PC               ;EXIT
 649                                 ;///////////////////////////////////////////////////////////////////
 650                                 ;///////////////////////////////////////////////////////////////////
 651                                 ;///////////////////////////////////////////////////////////////////
 652   002736                        BGIN:
 653   002736  012737  010342  000024        MOV     #PWRFAL,PWRVEC   ;TAKE CARE OF BIT 4(S) BEING SET RANDOMLY IN CR2(S)
 654   002744  004737  010240                JSR     PC,STVEC         ;SET UP VEC(S) FOR RANDOM ERRS
 655   002750  004737  002700                JSR     PC,CLRREG        ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
 656   002754  005037  001174                CLR     $TMPO            ;CLEAR TEMPORARY REG
 657   002760  005037  001160                CLR     $REG0            ;CLEAR COUNTER
 658   002764  013737  017152  001174        MOV     $LSTBK,$TMPO     ;********************
 659   002772  032777  000200  176136        BIT     #BIT07,@SWR      ;IS SWITCH 7 UP?
 660   003000  001002                        BNE     2$               ;IF UP, GO TO 2$
 661   003002  000137  003506                JMP     5$               ;ELSE SKIP THE TYPEOUTS
 662   003006  005237  001160        2$:     INC     $REG0            ;ADD 1 TO REG0
 663   003012  006237  001174                ASR     $TMPO            ;SHIFT TO DIVIDE
 664   003016  022737  000005  001160        CMP     #5,$REG0         ;IS TMPO DIVIDED BY 40(5 ASR'S)
 665   003024  001370                        BNE     2$               ;IF NOT, SHIFT AGAIN
 666   003026  005237  001174                INC     $TMPO            ;ADD 1 FOR FUDGE FACTOR
 667   003032  104400  001221                TYPE    ,$CRLF
 668   003036  013746  001174                MOV     $TMPO,-(SP)      ;;SAVE $TMPO FOR TYPEOUT
 669                                                                  ;;THIS IS THE # OF K OF MEMORY
 670   003042  104404                        TYPDS                    ;;GO TYPE--DECIMAL ASCII WITH SIGN
 671   003044  104400  003052                TYPE    .,+4             ;;TYPE ASCIZ STRING
 672   003050  000424                        BR      64$              ;;GET OVER THE ASCIZ
 673                                 ;;.ASCIZ      /K OF MEMORY IS AVAILABLE IN THIS SYSTEM/
 674   003122                        64$:
 675   003122  104400  003130                TYPE    .,+4             ;;TYPE ASCIZ STRING
 676   003126  000431                        BR      65$              ;;GET OVER THE ASCIZ
 677                                 ;;.ASCIZ      <15><12>/THE FOLLOWING # OF EXERCISERS ARE ON THE BUS:  /
 678   003212                        65$:
 679   003212  013746  001700                MOV     DEVCNT,-(SP)     ;;SAVE DEVCNT FOR TYPEOUT
 680   003216  104402                        TYPOS                    ;;GO TYPE--OCTAL ASCII
```

```
681  003220    001                              .BYTE   1               ;;TYPE 1 DIGIT(S)
682  003221    000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS
683  003222  104400    003230                   TYPE    ,.+4            ;;TYPE ASCIZ STRING
684  003226  000436                             BR      66$             ;;GET OVER THE ASCIZ
685                                             ;;.ASCIZ    <15><12>/THE LOWEST ELECT. PRIORITY UBE SHOULD NOT HAVE W1 JUMPER/
686  003324                          66$:
687  003324  104400    003332                   TYPE    ,.+4            ;;TYPE ASCIZ STRING
688  003330  000415                             BR      67$             ;;GET OVER THE ASCIZ
689                                             ;;.ASCIZ    <15><12>/DEVICE ADDRESS(ES): /<15><12>
690  003364                          67$:
691  003364  005037    001174                   CLR     $TMPO           ;CLEAR TMPO(USED AS COUNTER)
692  003370  012700    001600                   MOV     #BE1DB,R0       ;USE R0 AS POINTER TO ADDRESSES
693  003374                          4$:
694  003374  005237    001174                   INC     $TMPO           ;ADD 1 TO TMPO
695  003400  011037    001160                   MOV     (R0),$REG0      ;MOVE FOR TYPEOUT REASONS
696  003404  104400    003412                   TYPE    ,.+4            ;;TYPE ASCIZ STRING
697  003410  000403                             BR      68$             ;;GET OVER THE ASCIZ
698                                             ;;.ASCIZ    / DEV/
699  003420                          68$:
700  003420  013746    001174                   MOV     $TMPO,-(SP)     ;;SAVE $TMPO FOR TYPEOUT
701  003424  104402                             TYPOS                   ;;GO TYPE--OCTAL ASCII
702  003426    002                              .BYTE   2               ;;TYPE 2 DIGIT(S)
703  003427    000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS
704  003430  104400    003436                   TYPE    ,.+4            ;;TYPE ASCIZ STRING
705  003434  000402                             BR      69$             ;;GET OVER THE ASCIZ
706                                             ;;.ASCIZ    / = /
707  003442                          69$:
708  003442  013746    001160                   MOV     $REG0,-(SP)     ;;SAVE $REG0 FOR TYPEOUT
709  003446  104402                             TYPOS                   ;;GO TYPE--OCTAL ASCII
710  003450   \ 006                             .BYTE   6               ;;TYPE 6 DIGIT(S)
711  003451    000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS
712  003452  062700    000014                   ADD     #14,R0          ;ADD 14 FOR NEXT ADDR
713  003456  023737    001174  001700           CMP     $TMPO,DEVCNT    ;SEE IF TMPO = # OF DEVICES
714  003464  001343                             BNE     4$              ;IF NOT, GO TYPE NEXT ADDR
715  003466  104400    001221                   TYPE    ,$CRLF          ;TYPE <CR><LF>
716  003472  022737    000004  001700           CMP     #4,DEVCNT       ;SEE IF THERE ARE 4 DEVICES
717  003500  001002                             BNE     5$              ;IF NOT,SKIP THE TYPE OUT
718  003502  104400    011277                   TYPE    ,FOR4           ;ELSE TYPE MSSG FOR 4TH DEV
719  003506                          5$:
720
721
722                                   ;*************************************************************************
723                                   ;*TEST 1        NO BUS GRANTS ISSUED WITH PROCESSOR AT HIGHER PRIORITY THAN BUS REQUEST
724                                   ;*THIS TEST IS TO INSURE THAT ANY REQUEST IS NOT
725                                   ;*HONORED AS LONG AS THE PROCESSOR IS AT THE SAME OR
726                                   ;*HIGHER PRIORITY
727                                   ;*************************************************************************
728  003506  000004                  TST1:    SCOPE
729  003510  004737    002700                   JSR     PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
730  003514                          NG:
731
732  003514  012777    004730  176136           MOV     #ERRCHK,@BE1VEC ;SET UP DEVICE 1 INTR VECTOR
733  003522  012777    000340  176132           MOV     #PR7,@BE1PSW    ;SET UP DEVICE 1 PSW VECTOR
734  003530  012737    002662  000004           MOV     #TYMOUT,ERRVEC  ;SET UP TRAP THRU LOC 4(TIME OUT VEC)
735  003536  012700    000340                   MOV     #PR7,R0         ;MOVE PS=7 TO R0
736  003542  012701    002021                   MOV     #2021,R1        ;MOVE FUN 1-DATI-BR7 TO R1
```

```
737   003546   004737   010636                    JSR    PC,NOG           ;DO  NOG
738   003552   012700   000300                    MOV    #PR6,R0          ;MOVE PS=6 TO R0
739   003556   012701   002011                    MOV    #2011,R1         ;MOVE FUN 1-DATI-BR6 TO R1
740   003562   004737   010636                    JSR    PC,NOG           ;DO NOG
741   003566   012700   000240                    MOV    #PR5,R0          ;MOVE PS=5 TO R0
742   003572   012701   002005                    MOV    #2005,R1         ;MOVE FUN 1-DATI-BR5 TO R1
743   003576   004737   010636                    JSR    PC,NOG           ;DO NOG
744   003602   012700   000200                    MOV    #PR4,R0          ;MOVE PS=4 TO R0
745   003606   012701   002003                    MOV    #2003,R1         ;MOVE FUN 1-DATI-BR4 TO R1
746   003612   004737   010636                    JSR    PC,NOG           ;DO NOG
747   003616   052777   004000   175762            BIS    #BIT11,@BE1CR1   ;SET BIT 11 TO DO FUN 3
748   003624   052777   000040   175754            BIS    #BIT05,@BE1CR1   ;SET OFF DEV AT NPR LEVEL
749   003632   000240                              NOP                     ;ALLOW TIME FOR XFER
750
751
752                                    ;****************************************************************
753                                    ;*TEST 2           ISSUING OF NON-PROCESSOR GRANTS AND ARBITRATION TESTS
754                                    ;*THIS TEST WILL REQUEST ON NPR THRU BR4 LEVELS
755                                    ;*WITH THE PROCESSOR STATUS INITIALLY AT LEVEL 7 AND MAKE
756                                    ;*SURE THE DEVICE EXERCISES AN NPG TO DO A FUN 1-DATI,
757                                    ;*THEN THE REQUESTS WILL BE REPEATED WHILE SEQUENTIALLY
758                                    ;*LOWERING THE PROCESSOR STATUS FROM 7 TO 0 TO ALLOW
759                                    ;*ARBITRATION OF ALL REQUESTS AND THE ISSUING OF NPG
760                                    ;****************************************************************
761   003634   000004           TST2:  SCOPE
762
763   003636                    NPRTST:
764   003636   012700   000340            MOV    #PR7,R0
765   003642                    2$:
766   003642   123737   001115   001103    CMPB   $ERMAX,SERFLG    ;MAX ERRS FOR THIS TEST OCCURRED?
767   003650   100451                       BMI    TST3             ;;BR IF YES TO NEXT TEST
768   003652   012737   000340   177776     MOV    #PR7,PSW                 ;INITIAL PS
769   003660   012777   004730   175772     MOV    #ERRCHK,@BE1VEC  ;SET UP VECTOR LOCATION
770   003666   012777   000340   175766     MOV    #PR7,@BE1PSW     ;SET UP DEVICE INTR PSW
771   003674   012777   020342   175702     MOV    #ATEND,@BE1BA    ;SET UP ADDR REG
772   003702   012777   177777   175672     MOV    #-1,@BE1CC       ;SET CYCLE COUNT = 1
773   003710   012777   002077   175670     MOV    #2077,@BE1CR1    ;LOAD #2077 FUNTIONS
774   003716   010037   177776              MOV    R0,PSW           ;LOWER PROC STATUS
775
776   003722   022777   177777   175652     CMP    #-1,@BE1CC       ;SEE IF DEVICE WENT OFF
777   003730   001014                       BNE    5$               ;IF IT DID,SKIP ERR TYPEOUT
778   003732   017737   175650   001160     MOV    @BE1CR1,$REG0    ;NEXT MOVES ARE FOR TYPEOUTS
779   003740   017737   175636   001162     MOV    @BE1CC,$REG1
780   003746   012737   000340   001164     MOV    #PR7,$REG2
781   003754   010037   001166              MOV    R0,$REG3
782   003762   104003                       ERROR  3                ;TYPE ERROR MESSG
783   003762                    5$:
784   003762   162700   000040              SUB    #40,R0           ;LOWER PS BY 1 LEVEL
785   003766   020027   000000              CMP    R0,#PR0  ;SEE IF R0 IS LESS THAN 0
786   003772   100323                       BPL    2$               ;IF PLUS ,GO BACK AND DO ANOTHER CYCLE
787
788                                    ;****************************************************************
789                                    ;*TEST 3           ISSUING OF BUS GRANT 7 AND ARBITRATION TESTS
790                                    ;*THIS TEST WILL ARBITRATE FOR A BG7,
791                                    ;*THE REQUESTS WILL BE ON LEVELS BR7 THRU BR4, DOING
792                                    ;*FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS
```

```
793                                          ;*LOWERED SEQUENTIALLY FROM 7 TO 0.
794                                          ;***********************************************************
795   003774   000004                  TST3:    SCOPE
796   003776                           BR7TST:
797   003776   012700   000300             MOV      #PR6,R0          ;2ND PS WILL = 6
798   004002                           2$:
799   004002   123737   001115  001103     CMPB     $ERMAX,$ERFLG    ;MAX ERRS FOR THIS TEST OCCURRED?
800   004010   100451                       BMI      TST4             ;;BR IF YES TO NEXT TEST
801   004012   012737   000340  177776      MOV      #PR7,PSW                  ;INITIAL PS
802   004020   012777   004730  175632      MOV      #ERRCHK,@BE1VEC  ;SET UP VECTOR LOCATION
803   004026   012777   000340  175626      MOV      #PR7,@BE1PSW     ;SET UP DEVICE INTR PSW
804   004034   012777   020342  175542      MOV      #ATEND,@BE1BA    ;SET UP ADDR REG
805   004042   012777   177777  175532      MOV      #-1,@BE1CC       ;SET CYCLE COUNT = 1
806   004050   012777   002037  175530      MOV      #2037,@BE1CR1    ;LOAD #2037 FUNTIONS
807   004056   010037   177776             MOV      R0,PSW           ;LOWER PROC STATUS
808
809   004062   022777   177777  175512      CMP      #-1,@BE1CC       ;SEE IF DEVICE WENT OFF
810   004070   001014                       BNE      5$               ;IF IT DID,SKIP ERR TYPEOUT
811   004072   017737   175510  001160      MOV      @BE1CR1,$REG0    ;NEXT MOVES ARE FOR TYPEOUTS
812   004100   017737   175476  001162      MOV      @BE1CC,$REG1
813   004106   012737   000340  001164      MOV      #PR7,$REG2
814   004114   010037   001166             MOV      R0,$REG3
815   004120   104004                       ERROR    4                ;TYPE ERROR MESSG
816   004122                           5$:
817   004122   162700   000040             SUB      #40,R0           ;LOWER PS BY 1 LEVEL
818   004126   020027   000000             CMP      R0,#PR0 ;SEE IF R0 IS LESS THAN 0
819   004132   100323                       BPL      2$               ;IF PLUS ,GO BACK AND DO ANOTHER CYCLE
820
821
822                                          ;***********************************************************
823                                          ;*TEST 4          ISSUING OF BUS GRANT 6 AND ARBITRATION TESTS
824                                          ;*THIS TEST WILL ARBITRATE FOR A BG6.
825                                          ;*THE REQUESTS WILL BE ON LEVELS BR6 THRU BR4, DOING
826                                          ;*FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS
827                                          ;*LOWERED SEQUENTIALLY FROM 6 TO 0.
828                                          ;***********************************************************
829   004134   000004                  TST4:    SCOPE
830   004136                           BR6TST:
831   004136   012700   000240             MOV      #PR5,R0          ;2ND PS WILL = 5
832   004142                           2$:
833   004142   123737   001115  001103     CMPB     $ERMAX,$ERFLG    ;MAX ERRS FOR THIS TEST OCCURRED?
834   004150   100451                       BMI      TST5             ;;BR IF YES TO NEXT TEST
835   004152   012737   000300  177776      MOV      #PR6,PSW                  ;INITIAL PS
836   004160   012777   004730  175472      MOV      #ERRCHK,@BE1VEC  ;SET UP VECTOR LOCATION
837   004166   012777   000340  175466      MOV      #PR7,@BE1PSW     ;SET UP DEVICE INTR PSW
838   004174   012777   020342  175402      MOV      #ATEND,@BE1BA    ;SET UP ADDR REG
839   004202   012777   177777  175372      MOV      #-1,@BE1CC       ;SET CYCLE COUNT = 1
840   004210   012777   002017  175370.     MOV      #2017,@BE1CR1    ;LOAD #2017 FUNTIONS
841   004216   010037   177776             MOV      R0,PSW           ;LOWER PROC STATUS
842
843   004222   022777   177777  175352      CMP      #-1,@BE1CC       ;SEE IF DEVICE WENT OFF
844   004230   001014                       BNE      5$               ;IF IT DID,SKIP ERR TYPEOUT
845   004232   017737   175350  001160      MOV      @BE1CR1,$REG0    ;NEXT MOVES ARE FOR TYPEOUTS
846   004240   017737   175336  001162      MOV      @BE1CC,$REG1
847   004246   012737   000300  001164      MOV      #PR6,$REG2
848   004254   010037   001166             MOV      R0,$REG3
```

L 2

UNIBUS EXERCISER   MACY11 30A(1052) 03-MAY-78 14:28 PAGE 19
CZKUAD.P11   26-APR-78 17:02   T4   ISSUING OF BUS GRANT 6 AND ARBITRATION TESTS   SEQ 0024

```
849  004260  104005                         ERROR   5                  ;TYPE ERROR MESSG
850  004262                          5$:
851  004262  162700  000040                 SUB     #40,R0             ;LOWER PS BY 1 LEVEL
852  004266  020027  000000                 CMP     R0,#PR0 ;SEE IF R0 IS LESS THAN 0
853  004272  100323                          BPL     2$                 ;IF PLUS ,GO BACK AND DO ANOTHER CYCLE
854
855
856                                  ;********************************************************************
857                                  ;*TEST 5          ISSUING OF BUS GRANT 5 AND ARBITRATION TESTS
858                                  ;*THIS TEST WILL ARBITRATE FOR A BG5,
859                                  ;*THE REQUESTS WILL BE ON LEVELS BR5 THRU BR4, DOING
860                                  ;*FUN 1-DATI TRANSFERS, AND THE PROCESSOR STATUS
861                                  ;*LOWERED SEQUENTIALLY FROM 5 TO 0.
862                                  ;********************************************************************
863  004274  000004              TST5:   SCOPE
864  004276                      BR5TST:
865  004276  012700  000200                 MOV     #PR4,R0            ;2ND PS WILL = 4
866  004302                          2$:
867  004302  123737  001115  001103          CMPB    $ERMAX,$ERFLG      ;MAX ERRS FOR THIS TEST OCCURRED?
868  004310  100451                          BMI     TST6               ;;BR IF YES TO NEXT TEST
869  004312  012737  000240  177776          MOV     #PR5,PSW                  ;INITIAL PS
870  004320  012777  004730  175332          MOV     #ERRCHK,@BE1VEC ;SET UP VECTOR LOCATION
871  004326  012777  000340  175326          MOV     #PR7,@BE1PSW      ;SET UP DEVICE INTR PSW
872  004334  012777  020342  175242          MOV     #ATEND,@BE1BA      ;SET UP ADDR REG
873  004342  012777  177777  175232          MOV     #-1,@BE1CC        ;SET CYCLE COUNT = 1
874  004350  012777  002007  175230          MOV     #2007,@BE1CR1      ;LOAD #2007 FUNTIONS
875  004356  010037  177776                 MOV     R0,PSW             ;LOWER PROC STATUS
876
877  004362  022777  177777  175212          CMP     #-1,@BE1CC        ;SEE IF DEVICE WENT OFF
878  004370  001014                          BNE     5$                 ;IF IT DID,SKIP ERR TYPEOUT
879  004372  017737  175210  001160          MOV     @BE1CR1,$REG0      ;NEXT MOVES ARE FOR TYPEOUTS
880  004400  017737  175176  001162          MOV     @BE1CC,$REG1
881  004406  012737  000240  001164          MOV     #PR5,$REG2
882  004414  010037  001166                 MOV     R0,$REG3
883  004420  104006                          ERROR   6                  ;TYPE ERROR MESSG
884  004422                          5$:
885  004422  162700  000040                 SUB     #40,R0             ;LOWER PS BY 1 LEVEL
886  004426  020027  000000                 CMP     R0,#PR0 ;SEE IF R0 IS LESS THAN 0
887  004432  100323                          BPL     2$                 ;IF PLUS ,GO BACK AND DO ANOTHER CYCLE
888
889
890                                  ;********************************************************************
891                                  ;*TEST 6          ISSUING OF BUS GRANT 4 AND ARBITRATION TESTS
892                                  ;*THIS TEST WILL ARBITRATE FOR A BG4,
893                                  ;*THE REQUESTS WILL BE ON LEVEL BR4, DOING
894                                  ;*FUNC 1-DATI TRANSFERS, AND THE PROCESSOR STATUS
895                                  ;*LOWERED SEQUENTIALLY FROM 4 TO 0.
896                                  ;********************************************************************
897  004434  000004              TST6:   SCOPE
898  004436                      BR4TST:
899  004436  012700  000140                 MOV     #PR3,R0            ;2ND PS WILL = 3
900  004442                          2$:
901  004442  123737  001115  001103          CMPB    $ERMAX,$ERFLG      ;MAX ERRS FOR THIS TEST OCCURRED?
902  004450  100451                          BMI     TST7               ;;BR IF YES TO NEXT TEST
903  004452  012737  000200  177776          MOV     #PR4,PSW                  ;INITIAL PS
904  004460  012777  004730  175172          MOV     #ERRCHK,@BE1VEC ;SET UP VECTOR LOCATION
```

```
 905   004466  012777  000340  175166          MOV     #PR7,@BE1PSW      ;SET UP DEVICE INTR PSW
 906   004474  012777  020342  175102          MOV     #ATEND,@BE1BA     ;SET UP ADDR REG
 907   004502  012777  177777  175072          MOV     #-1,@BE1CC        ;SET CYCLE COUNT = 1
 908   004510  012777  002003  175070          MOV     #2003,@BE1CR1     ;LOAD #2003 FUNTIONS
 909   004516  010037  177776                  MOV     R0,PSW            ;LOWER PROC STATUS
 910
 911   004522  022777  177777  175052          CMP     #-1,@BE1CC        ;SEE IF DEVICE WENT OFF
 912   004530  001014                          BNE     5$                ;IF IT DID,SKIP ERR TYPEOUT
 913   004532  017737  175050  001160          MOV     @BE1CR1,$REG0     ;NEXT MOVES ARE FOR TYPEOUTS
 914   004540  017737  175036  001162          MOV     @BE1CC,$REG1
 915   004546  012737  000200  001164          MOV     #PR4,$REG2
 916   004554  010037  001166                  MOV     R0,$REG3
 917   004560  104007                          ERROR   7                 ;TYPE ERROR MESSG
 918   004562                          5$:
 919   004562  162700  000040                  SUB     #40,R0            ;LOWER PS BY 1 LEVEL
 920   004566  020027  000000                  CMP     R0,#PR0 ;SEE IF R0 IS LESS THAN 0
 921   004572  100323                          BPL     2$                ;IF PLUS ,GO BACK AND DO ANOTHER CYCLE
 922                              ;****************************************************************
 923
 924                              ;****************************************************************
 925                              ;*TEST 7         CPU TEST FOR NO SACK TIME OUT
 926                                      ;*THIS TEST WILL CHECK THAT THE CPU TIMES OUT AND
 927                                      ;*DROPS A GRANT IF NO SACK SIGNAL IS RECEIVED
 928                                      ;*IF THE CPU TIME OUT IS INOPERATIVE, THE BUS EXERCISER
 929                                      ;*WILL TIME OUT AND SEND THE SACK SIGNAL TO PREVENT
 930                                      ;*A BUS HANG AND SET AN ERROR FLAG IN CR2
 931                              ;****************************************************************
 932   004574  000004           TST7:   SCOPE
 933   004576  004737  002700            JSR     PC,CLRREG         ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
 934   004602  012777  177777  174772    MOV     #-1,@BE1CC        ;SET CYCLE COUNT = 1
 935   004610  012777  020342  174766    MOV     #ATEND,@BE1BA     ;SET UP DEVICE REG ADDR
 936   004616  012737  000340  177776    MOV     #PR7,PSW          ;SET PS=7
 937   004624  012737  002662  000004    MOV     #TYMOUT,ERRVEC    ;SET UP TIME OUT VECTOR
 938   004632  012777  004730  175020    MOV     #ERRCHK,@BE1VEC   ;SET UP DEVICE INTR VECTOR
 939   004640  012777  000340  175014    MOV     #PR7,@BE1PSW      ;SET UP DEVICE INTR PSW
 940   004646  052777  000010  174736    BIS     #BIT03,@BE1CR2    ;INHIBIT SACK RETURN
 941   004654  012777  006003  174724    MOV     #6003,@BE1CR1     ;DO FUN 3--BR4
 942   004662  012737  000140  177776    MOV     #PR3,PSW          ;LOWER PROC. STATUS TO 3
 943   004670  004737  011046            JSR     PC,CNTR           ;DELAY FOR TIMEOUT
 944   004674  042777  000010  174710    BIC     #BIT03,@BE1CR2    ;ALLOW FUTURE SACKS
 945   004702  105777  174704            TSTB    @BE1CR2           ;CHECK IF NO-NO SACK BIT IS SET
 946   004706  100024                    BPL     TST10             ;;IF NOT SET, GO TO NEXT TEST
 947   004710  017737  174672  001160    MOV     @BE1CR1,$REG0     ;MOVE FOR TYPEOUT REASONS
 948   004716  017737  174670  001162    MOV     @BE1CR2,$REG1     ;MOVE FOR TYPEOUT
 949   004724  104012                    ERROR   12                ;ERROR IF SET-DEVICE FORCED TO SEND SACK
 950   004726  000414                    BR      TST10             ;;GO TO NEXT TEST
 951                              ;****************************************************************
 952                              ;****************************************************************
 953   004730                    ERRCHK:
 954   004730  033777  001574  174654    BIT     ALLERR,@BE1CR2  ;CHECK FOR ANY ERRS IN CR2
 955   004736  001407                    BEQ     5$                ;IF NONE, EXIT
 956   004740  011637  001172            MOV     (SP),$REG5        ;FOR TYPEOUT OF PC
 957   004744  012737  000001  001204    MOV     #1,$TMP4          ;INDICATOR FOR DEVICE 1
 958   004752  004737  010406            JSR     PC,ERRTN          ;CHECK TO SEE IF ANY ERRORS OCCURED
 959   004756                    5$:
 960   004756  000002                    RTI                       ;EXIT TRAP
```

```
 961
 962
 963                                   ;***********************************************************
 964                                   ;*TEST 10        CPU TEST FOR RECEIVING SACK
 965                                         ;*THIS TEST IS TO INSURE THAT THE CPU CAN RECEIVE THE
 966                                         ;*SACK SIGNAL; THE TIME DELAY WILL BE SET ON DEVICE 1
 967                                         ;*AND SEVERAL DATI TRANSFERS MADE, IF THERE IS NO BUS
 968                                         ;*LATE ERROR, THE CPU RECEIVED SACK CORRECTLY
 969                                         ;*IT IS ASSUMED THAT DEV 1 TIME DELAY IS SET FOR 10 US
 970                                   ;***********************************************************
 971   004760  000004                 TST10:  SCOPE
 972
 973   004762  012737  000340  177776          MOV    #PR7,PSW        ;PS = 7
 974   004770  004737  002700                  JSR    PC,CLRREG       ;CLEAR ALL DEVICE REGISTERS
 975   004774  012702  020342                  MOV    #ATEND,R2       ;R2 WILL POINT TO END OF PROG
 976   005000  012705  000010                  MOV    #10,R5          ;R5 = # OF TEST WORDS TO CREATE
 977   005004  004737  011030                  JSR    PC,DOUP         ;CREATE THOSE TEST WORDS
 978
 979   005010  012777  004730  174642          MOV    #ERRCHK,@BE1VEC ;SET UP VECTOR LOCATION
 980   005016  012777  000340  174636          MOV    #PR7,@BE1PSW    ;SET UP DEVICE INTR PSW
 981   005024  012777  177770  174550          MOV    #-10,@BE1CC     ;SET UP CYCLE COUNT
 982   005032  012777  020342  174544          MOV    #ATEND,@BE1BA   ;SET UP ADDR REGISTER
 983   005040  052777  040000  174544          BIS    #BIT14,@BE1CR2  ;SET BIT 14 OF CR2 FOR TIME DELAY
 984   005046  012777  024441  174532          MOV    #24441,@BE1CR1  ;DO FUN 2-DATIP/NO ROL-NPR
 985   005054  012737  000000  177776          MOV    #PR0,PSW        ;LOWER PS TO ALLOW INTERRUPTS
 986   005062                         5$:
 987   005062  105777  17+520                  TSTB   @BE1CR1         ;SEE IF DONE BIT SET
 988   005066  100375                          BPL    5$              ;IF NOT, GO BACK AND WAIT
 989   005070  042777  040000  174514          BIC    #BIT14,@BE1CR2  ;ELSE CLEAR BIT 14 OF CR2
 990   005076  022777  000010  174474          CMP    #10,@BE1DB      ;DID LAST XFER MOVE 10 INTO DB
 991   005104  001407                          BEQ    10$             ;IF IT DID,GO TO 10$
 992   005106  017737  174466  001162          MOV    @BE1DB,$REG1    ;ELSE MOVE FOR ERR TYPE OUT
 993   005114  012737  000010  001164          MOV    #10,$REG2
 994   005122  104034                          ERROR  34              ;TYPE ERR MSSG
 995   005124                         10$:
 996   005124  032777  004000  174460          BIT    #BIT11,@BE1CR2  ;SEE IF NO SSYN ON INTR ERR SET
 997   005132  001402                          BEQ    TST11           ;;IF NOT SET, GO TO NEXT TEST
 998   005134  104023                          ERROR  23              ;ELSE TYPE ERR MSSG
 999   005136  000400                          BR     TST11           ;;THEN GO TO NEXT TEST
1000
1001
1002                                   ;***********************************************************
1003                                   ;*TEST 11        PASSING OF GRANTS AND INTERRUPT TEST
1004                                         ;*THIS TEST WILL SET OFF ALL AVAILABLE DEVICES SIMULTANEOUSLY
1005                                         ;*WHOSE ONLY FUNCTIONS WILL BE TO INTERRUPT, THE REQUESTS
1006                                         ;*WILL ALL BE AT LEVEL 7 SO THAT THE DEVICE CLOSEST TO THE CPU
1007                                         ;*SHOULD RECEIVE BG7 FIRST AND INTERRUPT FIRST, THE NEXT
1008                                         ;*CLOSEST SHOULD INTERRUPT NEXT AND SO ON.
1009                                   ;***********************************************************
1010   005140  000004                 TST11:  SCOPE
1011   005142  012737  000340  177776          MOV    #PR7,PSW        ;PS=7
1012   005150  004737  002700                  JSR    PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
1013   005154                         LOAD1:
1014   005154  012704  001702                  MOV    #DEVS,R4        ;DEVS CONTAINS SEQUENCE OF INTR'G DEVICE ADDRS
1015   005160  012777  005402  174472          MOV    #INTR1,@BE1VEC  ;SET UP DEVICE 1 INTR VECTOR
1016   005166  012777  000340  174466          MOV    #PR7,@BE1PSW    ;SET UP INTR PSW
```

B 3

UNIBUS EXERCISER   MACY11 30A(1052) 03-MAY-78 14:28 PAGE 22
CZKUAD.P11   26-APR-78 17:02   T11   PASSING OF GRANTS AND INTERRUPT TEST   SEQ 0027

```
1017  005174  012777  000036  174404              MOV     #36,@BE1CR1      ;DO FUN 0 - BR7 THRU BR4
1018  005202  122737  000001  001700              CMPB    #1,DEVCNT        ;IF ONLY 1 DEVICE ON BUS
1019  005210  001443                              BEQ     GO               ;BRANCH TO GO
1020  005212  012777  005420  174444   LOAD2:     MOV     #INTR2,@BE2VEC   ;SET UP DEVICE 2 INTR VECTOR
1021  005220  012777  000340  174440              MOV     #PR7,@BE2PSW     ;SET UP DEVICE 2 PSW VECTOR
1022  005226  012777  000036  174366              MOV     #36,@BE2CR1      ;DO FUN 0 - BR7 THRU BR4
1023  005234  122737  000002  001700              CMPB    #2,DEVCNT        ;IF ONLY 2 DEVICES ON BUS
1024  005242  001426                              BEQ     GO               ;BRANCH TO GO
1025  005244  012777  005436  174416   LOAD3:     MOV     #INTR3,@BE3VEC   ;SET UP DEVICE 3 INTR VECTOR
1026  005252  012777  000340  174412              MOV     #PR7,@BE3PSW     ;SET UP DEVICE 3 PSW VECTOR
1027  005260  012777  000036  174350              MOV     #36,@BE3CR1      ;DO FUN 0 - BR7 THRU BR4
1028  005266  122737  000003  001700              CMPB    #3,DEVCNT        ;IF ONLY 3 DEVICES ON BUS
1029  005274  001411                              BEQ     GO               ;BRANCH TO GO
1030  005276                           LOAD4:
1031  005276  012777  005454  174370              MOV     #INTR4,@BE4VEC   ;SET UP DEVICE 4 INTR VECTOR
1032  005304  012777  000340  174364              MOV     #PR7,@BE4PSW     ;SET UP DEVICE 4 PSW VECTOR
1033  005312  012777  000036  174332              MOV     #36,@BE4CR1      ;DO FUN 0 - BR7 THRU BR4
1034  005320                           GO:
1035  005320  005001                              CLR     R1               ;CLEAR R1 FOR COUNTING
1036  005322  005277  174250                      INC     @SIMLGO          ;SET SIMULTANEOUS GO REGISTER
1037  005326  012737  000000  177776              MOV     #PRO,PSW         ;LOWER PS TO ALLOW INTERRUPTS
1038  005334  004737  011046                      JSR     PC,CNTR          ;ALLOW TIME FOR INTERRUPTS BY COUNTING
1039  005340  020137  001700    CMPARE:           CMP     R1,DEVCNT        ;COMPARE THE TWO
1040  005344  001456                              BEQ     TST12            ;;IF BUFFERS INCREMENTED IN CORRECT SEQUENCE, GO TO NEXT
1041  005346  013737  001702  001162              MOV     DEVS,$REG1       ;MOVE FOR TYPEOUT REASONS
1042  005354  013737  001704  001164              MOV     DEVS+2,$REG2     ;MOVE FOR TYPEOUT REASONS
1043  005362  013737  001706  001166              MOV     DEVS+4,$REG3     ;MOVE FOR TYPEOUT REASONS
1044  005370  013737  001710  001170              MOV     DEVS+6,$REG4     ;MOVE FOR TYPEOUT REASONS
1045  005376  104010                              ERROR   10               ;TYPE ERR MSSG
1046  005400  000440                              BR      TST12            ;;GO TO NEXT TEST
1047                                   ;************************************************************************
1048                                   ;************************************************************************
1049  005402                           INTR1:
1050  005402  005201                              INC     R1               ;ADD 1 TO COUNTER ON INTR
1051  005404  013724  001600                      MOV     BE1DB,(R4)+      ;MOVE ADDR FOR TYPEOUT
1052  005410  012737  000001  001204              MOV     #1,$TMP4         ;INDICATOR FOR DEVICE 1
1053  005416  000424                              BR      INTRTN           ;BRANCH TO REST OF INTR RTN
1054  005420                           INTR2:
1055  005420  005201                              INC     R1               ;ADD 1 TO COUNTER ON INTR
1056  005422  013724  001614                      MOV     BE2DB,(R4)+      ;MOVE ADDR FOR TYPEOUT
1057  005426  012737  000002  001204              MOV     #2,$TMP4         ;INDICATOR FOR DEVICE 2
1058  005434  000415                              BR      INTRTN           ;BRANCH TO REST OF INTR RTN
1059  005436                           INTR3:
1060  005436  005201                              INC     R1               ;ADD 1 TO COUNTER ON INTR
1061  005440  013724  001630                      MOV     BE3DB,(R4)+      ;MOVE ADDR FOR TYPEOUT
1062  005444  012737  000003  001204              MOV     #3,$TMP4         ;INDICATOR FOR DEVICE 3
1063  005452  000406                              BR      INTRTN           ;BRANCH TO REST OF INTR RTN
1064  005454                           INTR4:
1065  005454  005201                              INC     R1               ;ADD 1 TO COUNTER ON INTR
1066  005456  013724  001644                      MOV     BE4DB,(R4)+      ;MOVE ADDR FOR TYPEOUT
1067  005462  012737  000004  001204              MOV     #4,$TMP4         ;INDICATOR FOR DEVICE 4
1068  005470                           INTRTN:
1069  005470  011637  001172                      MOV     (SP),$REG5       ;FOR TYPEOUT OF PC
1070  005474  004737  010406                      JSR     PC,ERRTN         ;SEE IF ERROR CAUSED INTR
1071  005500  000002                              RTI                      ;EXIT
1072
```

```
1073
1074                                     ;********************************************************
1075                                     ;*TEST 12          ADDRESS LINES (14 - 17) CHECK
1076                                             ;*THIS TEST WILL CHECK BUS ADDRESS LINES 14 THRU 17
1077                                             ;*BY DOING A FUN 1-DATI-NPR TO THOSE ADDRESSES
1078                                             ;*IF THE ADDRESSES DON'T EXIST THE INTERRUPT ROUTINE
1079                                             ;*WILL IGNORE ANY NO SSYN ERROR.
1080                                     ;********************************************************
1081   005502  000004                   TST12:  SCOPE
1082
1083   005504  004737  002700                   JSR     PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
1084   005510  012737  000140  177776           MOV     #PR3,PSW        ;PS=3
1085   005516  012777  005604  174134           MOV     #BRK,@BE1VEC    ;SET UP DEVICE INTR VEC
1086   005524  012777  000340  174130           MOV     #PR7,@BE1PSW    ;SET UP DEVICE PSW VEC
1087   005532                           DO14:
1088   005532  004737  011074                   JSR     PC,ADLI         ;TEST ADDR LINES 14 &15
1089   005536  152777  000001  174046           BISB    #1,@BE1CR2      ;ELSE SET BIT 0 OF CR2(ADDR LINE 16)
1090   005544  004737  011074                   JSR     PC,ADLI
1091   005550  042777  000001  174034           BIC     #1,@BE1CR2      ;CLEAR BIT 0(ADDR LINE 16)
1092   005556  152777  000002  174026           BISB    #2,@BE1CR2      ;SET BIT 1 OF CR2(ADDR LINE 17)
1093   005564  004737  011074                   JSR     PC,ADLI
1094   005570  152777  000003  174014           BISB    #3,@BE1CR2      ;ELSE SET BITS 0 AND 1 OF CR2
1095                                                                     ;SETS ADDR LINES 16 & 17
1096   005576  004737  011074                   JSR     PC,ADLI
1097   005602  000431                           BR      TST13           ;;GO TO NEXT TEST
1098                                     ;********************************************************
1099                                     ;********************************************************
1100   005604                           BRK:
1101   005604  011637  001172                   MOV     (SP),$REG5      ;FOR TYPEOUT OF PC
1102   005610  012737  000001  001204           MOV     #1,$TMP4        ;INDICATOR FOR DEVICE 1
1103   005616  032777  007340  173766           BIT     #7340,@BE1CR2   ;CHECK FOR ALL ERRS EXCEPT NO SSYN ERR
1104   005624  001003                           BNE     1$              ;IF ANY ARE SET,SEE WHICH ONES
1105   005626  005077  173756                   CLR     @BE1CLR         ;ELSE CLEAR THE NO SSYN ERR
1106   005632  000414                           BR      EXBRK           ;AND EXIT
1107   005634                           1$:
1108   005634  017737  173746  001162           MOV     @BE1CR1,$REG1   ;MOVES ARE FOR TYPEOUTS
1109   005642  017737  173744  001164           MOV     @BE1CR2,$REG2
1110   005650  017737  173730  001166           MOV     @BE1BA,$REG3
1111   005656  104011                           ERROR   11              ;ERR ON ACCESSING A14 - A17
1112   005660  004737  010406                   JSR     PC,ERRTN        ;DO ERR CHECK SUB-ROUTINE
1113   005664  000002                   EXBRK:  RTI                     ;EXIT
1114
1115
1116                                     ;********************************************************
1117                                     ;*TEST 13          CPU TEST FOR ACLO/DCLO SEQUENCE
1118                                             ;*THIS TEST CHECKS THE ASSERTION OF ACLO AND DCLO
1119                                             ;*AND THAT THE CPU TRAPS TO THE CORRECT SERVICE ROUTINE,
1120                                             ;*IF THIS PROGRAM IS RUNNING UNDER ACT11 THIS TEST
1121                                             ;*WILL BE SKIPPED.
1122                                     ;********************************************************
1123   005666  000004                   TST13:  SCOPE
1124   005670  012737  000001  001210           MOV     #1,$TIMES       ;;DO 1 ITERATION
1125   005676  005737  000042                   TST     42              ;SEE IF PROGRAM IS UNDER ACT11
1126   005702  001061                           BNE     TST14           ;;IF UNDER ACT, DO NOT PERFORM THIS TEST
1127   005704  012705  000001                   MOV     #1,R5           ;INIT R5 WITH A VALUE OF 1
1128   005710                           6$:
```

```
1129  005710  005205                            INC     R5              ;ADD 1 TO R5
1130  005712  100376                            BPL     6$              ;KEEP ADDING AS LONG AS R5   OS
1131  005714  012737  000001  001204            MOV     #1,$TMP4        ;INDICATOR FOR DEVICE 1
1132  005722  012737  000340  177776            MOV     #PR7,PSW        ;SET PS=7
1133  005730  012777  004730  173722            MOV     #ERRCHK,@BE1VEC ;SET UP INTR VECTOR
1134  005736  012777  000340  173716            MOV     #PR7,@BE1PSW    ;SET UP DEVICE INTR PSW
1135  005744  005037  001176                    CLR     $TMP1           ;CLEAR TEMPORARY REGISTER(TMP1)
1136  005750  012737  006026  000024            MOV     #TMPPWR,PWRVEC  ;SET UP SPECIAL POWER RTN
1137  005756  052777  000020  173626            BIS     #BIT04,@BE1CR2  ;INDICATE PWR FAILURE BY SETTING BIT 4
1138  005764  004737  011046                    JSR     PC,CNTR         ;PAUSE FOR TIME
1139  005770  012737  010342  000024            MOV     #PWRFAL,PWRVEC  ;RESTORE PWRFAL SEQ FOR A PWR FAIL
1140  005776  042777  000020  173606            BIC     #BIT04,@BE1CR2  ;MAKE SURE BIT 4 IS CLEARED
1141  006004                          FAILCK:
1142  006004  022737  001152  020320            CMP     #$NULL,$PWRMG   ;IF THIS TEST IS CAUSE OF
1143                                                                    ;PWR FAIL --TYPE NULL CHAR
1144  006012  001401                            BEQ     XTST            ;IF EQUAL, EXIT TEST
1145  006014  104013                            ERROR   13              ;TYPE ERR MSSG IF FAILURE
1146  006016                          XTST:
1147  006016  012737  020332  020320            MOV     #$POWER,$PWRMG  ;RESTORE TYPE OUT OF 'POWER'
1148  006024  000410                            BR      TST14           ;;GO TO NEXT TEST
1149                                  ;************************************************************
1150                                  ;************************************************************
1151  006026                          TMPPWR:                           ;SPECIAL PWR RTN; OTHER THAN SYSMAC'S
1152  006026  012737  001152  020320            MOV     #$NULL,$PWRMG   ;CHANGE PWR MSSG TO NULL CHAR
1153  006034  042777  000020  173550            BIC     #BIT04,@BE1CR2  ;CLEAR POWER DOWN/UP BIT
1154  006042  000137  020202                    JMP     $PWRDN          ;GO TO THAT RTN
1155
1156
1157                                  ;************************************************************
1158                                  ;*TEST 14      PARITY ERROR TEST
1159                                  ;*THIS TEST WILL CAUSE PARITY ERROR AND CHECKS
1160                                  ;*THAT THE CPU TRAPS TO THE CORRECT VECTOR.
1161                                  ;*THIS TEST SHOULD BE DESELECTED IF THE MEMORY
1162                                  ;*PARITY OPTION IS NOT PRESENT, ELSE AN
1163                                  ;*ERROR WILL BE REPORTED ALTHOUGH HARDWARE IS
1164                                  ;*FUNCTIONING PROPERLY.
1165                                  ;*SW06=1       INHIBIT TEST 14 AND GO TO NEXT TEST
1166                                  ;************************************************************
1167  006046  000004                  TST14:  SCOPE
1168  006050  032777  000100  173060            BIT     #BIT06, @SWR    ;INHIBIT TEST 14?
1169  006056  001105                            BNE     TST15           ;GO TO NEXT TEST
1170  006060  012737  006270  000010            MOV     #NODO,10        ;SET UP RESERVED INSTR VECTOR
1171  006066  012737  000340  000012            MOV     #PR7,12         ;PSW=7
1172  006074  005037  001202                    CLR     $TMP3           ;SET $TMP3 = 0
1173  006100  000270                            SEN                     ;SET N BIT OF CC
1174  006102  006737  001202                    SXT     $TMP3           ;IF VALID INSTR, $TMP3 WILL = -1
1175  006106  005737  001202                    TST     $TMP3           ;IF INVALID, $TMP3 WILL REMAIN 0
1176  006112  100033                            BPL     NXT             ;IF CP NOT= 35,40,45,OR 70,GO TO NEXT TEST
1177  006114  012737  000140  177776            MOV     #PR3,PSW        ;PS=3
1178  006122  012777  006234  173530            MOV     #PBERR,@BE1VEC  ;SET UP DEVICE INTR
1179  006130  012737  006256  000114            MOV     #PBRTN,PBVEC    ;SET UP PARITY BIT VECTOR
1180  006136  012737  000340  000116            MOV     #340,PBPSW      ;SET UP PARITY BIT PSW
1181  006144  012777  020342  173432            MOV     #ATEND,@BE1BA   ;SET UP ADDR REG
1182  006152  012777  177777  173422            MOV     #-1,@BE1CC      ;SET UP CYCLE COUNT
1183  006160  052777  010000  173424            BIS     #BIT12,@BE1CR2  ;SET BIT 12 FOR PARITY ERROR
1184  006166  005777  173420                    TST     @BE1CR2         ;SET OFF PARITY ERR SEQUENCE
```

```
1185  006172  012777  013161  173406          MOV    #13161,@BE1CR1  ;TRY FUN 1-DATO FROM CC-NPR-INTR ON DONE(7)
1186  006200  000240                           NOP                    ;ALLOW TIME FOR ATTEMPTED XFER
1187  006202                          NXT:
1188  006202  012737  000116  000114          MOV    #PBPSW,PBVEC    ;RESTORE
1189  006210  012737  000000  000116          MOV    #0,PBPSW        ;TRAP CATCHER HERE AND
1190  006216  012737  000012  000010          MOV    #12,10          ;AT RESERVED
1191  006224  012737  000000  000012          MOV    #0,12           ;INSTRUCTION VECTOR
1192  006232  000417                           BR     TST15          ;;BRANCH TO NEXT TEST IF PARITY TRAP OCCURRED
1193  006234                          PBERR:
1194  006234  011637  001172                  MOV    (SP),$REG5      ;FOR TYPEOUT OF PC
1195  006240  104014                           ERROR  14             ;TYPE ERR MSSG IF DEVICE INTERRUPTED
1196  006242  012737  000001  001204          MOV    #1,$TMP4        ;INDICATOR FOR DEVICE 1
1197  006250  004737  010406                  JSR    PC,ERRTN        ;CHECK TO SEE IF ANY ERRORS OCCURED
1198  006254  000002                           RTI                   ;EXIT TRAP
1199                                   ;********************************************************************
1200                                   ;********************************************************************
1201  006256                          PBRTN:                         ;PARITY BIT TRAP RTN
1202  006256  012777  000000  173326          MOV    #0,@BE1CR2      ;CLEAR PARITY BIT ERROR-MUST BE DONE
1203                                                                  ;BY MOVING 0(S) TO BE1CR2
1204  006264  012716  006202                  MOV    #NXT,(SP)       ;SET STACK FOR NEXT TEST
1205  006270  000002                  NODO:    RTI
1206
1207
1208                                   ;********************************************************************
1209                                   ;*TEST 15       MULTITRANSFERS I
1210                                   ;*THIS TEST WILL CAUSE ANY BUS EXERCISERS, UP TO 4,
1211                                   ;*TO CREATE A LOT OF TRAFFIC ON THE BUS AND
1212                                   ;*CHECK THAT THE CPU CAN HANDLE IT; ALL DEVICES
1213                                   ;*ARE SET OFF SIMULTANEOUSLY
1214                                   ;********************************************************************
1215  006272  000004                  TST15:  SCOPE
1216  006274  004737  002700                  JSR    PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
1217  006300  012703  000000                  MOV    #0,R3           ;SET DATA PATTERN = 0
1218  006304  012704  177777                  MOV    #177777,R4      ;SET DATA PATTERN = ALL 1'S
1219  006310  004737  007500                  JSR    PC,MULT1        ;LOAD & EXECUTE ALL DEVICES
1220  006314  022737  000002  001700          CMP    #2,DEVCNT       ;ARE THERE MORE THAN 2 DEVICES?
1221  006322  100115                           BPL    TST16          ;;IF 2 OR LESS, GO TO NEXT TEST
1222  006324  012703  161610                  MOV    #161610,R3      ;ELSE LOAD R3 AND R4 WITH
1223  006330  012704  016161                  MOV    #016161,R4      ;ANOTHER PATTERN
1224  006334                          5$:
1225  006334  123737  001115  001103          CMPB   $ERMAX,$ERFLG   ;MAX ERRS FOR THIS TEST OCCURRED?
1226  006342  100505                           BMI    TST16          ;;BR IF YES TO NEXT TEST
1227  006344  004737  010600                  JSR    PC,ROTATE       ;ROTATE DATA PATTERNS
1228  006350  004737  007500                  JSR    PC,MULT1        ;LOAD & EXECUTE ALL DEVICES
1229  006354  022703  107070                  CMP    #107070,R3      ;IS R3 = 107070?
1230  006360  001365                           BNE    5$             ;IF NOT,ROTATE AND DO AGAIN
1231  006362  012703  167777                  MOV    #167777,R3      ;ELSE MOVE NEW PATTERNS
1232  006366  012704  010000                  MOV    #010000,R4      ;INTO R3 AND R4
1233  006372                          10$:
1234  006372  123737  001115  001103          CMPB   $ERMAX,$ERFLG   ;MAX ERRS FOR THIS TEST OCCURRED?
1235  006400  100466                           BMI    TST16          ;;BR IF YES TO NEXT TEST
1236  006402  004737  010600                  JSR    PC,ROTATE       ;ROTATE DATA PATTERNS
1237  006406  004737  007500                  JSR    PC,MULT1        ;LOAD & EXECUTE ALL DEVICES
1238  006412  022703  167777                  CMP    #167777,R3      ;IS R3 = 167777 AGAIN?
1239  006416  001365                           BNE    10$            ;IF NOT,ROTATE AND DO AGAIN
1240  006420  000456                           BR     TST16          ;;GO TO NEXT TEST
```

```
1241                                    ;*********************************************************
1242                                    ;*********************************************************
1243                                    ;*********************************************************
1244    006422                          SERV1:
1245    006422  017737  173156  001712          MOV     @BE1BA,DATA1    ;MOVE ADDR IN BE1BA TO DATA1 AND
1246    006430  162737  000001  001712          SUB     #1,DATA1        ;SUB 1 TO GET ACTUAL ADDR
1247    006436  012737  000001  001204          MOV     #1,STMP4        ;INDICATOR FOR DEVICE 1
1248    006444  000435                          BR      INK             ;BRANCH TO INK
1249    006446                          SERV2:
1250    006446  017737  173146  001714          MOV     @BE2BA,DATA2    ;MOVE ADDR IN BE2BA TO DATA2 AND
1251    006454  162737  000002  001714          SUB     #2,DATA2        ;SUB 2 TO GET ACTUAL ADDR
1252    006462  012737  000002  001204          MOV     #2,STMP4        ;INDICATOR FOR DEVICE 2
1253    006470  000423                          BR      INK             ;BRANCH TO INK
1254    006472                          SERV3:
1255    006472  017737  173136  001716          MOV     @BE3BA,DATA3    ;MOVE ADDR IN BE3BA TO DATA3 AND
1256    006500  162737  000002  001716          SUB     #2,DATA3        ;SUB 2 TO GET ACTUAL ADDR
1257    006506  012737  000003  001204          MOV     #3,STMP4        ;INDICATOR FOR DEVICE 3
1258    006514  000411                          BR      INK             ;BRANCH TO INK
1259    006516                          SERV4:
1260    006516  017737  173126  001720          MOV     @BE4BA,DATA4    ;MOVE ADDR IN BE4BA TO DATA4 AND
1261    006524  162737  000002  001720          SUB     #2,DATA4        ;SUB 2 TO GET ACTUAL ADDR
1262    006532  012737  000004  001204          MOV     #4,STMP4        ;INDICATOR FOR DEVICE 4
1263    006540                          INK:
1264    006540  005237  001164                  INC     SREG2           ;INCREMENT REG
1265    006544  011637  001172                  MOV     (SP),SREG5      ;FOR TYPEOUT OF PC
1266    006550  004737  010406                  JSR     PC,ERRTN        ;CHECK FOR ANY ERRS
1267    006554  000002                          RTI                     ;EXIT
1268                                    ;*********************************************************
1269
1270                                    ;*********************************************************
1271                                    ;*TEST 16       MULTITRANSFERS II
1272                                            ;*THIS TEST WILL HAVE THE AVAILABLE EXERCISERS DOING
1273                                            ;*VARIOUS TRANSFERS AND/OR INTERRUPTS AT DIFFERENT
1274                                            ;*REQUEST LEVELS TO FURTHER CHECK CPU HANDLING CAPABILITIES
1275                                    ;*********************************************************
1276    006556  000004                  TST16:  SCOPE
1277    006560  012702  020342                  MOV     #ATEND,R2       ;R2 = END OF PROG
1278    006564  012705  005000                  MOV     #5000,R5        ;R5 = THE # OF DATA WORDS
1279    006570  004737  011030                  JSR     PC,DOUP         ;CREATE THOSE WORDS IN BUFFER MEMORY
1280    006574  004737  011142                  JSR     PC,TSTOVR       ;SET UP PATTERN IN MEMORY BUFFER AREA
1281    006600  004737  002700                  JSR     PC,CLRREG       ;CLEAR CONTENTS OF ALL AVAILABLE DEVS
1282
1283    006604  012737  000000  177776          MOV     #PR0,PSW        ;PS=0
1284    006612  012777  007344  173040          MOV     #S1,@BE1VEC     ;SET UP DEVICE 1 INTR VECTOR
1285    006620  012777  000340  173034          MOV     #PR7,@BE1PSW    ;SET UP DEVICE 1 PSW VECTOR
1286    006626  012777  022342  172750          MOV     #ATEND+2000,@BE1BA  ;SET UP ADDR REG
1287    006634  012777  176000  172740          MOV     #-2000,@BE1CC   ;SET UP CYCLE COUNT
1288    006642  012777  015551  172736          MOV     #15551,@BE1CR1  ;DO FUN 2-DATOB FROM CC-NPR-INTR ON DONE(6)
1289    006650  022737  000001  001700          CMP     #1,DEVCNT       ;CHECK FOR MORE THAN 1 DEVICE
1290    006656  001467                          BEQ     1S              ;IF NOT, GO CHECK RESULTS
1291
1292    006660  012777  007376  172776          MOV     #S2,@BE2VEC     ;SET UP DEVICE 2 INTR VECTOR
1293    006666  012777  000340  172772          MOV     #PR7,@BE2PSW    ;SET UP DEVICE 2 PSW VECTOR
1294    006674  012777  177000  172714          MOV     #-1000,@BE2CC   ;SET UP CYCLE COUNT FOR 1000 XFERS
1295    006702  012777  020342  172710          MOV     #ATEND,@BE2BA   ;SET UP ADDR REG=1ST LOCATION AFTER PROG
1296    006710  012777  002561  172704          MOV     #2561,@BE2CR1   ;DO FUN 1-DATIP-NPR-INTR ON DONE(7)
```

```
1297  006716  022737  000002  001700          CMP    #2,DEVCNT         ;CHECK FOR MORE THAN 2 DEVICES
1298  006724  001444                          BEQ    1$                ;IF NOT, GO CHECK RESULTS
1299
1300  006726  012777  007430  172734          MOV    #S3,@BE3VEC       ;SET UP DEVICE 3 INTR VECTOR
1301  006734  012777  000340  172730          MOV    #PR7,@BE3PSW      ;SET UP PSW VECTOR
1302  006742  012777  176776  172662          MOV    #-1002,@BE3CC     ;SET UP CYCLE COUNT
1303  006750  012777  020342  172656          MOV    #ATEND,@BE3BA     ;SET UP ADDR REG
1304  006756  012777  004005  172652          MOV    #4005,@BE3CR1     ;DO FUN 2-DATI-BR5
1305  006764  022737  000003  001700          CMP    #3,DEVCNT         ;CHECK FOR MORE THAN 3 DEVICES
1306  006772  001421                          BEQ    1$                ;IF NOT, GO CHECK RESULTS
1307
1308  006774  005037  001170                  CLR    $REG4             ;USE REG4 TO COUNT INTRS
1309  007000  012777  007446  172666          MOV    #S4,@BE4VEC       ;SET UP DEVICE 4 INTR VECTOR
1310  007006  012777  000340  172662          MOV    #PR7,@BE4PSW      ;SET UP PSW VECTOR
1311  007014  012777  175000  172624          MOV    #-3000,@BE4CC     ;SET UP CYCLE COUNT
1312  007022  052777  040000  172626          BIS    #BIT14,@BE4CR2    ;SET TIME DELAY BIT OF DEVICE 4
1313  007030  012777  000021  172614          MOV    #21,@BE4CR1       ;DO FUN 0-BR7
1314  007036                          1$:
1315  007036  012700  001606                  MOV    #BE1CR1,R0        ;USE R0 TO POINT TO DEVICE 1'S CR1
1316  007042  004737  010754                  JSR    PC,BKGD           ;PERFORM BACKGROUND ROUTINE

      ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
      ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/*  /* :/* :/* :
            ;BE1 TRANSFER CHECKS
      ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *
      ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *

1324  007046  012702  022342                  MOV    #ATEND+2000,R2    ;USE R2 TO POINT TO 2000 BYTES AFTER END OF PROGRAM
1325  007052  012737  176000  001164          MOV    #-2000,$REG2      ;SET UP $REG2 WITH EXPECTED RESULTS
1326  007060  023702  001712          10$:    CMP    DATA1,R2          ;CHECK FOR 4000 BYTES PAST END OF PROGRAM
1327  007064  001417                          BEQ    14$               ;IF EQUAL, GO CHECK FOR ANOTHER DEVICE
1328  007066  023722  001164                  CMP    $REG2,(R2)+       ;ELSE COMPARE EXPECTED RESULTS WITH THAT IN R2
1329  007072  001004                          BNE    12$               ;IF NOT EQUAL, GO TO ERR MSSG
1330  007074  062737  000002  001164          ADD    #2,$REG2          ;INCREMENT REG2 AS CC WOULD
1331  007102  000766                          BR     10$               ;AND GO BACK TO CHECK NEXT LOC
1332  007104                          12$:
1333  007104  014237  001162                  MOV    -(R2),$REG1       ;MOVE R2 TO REG 5 FOR TYPEOUT
1334  007110  010237  001160                  MOV    R2,$REG0
1335  007114  013737  001164  001166          MOV    $REG2,$REG3
1336  007122  104021                          ERROR  21                ;TYPE ERR MSSG
1337  007124                          14$:
1338  007124  022737  000001  001700          CMP    #1,DEVCNT         ;CHECK IF ONLY 1 DEVICE SHOULD HAVE OPERATED
1339  007132  001470                          BEQ    TST17             ;;<IF EQUAL, GO TO NEXT TEST>

      ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
      ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
            ;BE2 TRANSFER CHECKS
      ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *
      ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *

1347  007134  012701  020342                  MOV    #ATEND,R1         ;USE R1 TO POINT TO END OF PROG
1348  007140  023701  001714          5$:     CMP    DATA2,R1          ;CHECK FOR 2000 BYTES PAST END
1349  007144  001413                          BEQ    6$                ;IF EQUAL, EXIT
1350  007146  022721  052525                  CMP    #052525,(R1)+     ;ELSE COMPARE EXPECTED RESULTS WITH R1
1351  007152  001772                          BEQ    5$                ;IF EQUAL, GO CHECK NEXT LOC
1352  007154  014137  001162                  MOV    -(R1),$REG1       ;MOVE R1 TO REG1 FOR TYPEOUT
```

```
1353   007160  010137  001160              MOV     R1,$REG0
1354   007164  012737  052525  001166      MOV     #052525,$REG3
1355   007172  104020                      ERROR   20              ;ELSE TYPE ERR MSSG
1356   007174                      6$:
1357   007174  022737  000002  001700      CMP     #2,DEVCNT       ;CHECK IF ONLY 2 DEVICES OPERATED
1358   007202  001444                      BEQ     TST17           ;;<IF EQUAL, GO TO NEXT TEST>

       ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* ./* :/* :
       ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
             ;BE3 CHECK ROUTINE
       ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *
       ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *

1366   007204  105777  172426      16$:    TSTB    @BE3CR1         ;CHECK IF DEVICE 3 DONE
1367   007210  100401                      BMI     20$             ;IF YES, CHECK NEXT DEVICE
1368   007212  000774                      BR      16$             ;AND GO BACK TO SEE IF DONE YET
1369   007214                      20$:
1370   007214  022777  176002  172406      CMP     #-1776,@BE3DB   ;CHECK FOR CORR VALUE IN BE3DB
1371   007222  001407                      BEQ     25$             ;IF EQUAL, SKIP ERR TYPE OUT
1372   007224  017737  172400  001162      MOV     @BE3DB,$REG1    ;MOVE FOR ERR TYPE OUT
1373   007232  012737  176002  001164      MOV     #-1776,$REG2
1374   007240  104032                      ERROR   32              ;TYPE ERR MSSG
1375   007242                      25$:
1376   007242  022737  000003  001700      CMP     #3,DEVCNT       ;CHECK IF ONLY 3 DEVICES OPERATED
1377   007250  001421                      BEQ     TST17           ;;<IF EQUAL, GO TO NEXT TEST>

       ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
       ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
             ;BE4 CHECK ROUTINE
       ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *
       ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *

1385   007252                      22$:
1386   007252  105777  172374              TSTB    @BE4CR1         ;TEST IF DEVICE 4 IS DONE
1387   007256  100375                      BPL     22$             ;IF NOT, GO BACK TO DELAY RTN
1388   007260  022737  003000  001170      CMP     #3000,$REG4     ;CHECK IF REG4 COUNTED 3000 INTRS
1389   007266  001407                      BEQ     26$             ;IF EQUAL SKIP ERR TYPE OUT
1390   007270  013737  001170  001162      MOV     $REG4,$REG1     ;MOVE FOR TYPE OUT
1391   007276  012737  003000  001164      MOV     #3000,$REG2
1392   007304  104033                      ERROR   33              ;TYPE ERR MSSG
1393   007306                      26$:
1394   007306  042777  040000  172342      BIC     #BIT14,@BE4CR2  ;ELSE CLEAR TIME DELAY BIT
1395                               ;*********************************************************************
1396                               ;*TEST 17        DUMMY END OF PROGRAM
1397                               ;*********************************************************************
1398   007314  000004              TST17:  SCOPE
1399   007316  012737  000001  001210      MOV     #1,$TIMES       ;;DO 1 ITERATION
1400
1401   007324  017700  171614              MOV     @$TKB,R0        ;MOVE READ BUFF CONTENTS TO R0
1402   007330  022700  000210              CMP     #210,R0         ;DOES THE VALUE = " H" ?
1403   007334  001001                      BNE     10$             ;IF NOT GO TO 10$
1404   007336  000000                      HALT                    ;ELSE HALT THE PROGRAM
1405   007340                      10$:
1406   007340  000137  015446              JMP     $EOP
1407                               ;*********************************************************************
1408                               ;SET UP DEVICE INTR VECTOR
```

```
1409                                  ;*********************************************************
1410   007344                         S1:
1411
1412   007344  017737  172234  001712         MOV     @BE1BA,DATA1    ;MOVE ADDR IN BE1BA TO DATA1 AND
1413   007352  162737  000002  001712         SUB     #2,DATA1        ;SUB 2 TO GET ACTUAL ADDR
1414   007360  012737  000001  001204         MOV     #1,$TMP4        ;SET INDICATOR FOR DEVICE 1
1415   007366  005777  172214                 TST     @BE1CR1         ;TEST FOR ERROR
1416   007372  100041                         BPL     EXS             ;IF PLUS, EXIT
1417   007374  000434                         BR      CHEX            ;ELSE FIND CAUSE OF INTR
1418   007376                         S2:
1419   007376  017737  172216  001714         MOV     @BE2BA,DATA2    ;MOVE ADDR IN BE2BA TO DATA2 AND
1420   007404  162737  000002  001714         SUB     #2,DATA2        ;SUB 2 TO GET ACTUAL ADDR
1421   007412  012737  000002  001204         MOV     #2,$TMP4        ;SET INDICATOR FOR DEVICE 2
1422   007420  005777  172176                 TST     @BE2CR1         ;TEST FOR ERROR
1423   007424  100024                         BPL     EXS             ;IF PLUS EXIT
1424   007426  000417                         BR      CHEX            ;ELSE FIND CAUSE OF INTR
```

```
 1425  007430                           S3:
 1426  007430  012737  000003  001204        MOV     #3,$TMP4        ;SET INDICATOR FOR DEVICE 3
 1427  007436  005777  172174                TST     @BE3CR1         ;TEST FOR ERROR
 1428  007442  100015                        BPL     EXS             ;IF PLUS, EXIT
```

```
1429  007444  000410                        BR      CHEX        ;ELSE FIND CAUSE OF INTR
1430  007446                        S4:
1431  007446  005237  001170                INC     $REG4       ;COUNT DEVICE 4'S INTRS
```

```
1432   007452   012737   000004   001204          MOV     #4,$TMP4           ;SET INDICATOR FOR DEVICE 4
1433   007460   005777   172166                    TST     @BE4CR1            ;TEST FOR ERROR
1434   007464   100004                             BPL     EXS                ;IF PLUS, EXIT
1435   007466                             CHEX:
1436   007466   011637   001172                    MOV     (SP),$REG5         ;FOR TYPEOUT OF PC
1437   007472   004737   010406                    JSR     PC,ERRTN           ;ELSE FIND CAUSE OF INTR
1438   007476                             EXS:
1439   007476   000002                             RTI
1440                                      ;*************************************************************
1441                                      ;*************************************************************
1442   007500                             MULT1:
1443   007500   012702   020342                    MOV     #ATEND,R2          ;R2 = END OF PROG
1444   007504   012705   005000                    MOV     #5000,R5           ;R5 = THE # OF DATA WORDS
1445   007510   004737   011030                    JSR     PC,DOUP            ;CREATE THOSE WORDS IN BUFFER MEMORY
1446   007514   004737   011142                    JSR     PC,TSTOVR          ;SET UP PATTERN IN MEMORY BUFFER AREA
1447   007520   012777   020342   172056           MOV     #ATEND,@BE1BA      ;SET REG ADDR= 1ST LOCATION AFTER END OF PROGRAM
1448   007526   012777   176000   172046           MOV     #-2000,@BE1CC      ;SET CYCLE COUNT FOR 2000 XFERS
1449   007534   012777   006422   172116           MOV     #SERV1,@BE1VEC     ;SET UP DEVICE INTR VECTOR
1450   007542   012777   000340   172112           MOV     #PR7,@BE1PSW       ;SET UP DEVICE PSW VECTOR
1451   007550   052777   040000   172034           BIS     #BIT14,@BE1CR2     ;SET BIT 14 FOR TIME DELAY ENABLE
1452   007556   012777   042560   172022           MOV     #42560,@BE1CR1     ;DO DATIP/DATOB-FUN 1-NPR-INTR ON DONE(7)
1453   007564   122737   000001   001700           CMPB    #1,DEVCNT          ;IF MORE THAN 1 DEVICE, LOAD THEIR REGISTERS
1454   007572   001474                             BEQ     6$                 ;OTHERWISE BEGIN TESTING
1455   007574                             3$:
1456   007574   012777   006446   172062           MOV     #SERV2,@BE2VEC     ;SET UP DEVICE 2 INTR VECTOR
1457   007602   012777   000340   172056           MOV     #PR7,@BE2PSW       ;SET UP DEVICE 2 PSW VECTOR
1458   007610   012777   020342   172002           MOV     #ATEND,@BE2BA      ;SET UP ADDR REG FOR SAME LOCATIONS AS DEVICE 1
1459   007616   012777   177000   171772           MOV     #-1000,@BE2CC      ;SET CYCLE COUNT FOR A 1000 XFERS
1460   007624   012777   024510   171770           MOV     #24510,@BE2CR1     ;DO DATIP/NO ROTATE-FUN 2-BR6-INTR ON DONE(6)
1461   007632   122737   000002   001700           CMPB    #2,DEVCNT          ;IF MORE THAN 2 DEVICES, LOAD THER REGISTERS
1462   007640   001451                             BEQ     6$                 ;OTHERWISE BEGIN TESTING
1463   007642                             4$:
1464   007642   012777   000340   172022           MOV     #PR7,@BE3PSW       ;SET UP DEVICE 3 PSW VECTOR
1465   007650   010377   171754                    MOV     R3,@BE3DB          ;MOVE PATTERN IN R3 TO DEVICE DATA REG
1466   007654   012777   006472   172006           MOV     #SERV3,@BE3VEC     ;SET UP DEVICE INTR VECTOR
1467   007662   012777   022342   171744           MOV     #ATEND+2000,@BE3BA  ;SET UP ADDR REG
1468   007670   012777   177000   171734           MOV     #-1000,@BE3CC      ;SET UP FOR 1000 XFERS
1469   007676   052777   040000   171736           BIS     #BIT14,@BE3CR2     ;SET BIT 14 FOR TIME DELAY ENABLE
1470   007704   012777   003160   171724           MOV     #3160,@BE3CR1      ;DO DATO-FUN 1-FROM DB-NPR-INTR ON DONE(7)
1471   007712   122737   000003   001700           CMPB    #3,DEVCNT          ;IF A 4TH DEVICE, GO AND LOAD REGISTERS
1472   007720   001421                             BEQ     6$                 ;OTHERWISE BEGIN TESTING
1473   007722                             5$:
1474   007722   010477   171716                    MOV     R4,@BE4DB          ;MOVE PATTERN IN R4 TO DEVICE DATA REG
1475   007726   012777   000340   171742           MOV     #PR7,@BE4PSW       ;SET UP DEVICE 4 PSW VECTOR
1476   007734   012777   006516   171732           MOV     #SERV4,@BE4VEC     ;SET UP DEVICE 4 INTR VECTOR
1477   007742   012777   024342   171700           MOV     #ATEND+4000,@BE4BA  ;SET UP ADDR REG
1478   007750   012777   177000   171670           MOV     #-1000,@BE4CC      ;SET CYCLE COUNT FOR 1000 XFERS
1479   007756   012777   003104   171666           MOV     #3104,@BE4CR1      ;DO DATO-FUN 1-BR5-INTR ON DONE
1480   007764                             6$:
1481   007764   005277   171606                    INC     @SIMLGO            ;START DEVICES SIMULTANEOUSLY

                 ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
                 ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
                 ;BACKGROUND ROUTINE FOR MULTITRANSFERS I

1487   007770   012737   000001   001162           MOV     #1,$REG1           ;MOVE 1 TO TEMPORARY REG
```

```
1488   007776  012701  001162              MOV    #$REG1,R1        ;SET UP R1 AS POINTER
1489   010002  005121                 7$:  COM    (R1)+            ;COMPLEMENT TEMP REG
1490   010004  006041                      ROR    -(R1)            ;ROTATE CONTENTS RIGHT
1491   010006  123737  001700  001164      CMPB   DEVCNT,$REG2     ;CHECK IF ALL DEVICES ARE DONE
1492   010014  101372                      BHI    7$               ;IF NOT, CONTINUE WITH BACKGROUND RTN
1493                                        ;***************************************************************

       ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
       ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
                 ;DEVICE 1 TRANSFER CHECKS
                 ;THERE ARE NO CHECKS FOR BE2
       ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *
       ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *

1502   010016  042777  040000  171566      BIC    #BIT14,@BE1CR2   ;CLEAR TIME DELAY BIT
1503   010024  012700  020342              MOV    #ATEND,R0        ;START CHECKING FOR CORRECT XFERS
1504   010030                         8$:
1505   010030  122720  000125              CMPB   #125,(R0)+       ;COMPARE LOWER BYTE
1506   010034  001012                      BNE    20$              ;IF NOT EQUAL, BR TO ERR MSSG
1507   010036  023700  001712              CMP    DATA1,R0         ;IS THIS LAST BYTE COMPARE?
1508   010042  001422                      BEQ    9$               ;IF SO, BR TO 9$
1509   010044  122720  000124              CMPB   #124,(R0)+       ;ELSE COMPARE UPPER BYTE
1510   010050  001006                      BNE    22$              ;IF NOT EQUAL, BR TO ERR MSSG
1511   010052  023700  001712              CMP    DATA1,R0         ;IS THIS LAST BYTE COMPARE?
1512   010056  001414                      BEQ    9$               ;IF SO, BR TO 9$
1513   010060  000763                      BR     8$               ;ELSE CHECK NEXT ADDR
1514   010062                         20$:
1515   010062  105740                      TSTB   -(R0)            ;SUB 1 TO GET ERR ADDR
1516   010064  000401                      BR     24$              ;GO DO MOVES FOR ERR MSSG
1517   010066                         22$:
1518   010066  005740                      TST    -(R0)            ;SUB 2 TO GET ERR ADDR
1519   010070                         24$:
1520   010070  011037  001162              MOV    (R0),$REG1       ;MOVES ARE FOR ERR MSSG
1521   010074  010037  001160              MOV    R0,$REG0
1522   010100  012737  052125  001166      MOV    #052125,$REG3
1523   010106  104015                      ERROR  15               ;ELSE TYPE ERR MSSG
1524   010110                         9$:
1525   010110  022737  000001  001700      CMP    #1,DEVCNT        ;IF ONLY ONE DEVICE
1526   010116  001447                 25$: BEQ    13$              ;IF NO MORE DEVICES,EXIT RTN
1527   010120  122737  000002  001700      CMPB   #2,DEVCNT        ;CHECK FOR MORE THAN 2 DEVICES
1528   010126  001773                      BEQ    25$              ;IF NOT, EXIT TEST

       ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
       ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
                 ;BE3 TRANSFER CHECKS
       ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *
       ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *

1536   010130  042777  040000  171504      BIC    #BIT14,@BE3CR2   ;CLEAR TIME DELAY BIT OF DEVICE 3
1537   010136  012700  022342              MOV    #ATEND+2000,R0   ;CHECK NEXT 2000 LOCATIONS
1538   010142  023700  001716         10$: CMP    DATA3,R0         ;CHECK FOR 1000 XFERS
1539   010146  001411                      BEQ    11$              ;IF SO, CHECK NEXT BLOCK
1540   010150  020320                      CMP    R3,(R0)+         ;TEST FOR CORRECT PATTERNS
1541   010152  001773                      BEQ    10$              ;IF NO ERR, CHECK ANOTHER LOC
1542   010154  010337  001166              MOV    R3,$REG3         ;THE MOVE IS FOR TYPEOUT REASONS
1543   010160  014037  001162              MOV    -(R0),$REG1
```

```
 1544  010164  010037  001160            MOV    RO,$REGO
 1545  010170  104016                    ERROR  16               ;ELSE TYPE ERR MSSG
 1546  010172                     11$:
 1547  010172  122737  000003  001700    CMPB   #3,DEVCNT        ;CHECK FOR MORE THAN 3 DEVICES
 1548  010200  001416                    BEQ    13$              ;IF NO MORE DEVICES GO DO NEXT PATTERN

       ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
       ;/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :/* :
             ;BE4 TRANSFER CHECKS
       ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *
       ;/: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: */: *

 1556  010202  012700  024342            MOV    #ATEND+4000,RO   ;CHECK NEXT BLOCK OF 2000 LOCATIONS
 1557  010206  023700  001720     12$:   CMP    DATA4,RO         ;CHECK FOR 1000 XFERS
 1558  010212  001411                    BEQ    13$              ;IF SO, CHANGE DATA PATTERNS & START OVER AGAIN
 1559  010214  020420                    CMP    R4,(RO)+         ;ELSE CHECK FOR CORRECT PATTERNS
 1560  010216  001773                    BEQ    12$              ;IF NO ERR, CHECK ANOTHER LOCATION
 1561  010220  010437  001166            MOV    R4,$REG3         ;THE MOVE IS FOR TYPEOUT REASONS
 1562  010224  014037  001162            MOV    -(RO),$REG1
 1563  010230  010037  001160            MOV    RO,$REGO
 1564  010234  104017                    ERROR  17               ;ELSE TYPE ERR MSSG
 1565  010236                     13$:
 1566  010236  000207                    RTS    PC
 1567                             ;///////////////////////////////////////////////////////////////////////
 1568                             ;///////////////////////////////////////////////////////////////////////
 1569                             ;///////////////////////////////////////////////////////////////////////
 1570  010240                     STVEC:
 1571  010240  005000                    CLR    RO               ;RO IS COUNTER
 1572  010242  012701  010276            MOV    #Q1,R1           ;R1 IS ADDR OF INTR RTN
 1573  010246  012702  000510            MOV    #510,R2          ;R2 IS DEV INTR VEC ADDR
 1574  010252                     5$:
 1575  010252  010122                    MOV    R1,(R2)+         ;LOAD INTR RTN INTO INTR VEC
 1576  010254  012722  000340            MOV    #PR7,(R2)+       ;LOAD INTR PSW
 1577  010260  062701  000006            ADD    #6,R1            ;GET NEXT INTR RTN
 1578  010264  005200                    INC    RO               ;INC COUNTER
 1579  010266  020037  001700            CMP    RO,DEVCNT        ;ARE ALL AVAILABLE VECS LOADED?
 1580  010272  001367                    BNE    5$               ;IF NOT, GO AND LOAD NEXT ONE
 1581  010274  000207                    RTS    PC               ;ELSE EXIT
 1582                             ;********************************************************************
 1583                             ;********************************************************************
 1584
 1585                             ; THE FOLLOWING INTR ROUTINES SHOULD HANDLE ANY ERRORS
 1586                             ; WHICH MIGHT OCCUR BEFORE THE PROGRAM HAS A CHANCE TO
 1587                             ; PROPERLY CHECK OUT ALL DEVICES AND SET UP THEIR INTR VECTORS
 1588
 1589
 1590  010276                     Q1:
 1591  010276  012737  000001  001204    MOV    #1,$TMP4         ;INDICATOR FOR DEV 1
 1592  010304  000413                    BR     XQ
 1593  010306                     Q2:
 1594  010306  012737  000002  001204    MOV    #2,$TMP4         ;INDICATOR FOR DEV 2
 1595  010314  000407                    BR     XQ
 1596  010316                     Q3:
 1597  010316  012737  000003  001204    MOV    #3,$TMP4         ;INDICATOR FOR DEV 3
 1598  010324  000403                    BR     XQ
 1599  010326                     Q4:
```

```
1600  010326  012737  000004  001204          MOV     #4,$TMP4        ;INDICATOR FOR DEV 4
1601  010334                           XQ:
1602  010334  004737  010406                   JSR     PC,ERRTN        ;GO TO ERR ROUTINE
1603  010340  000002                           RTI
1604                              ;****************************************************************
1605                              ;****************************************************************
1606  010342                           PWRFAL:
1607  010342  010146                           MOV     R1,-(SP)        ;SAVE CONTENTS OF R1
1608  010344  010046                           MOV     R0,-(SP)        ;SAVE CONTENTS OF R0
1609  010346  012700  001612                   MOV     #BE1CR2,R0      ;R0 POINTS TO DEV 1 CR2 ADDR
1610  010352  005001                           CLR     R1              ;CLEAR R1
1611  010354                           5$:
1612  010354  042770  000020  000000           BIC     #BIT04,@(R0)    ;CLR BIT 4 OF CURRENT CR2
1613  010362  062700  000014                   ADD     #14,R0          ;ADD 14 TO POINT TO NEXT CR2
1614  010366  005201                           INC     R1              ;COUNT THE NUMBER OF DEVS
1615  010370  020137  001700                   CMP     R1,DEVCNT       ;REACHED MAX # ON BUS?
1616  010374  103767                           BLO     5$              ;IF NOT, CLR NEXT CR2
1617  010376  012600                           MOV     (SP)+,R0        ;ELSE RESTORE R0
1618  010400  012601                           MOV     (SP)+,R1        ;AND R1
1619  010402  000137  020202                   JMP     $PWRDN          ;THEN DO REGULAR PWR DOWN RTN
1620                              ;****************************************************************
1621                              ;****************************************************************
1622  010406                           ERRTN:
1623  010406  104407                           SAVREG                  ;SAVE REGISTERS
1624  010410  012700  001576                   MOV     #BE1CR2-14,R0 ;INITIALIZE R0
1625  010414  105005                           CLRB    R5              ;CLEAR DEVICE COUNTER
1626  010416                           1$:
1627  010416  105205                           INCB    R5              ;ADD 1 TO COUNTER
1628  010420  062700  000014                   ADD     #14,R0          ;SET R0=ADDR OF CR2 OF NEXT DEVICE
1629  010424  120537  001204                   CMPB    R5,$TMP4        ;IF COUNTER NOT EQUAL TO INDICATOR
1630  010430  001372                           BNE     1$              ;ADD 1 TO COUNTER & CHECK AGAIN
1631  010432                           CHKERR:
1632  010432  105770  000000                   TSTB    @(R0)           ;CHECK FOR NO NOSACK TIMEOUT
1633  010436  100001                           BPL     1$              ;IF NOT, SEE IF THERE ARE ANY ERRS
1634  010440  104022                           ERROR   22              ;TYPE ERR MESSG FOR NO NOSACK
1635  010442                           1$:
1636  010442  032770  007540  000000           BIT     #7540,@(R0)     ;CHECK FOR OTHER ERRORS
1637  010450  001436                           BEQ     LEEV            ;IF NO ERRORS,EXIT
1638  010452  032770  004000  000000           BIT     #BIT11,@(R0)    ;CHECK FOR NO SSYN ON INTR
1639  010460  001401                           BEQ     10$             ;IF NOT SET, CHECK FOR NEXT ERR
1640  010462  104023                           ERROR   23              ;TYPE ERR MSSG FOR NO SSYN ON INTR
1641  010464                           10$:
1642  010464  132770  000040  000000           BITB    #BIT05,@(R0)    ;CHECK FOR WRONG GRANT ERR
1643  010472  001401                           BEQ     2$              ;IF NOT, CHECK BIT 6
1644  010474  104024                           ERROR   24              ;ELSE TYPE ERR MESSG FOR WRONG GRANT
1645  010476                           2$:
1646  010476  132770  000100  000000           BITB    #BIT06,@(R0)    ;CHECK FOR BUS LATE ERR
1647  010504  001401                           BEQ     3$              ;IF NOT, CHECK BIT 8
1648  010506  104025                           ERROR   25              ;TYPE ERR MSSG FOR BUS LATE
1649  010510                           3$:
1650  010510  032770  000400  000000           BIT     #BIT08,@(R0)    ;CHECK FOR NO SSYN ERR
1651  010516  001401                           BEQ     4$              ;IF NOT, CHECK BIT 9
1652  010520  104026                           ERROR   26              ;TYPE ERR MSSG FOR NO SSYN
1653  010522                           4$:
1654  010522  032770  001000  000000           BIT     #BIT09,@(R0)    ;CHECK FOR WRONG ADDR ERR
1655  010530  001401                           BEQ     5$              ;IF NOT, CHECK BIT 10
```

```
1656  010532  104027                          ERROR   27              ;TYPE ERR MSSG FOR WRONG ADDR
1657  010534                          5$:
1658  010534  032770  002000  000000          BIT     #BIT10,a(R0)    ;CHECK FOR NO GRANT ERR
1659  010542  001401                          BEQ     LEEV            ;IF NOT, EXIT
1660  010544  104030                          ERROR   30              ;TYPE ERR MSSG FOR NO GRANT
1661  010546                          LEEV:
1662  010546  162700  000002                  SUB     #2,R0           ;POINT TO DEVICE CLEAR REG
1663  010552  005070  000000                  CLR     a(R0)           ;CLEAR ALL ERRORS
1664  010556  104410                          RESREG                  ;RESTORE REGISTERS
1665  010560  000207                          RTS     PC              ;EXIT
1666                                   ;*********************************************************************
1667                                   ;*********************************************************************
1668                                   ;*********************************************************************
1669  010562                          THRU0:
1670  010562  011637  001164                  MOV     (SP),$REG2      ;MOVE FOR ERR TYPE OUT
1671  010566  162737  000002  001164          SUB     #2,$REG2        ;SUB 2 FOR ACTUAL ADDR
1672  010574  104035                          ERROR   35              ;TYPE ERR MSSG
1673  010576  000002                          RTI
1674                                   ;*********************************************************************
1675                                   ;*********************************************************************
1676  010600                          ROTATE:
1677  010600  032703  000001                  BIT     #BIT00,R3       ;IS LSB A 1 OR 0 ?
1678  010604  001402                          BEQ     5$              ;IF 0, GO TO 5$
1679  010606  000261                          SEC                     ;ELSE SET C BIT OF COND CODES
1680  010610  000401                          BR      10$             ;AND GO ROTATE
1681  010612                          5$:
1682  010612  000241                          CLC                     ;CLEAR C BIT OF COND CODES
1683  010614                          10$:
1684  010614  006003                          ROR     R3              ;ROTATE R3
1685  010616  032704  000001                  BIT     #BIT00,R4       ;IS LSB A 1 OR 0 ?
1686  010622  001402                          BEQ     15$             ;IF 0, GO TO 15$
1687  010624  000261                          SEC                     ;ELSE SET C BIT OF COND CODES
1688  010626  000401                          BR      20$             ;AND GO ROTATE
1689  010630                          15$:
1690  010630  000241                          CLC                     ;CLEAR C BIT OF COND CODES
1691  010632                          20$:
1692  010632  006004                          ROR     R4              ;ROTATE R4
1693  010634  000207                          RTS     PC
1694                                   ;*********************************************************************
1695                                   ;*********************************************************************
1696  010636                          NOG:
1697  010636                          2$:
1698  010636  010037  177776                  MOV     R0,PSW          ;SET UP PROCESSOR STATUS
1699  010642  012777  020342  170734          MOV     #ATEND,aBE1BA   ;SET UP ADDR REG
1700  010650  012777  177777  170724          MOV     #-1,aBE1CC      ;SET UP CYCLE COUNT FOR 1 CYCLE
1701  010656  010177  170724                  MOV     R1,aBE1CR1      ;DO FUN 1 ON BR LEVELS IN R1
1702  010662  000240                          NOP                     ;WAIT FOR DEVICE TO ATTEMPT TO  DO XFER
1703  010664  022777  177777  170710          CMP     #-1,aBE1CC      ;SEE IF DEVICE OPERATED
1704  010672  001005                          BNE     4$              ;IF IT DID,GO TYPE ERR MSSG
1705  010674  106201                          ASRB    R1              ;SHIFT BYTE RIGHT TO LOWER BRR0
1706  010676  122701  000001                  CMPB    #1,R1           ;IF BYTE IS NOT EQUAL TO 1
1707  010702  001355                          BNE     2$              ;GO TO 2$
1708  010704  000402                          BR      EXNOG           ;EXIT
1709  010706                          4$:
1710  010706  004737  010714                  JSR     PC,ERRS
1711  010712  000207                  EXNOG:  RTS     PC              ;EXIT SUB RTN
```

```
1712                              ;****************************************************************
1713   010714                     ERRS:
1714   010714   017737  170660  001160        MOV     @BE1DB,$REG0      ;MOVES ARE FOR TYPEOUTS
1715   010722   017737  170654  001162        MOV     @BE1CC,$REG1
1716   010730   017737  170650  001164        MOV     @BE1BA,$REG2
1717   010736   017737  170644  001166        MOV     @BE1CR1,$REG3
1718   010744   010037  001170               MOV     R0,$REG4
1719   010750   104002                        ERROR   2
1720   010752   000207                        RTS     PC                ;EXIT ERROR RTN
1721                              ;****************************************************************
1722                              ;****************************************************************
1723   010754                     BKGD:
1724   010754   012737  031463  001162        MOV     #031 53,$REG1     ;START OF BACKGROUND ROUTINE
1725   010762   012701  001163               MOV     #$REG1+1,R1        ;USE R1 TO POINT TO TEST PATTERN
1726   010766   105441             1$:        NEGB    -(R1)             ;DECREMENT LOC AND NEGATE BYTE=(031715)
1727   010770   105421                        NEGB    (R1)+             ;NEGATE BYTE THEN INCREMENT LOC=(031463)
1728   010772   105421                        NEGB    (R1)+             ;NEGATE BYTE THEN INCREMENT LOC=(146463)
1729   010774   105770  000000               TSTB    @(R0)             ;TEST FOR DONE BIT OF DEVICE IN R0
1730   011000   100402                        BMI     2$                ;IF DONE, GO CHECK RESULTS
1731   011002   105441                        NEGB    -(R1)             ;ELSE DECREMENT LOC AND NEGATE BYTE=(031463)
1732   011004   000770                        BR      1$                ;CONTINUE WITH BACKGROUND
1733   011006                     2$:
1734
1735   011006   005741                        TST     -(R1)             ;BRING POINTER DOWN TO REG1
1736   011010   022711  146463               CMP     #146463,(R1)      ;COMPARE EXPECTED PATTERN WITH THAT IN R1
1737   011014   001404                        BEQ     BKEX              ;IF EQUAL, EXIT THIS RTN
1738   011016   012737  146463  001164        MOV     #146463,$REG2     ;MOVE FOR TYPE OUT
1739   011024   104031                        ERROR   31                ;ELSE TYPE ERR MSSG
1740   011026   000207             BKEX:      RTS     PC
1741                              ;****************************************************************
1742                              ;****************************************************************
1743   011030                     DOUP:
1744   011030   012701  000001               MOV     #1,R1             ;INIT R1 TO 1
1745   011034                     5$:
1746   011034   010122                        MOV     R1,(R2)+          ;MOVE CONTENTS OF R1 TO AREA IN R2
1747   011036   005201                        INC     R1                ;ADD 1 TO R1
1748   011040   020105                        CMP     R1,R5             ;IS # OF MOVES = TO # IN R5?
1749   011042   101774                        BLOS    5$                ;IF NOT, DO ANOTHER MOVE
1750   011044   000207                        RTS     PC                ;ELSE EXIT
1751                              ;****************************************************************
1752                              ;****************************************************************
1753
1754   011046                     CNTR:
1755   011046   012737  000001  001170        MOV     #1,$REG4          ;INITIALIZE COUNTER REG
1756   011054   062737  000001  001170 1$:    ADD     #1,$REG4          ;ADD 1 TO IT
1757   011062   022737  000106  001170        CMP     #70.,$REG4        ;DELAY AT LEAST 41 US
1758   011070   001371                        BNE     1$                ;IF NOT, GO BACK AND ADD 1 TO REG4
1759   011072   000207                        RTS     PC                ;EXIT
1760                              ;****************************************************************
1761                              ;****************************************************************
1762   011074                     ADLI:
1763   011074   012700  040000               MOV     #40000,R0         ;USE R0 TO SET BIT 14
1764   011100                     1$:
1765   011100   012777  177777  170474        MOV     #-1,@BE1CC        ;SET CYCLE COUNT = 1 XFER
1766   011106   010077  170472               MOV     R0,@BE1BA         ;SET ADDR AS SPECIFIED IN R0
1767   011112   012777  002041  170466        MOV     #2041,@BE1CR1     ;DO DATI-FUN 1-NPR
```

```
1768  011120  004737  011046              JSR     PC,CNTR        ;ALLOW TIME FOR RDY BIT TO SET
1769  011124  022700  100000              CMP     #100000,R0     ;CHECK IF BIT 15 OF R0 IS SET
1770  011130  001403                      BEQ     EXAD           ;IF SET, GO SET NEXT ADDR LINE
1771  011132  012700  100000              MOV     #100000,R0     ;ELSE, NOW SET BIT 15 OF R0
1772  011136  000760                      BR      1$             ;GO BACK AND CHECK THAT ADDR
1773  011140                      EXAD:
1774  011140  000207                      RTS     PC             ;EXIT SUB ROUTINE
1775                          ;*********************************************************
1776                          ;*********************************************************
1777  011142                      TSTOVR:
1778  011142  012737  000140  177776      MOV     #PR3,PSW       ;PS=3
1779  011150  005037  001164              CLR     SREG2          ;CLEAR REG FOR INTR ON DONF COUNTER
1780  011154  012700  020342              MOV     #ATEND,R0      ;SET UP R0 AS POINTER
1781  011160  012720  125252      1$:     MOV     #125252,(R0)+  ;MOVE DATA PATTERN TO AVAILABLE MEMORY
1782  011164  022700  022342              CMP     #ATEND+2000,R0 ;CHECK FOR A 2000 MOVES
1783  011170  001373                      BNE     1$             ;IF NOT, GO BACK AND MOVE AGAIN
1784  011172  000207                      RTS     PC             ;EXIT
1785                          ;*********************************************************
1786                          ;*********************************************************
1787
1788  011174  005015  005015  047125  QNO:    .ASCIZ  <15><12><15><12> UNIBUS SYSTEMS EXERCISER DIAGNOSTIC - CZKUA-D    BY:M.S
      011277     015  042012  053105  FOR4:   .ASCIZ  <15><12> DEV 4 MUST HAVE TIME DELAY SET @ 100 US OR LATENCY ERR MAY OCCU
      011404  050103  020125  051124  EM1:    .ASCIZ  CPU TRAPPED THRU LOCATION 4 -TIMEOUT
      011452  040440  042104  020122  DH1:    .ASCIZ  ADDR    SERRPC #ERR/TST#
      011503     103  052520  044440  EM2:    .ASCII  CPU ISSUED A BUS GRANT WITH PSW = 7
      011546  042504  020126  020061          .ASCIZ  DEV 1 SHOULD NOT HAVE BECOME BUS MASTER
      011616  042502  042061  020102  DH2:    .ASCIZ  BE1DB   BE1CC   BE1BA   BE1CR1   PSW    SERRPC #ERR/TST#
      011706  050103  020125  044504  EM3:    .ASCIZ  CPU DID NOT ISSUE BUS NPG
      011740  042502  041461  030522  DH3:    .ASCIZ  BE1CR1   BE1CC   FM-PS   TO-PS    SERRPC #ERR/TST#
      012021     103  052520  042040  EM4:    .ASCIZ  CPU DID NOT ISSUE BUS GRANT 7
      012057     103  052520  042040  EM5:    .ASCIZ  CPU DID NOT ISSUE BUS GRANT 6
      012115     103  052520  042040  EM6:    .ASCIZ  CPU DID NOT ISSUE BUS GRANT 5
      012153     103  052520  042040  EM7:    .ASCIZ  CPU DID NOT ISSUE BUS GRANT 4
      012211     117  042516  047440  EM10:   .ASCIZ  ONE OR MORE DEVICES DID NOT INTERRUPT
      012257     124  044510  020123  DH10:   .ASCII  THIS IS THE ORDER IN WHICH THEY INTERRUPTED <15><12>
      012334  020040  051461  020124          .ASCIZ   1ST     2ND     3RD     4TH     SERRPC #ERR/TST#
      012415     102  051525  040440  EM11:   .ASCIZ  BUS ADDRESS LINES <A17:A14> DID NOT FUNCTION PROPERLY
      012503     102  030505  051103  DH11:   .ASCIZ  BE1CR1  BE1CR2   BE1BA   SERRPC #ERR/TST#
      012554  050103  020125  047516  EM12:   .ASCIZ  CPU NO SACK TIMEOUT LOGIC FAILED(TO NEGATE BUS GRANT)
      012642  042502  041461  030522  DH12:   .ASCIZ  BE1CR1  BE1CR2  SERRPC #ERR/TST#
      012703     103  052520  042040  EM13:   .ASCIZ  CPU DID NOT PROPERLY EXECUTE AN ACLO/DCLO SEQUENCE
      012766  042444  051122  041520  DH13:   .ASCIZ  SERRPC #ERR/TST#
      013007     103  052520  042040  EM14:   .ASCIZ  CPU DID NOT TRAP FROM BUS PARITY ERROR PA/PB = 0/1
      013072  042504  020126  020061  EM15:   .ASCII  DEV 1 DID DATIP WITH ROL ON DATOB TO MEMORY <15><12>
      013147     124  042510  052040          .ASCIZ  THE TRANSFER TO THE FOLLOWING LOCATION WAS  INCORRECT
      013235     115  046505  051117  DH15:   .ASCII  MEMORY  ACTUAL  CORRECT <15><12>
      013266  046040  041517  020040          .ASCIZ   LOC     DATA    DATA    SERRPC #ERR/TST# SICNT #
      013347     104  053105  041511  EM16:   .ASCIZ  DEVICE 3'S DATO TO MEMORY DID NOT EQUAL PATTERN IN R3
      013435     104  053105  041511  EM17:   .ASCIZ  DEVICE 4'S DATO TO MEMORY DID NOT EQUAL PATTERN IN R4
      013523     104  053105  041511  EM20:   .ASCIZ  DEVICE 1 DOING FUN 1-NPR-DATIP; INCORRECT PATTERN IN MEMORY
      013617     104  053105  041511  EM21:   .ASCIZ  DEVICE 2 DOING FUN 2-NPR-DATOB; INCORRECT PATTERN IN MEMCRY
      013713     102  052111  033440  EM22:   .ASCIZ  BIT 7 OF CR2 SET-CPU DID NOT TIME OUT WITH SACK INHIBITED
      014005     104  053105  021011  DH22:   .ASCIZ  DEV #    PC      SERRPC #ERR/TST#
      014047     102  052111  030440  EM23:   .ASCIZ  BIT 11 OF CR2 SET-NO SSYN ON INTR SIGNAL
      014120  044502  020124  020065  EM24:   .ASCIZ  BIT 5 OF CR2 SET-RECEIVED WRONG GRANT
      014166  044502  020124  020066  EM25:   .ASCIZ  BIT 6 OF CR2 SET-BUS LATE
```

```
      014220  044502  020124  020070  EM26:  .ASCIZ    BIT 8 OF CR2 SET-NO SSYN OCCURRED
      014262  044502  020124  020071  EM27:  .ASCIZ    BIT 9 OF  CR2 SET-WRONG ADDRESS ON BUS
      014331     102  052111  030440  EM30:  .ASCIZ    BIT 10 OF CR2 SET-DEVICE RECEIVED OTHER THAN ONE GRANT
      014420  045502  047107  020104  EM31:  .ASCII    BKGND ROUTINE INSTRUCTIONS OF NEGB'S WERE NOT DONE <15><12>
      014504  047503  051122  041505         .ASCIZ    CORRECTLY TO $REG1 DURING MULTITRANSFERS II
      014560  041501  052524  046101  DH31:  .ASCII    ACTUAL  CORR'T    <15><12>
      014602  040504  040524  020040         .ASCIZ    DATA    DATA    $ERRPC #ERR/TST# $ICNT #
      014653     104  053105  031440  EM32:  .ASCIZ    DEV 3 DID DATI BUT HAS INCORRECT VALUES IN DATA REG
      014737     104  053105  032040  EM33:  .ASCIZ    DEV 4 DID NOT INTR THE CORRECT # OF TIMES
      015011     114  051501  020124  EM34:  .ASCII    LAST DATIP TO DEVICE 1 DB WAS INCORRECT- EITHER DEVICE DID <15><12>
      015105     116  052117  053440         .ASCIZ    NOT WORK OR BUFFER AREA WAS NOT SET UP PROPERLY
      015165     015  041412  052520  EM35:  .ASCIZ    <15><12> CPU TRAPPED THRU LOC 0 TO CATCH IMPROPERLY LOADED VECTORS
              015262                          .EVEN
      015262  001164  001162  001102  DT1:   .WORD     $REG2,$ERRPC,$TSTNM,0
      015272  001160  001162  001164  DT2:   .WORD     $REG0,$REG1,$REG2,$REG3,$REG4,$ERRPC,$TSTNM,0
      015312  001160  001162  001164  DT3:   .WORD     $REG0,$REG1,$REG2,$REG3,$ERRPC,$TSTNM,0
      015330  001162  001164  001166  DT10:  .WORD     $REG1,$REG2,$REG3,$REG4,$ERRPC,$TSTNM,0
      015346  001162  001164  001166  DT11:  .WORD     $REG1,$REG2,$REG3,$ERRPC,$TSTNM,0
      015362  001160  001162  001116  DT12:  .WORD     $REG0,$REG1,$ERRPC,$TSTNM,0
      015374  001116  001102  000000  DT13:  .WORD     $ERRPC,$TSTNM,0
      015402  001160  001162  001166  DT15:  .WORD     $REG0,$REG1,$REG3,$ERRPC,$TSTNM,$ICNT,0
      015420  001204  001172  001116  DT22:  .WORD     $TMP4,$REG5,$ERRPC,$TSTNM,0
      015432  001162  001164  001116  DT31:  .WORD     $REG1,$REG2,$ERRPC,$TSTNM,$ICNT,0
(2)                                          ;*****************************************************************
1789                                         ;*****************************************************************
1790                                         ;*****************************************************************
1791                                         ;*****************************************************************
1792
1793                                         .SBTTL  END OF PASS ROUTINE
1794
1795                                         ;*INCREMENT THE PASS NUMBER ($PASS)
1796                                         ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
1797                                         ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
1798                                         ;*IF THERES A MONITOR GO TO IT
1799                                         ;*IF THERE ISN'T JUMP TO TST1
1800
1801  015446                         $EOP:
1802  015446  000004                         SCOPE
1803  015450  005037  001102                 CLR    $TSTNM          ;;ZERO THE TEST NUMBER
1804  015454  005037  001210                 CLR    $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
1805  015460  005237  001100                 INC    $PASS           ;;INCREMENT THE PASS NUMBER
1806  015464  042737  100000  001100         BIC    #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
1807  015472  005327                         DEC    (PC)+           ;;LOOP?
1808  015474  000001                 $EOPCT: .WORD  1
1809  015476  003063                         BGT    $DOAGN          ;;YES
1810  015500  012737                         MOV    (PC)+,a(PC)+    ;;RESTORE COUNTER
1811  015502  000001                 $ENDCT: .WORD  1
1812  015504  015474                         $EOPCT
1813  015506  104400  015514                 TYPE   .,.+4           ;;TYPE ASCIZ STRING
1814  015512  000407                         BR     64$             ;;GET OVER THE ASCIZ
1815                                         ;;.ASCIZ    <12><15>/END PASS #/
1816  015532                         64$:
1817  015532  013746  001100                 MOV    $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
1818                                                                 ;;TYPE PASS NUMBER
1819  015536  104404                         TYPDS                  ;;GO TYPE--DECIMAL ASCII WITH SIGN
1820  015540  104400  015546                 TYPE   .,.+4           ;;TYPE ASCIZ STRING
```

```
1821  015544  000421                      BR      65$           ;;GET OVER THE ASCIZ
1822                              ;;.ASCIZ     / TOTAL ERRORS SINCE LAST REPORT /
1823  015610                      65$:
1824  015610  013746  001112              MOV     $ERTTL,-(SP)  ;;SAVE $ERTTL FOR TYPEOUT
1825                                                             ;;TOTAL NUMBER OF ERRORS
1826  015614  104404                       TYPDS                ;;GO TYPE--DECIMAL ASCII WITH SIGN
1827  015616  104400  001152               TYPE    ,$NULL       ;;TYPE NULL CHAR
1828  015622  005037  001112               CLR     $ERTTL       ;;CLEAR ERROR TOTAL
1829  015626                      $GET42:
1830
1831  015626  013700  000042              MOV     @#42,R0       ;;GET MONITOR ADDRESS
1832  015632  001405                      BEQ     $DOAGN        ;;BRANCH IF NO MONITOR
1833  015634  000005                      RESET                 ;;CLEAR THE WORLD
1834  015636  004710              $ENDAD: JSR     PC,(R0)       ;;GO TO MONITOR
1835  015640  000240                      NOP                   ;;SAVE ROOM
1836  015642  000240                      NOP                   ;;FOR
1837  015644  000240                      NOP                   ;;ACT11
1838  015646                      $DOAGN:
1839  015646  000137  003506              JMP     @#TST1        ;;RETURN
1840  015652     377     377  000 $ENULL: .BYTE   -1,-1,0       ;;NULL CHARACTER STRING
1841          015656                       .EVEN
1842                              ;****************************************************************
1843
1844                              .SBTTL  SCOPE HANDLER ROUTINE
1845
1846                              ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
1847                              ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
1848                              ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
1849                              ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1850                              ;*SW14=1          LOOP ON TEST
1851                              ;*SW11=1          INHIBIT ITERATIONS
1852                              ;*SW09=1          LOOP ON ERROR
1853                              ;*SW08=1          LOOP ON TEST IN SWR<5:0>
1854                              ;*CALL
1855                              ;*      SCOPE              ;;SCOPE=IOT
1856
1857  015656                      $SCOPE:
1858  015656  032777  040000  163252 1$:   BIT     #BIT14,@SWR   ;;LOOP ON PRESENT TEST?
1859  015664  001114                      BNE     $OVER         ;;YES IF SW14=1
1860                              ;#####START OF CODE FOR THE XOR TESTER#####
1861  015666  000416              $XTSTR: BR      6$            ;;IF RUNNING ON THE "XOR" TESTER CHANGE
1862                                                             ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
1863  015670  013746  000004              MOV     @#ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
1864  015674  012737  015714  000004      MOV     #5$,@#ERRVEC  ;;SET FOR TIMEOUT
1865  015702  005737  177060              TST     @#177060      ;;TIME OUT ON XOR?
1866  015706  012637  000004              MOV     (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
1867  015712  000466                      BR      $SVLAD        ;;GO TO THE NEXT TEST
1868  015714  022626              5$:      CMP     (SP)+,(SP)+   ;;CLEAR THE STACK AFTER A TIME OUT
1869  015716  012637  000004              MOV     (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
1870  015722  000426                      BR      7$            ;;LOOP ON THE PRESENT TEST
1871  015724                      6$:;#####END OF CODE FOR THE XOR TESTER#####
1872  015724  032777  000400  163204      BIT     #BIT08,@SWR   ;;LOOP ON SPEC. TEST?
1873  015732  001407                      BEQ     2$            ;;BR IF NO
1874  015734  017746  163176              MOV     @SWR,-(SP)    ;;SET DESIRED TEST NUM. FROM SWR
1875  015740  042716  000300              BIC     #$SWRMK,(SP)  ;;STRIP AWAY UNDESIRED BITS
1876  015744  122637  001102              CMPB    (SP)+,$TSTNM  ;;ON THE RIGHT TEST?
```

```
1877  015750  001462                          BEQ     $OVER           ;;BR IF YES
1878  015752  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
1879  015756  001421                          BEQ     3$              ;;BR IF NO
1880  015760  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
1881  015766  101015                          BHI     3$              ;;BR IF NO
1882  015770  032777  001000  163140          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
1883  015776  001404                          BEQ     4$              ;;BR IF NO
1884  016000  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
1885  016006  000443                          BR      $OVER
1886  016010  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
1887  016014  005037  001210                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
1888  016020  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
1889  016022  032777  004000  163106  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
1890  016030  001011                          BNE     1$              ;;BR IF YES
1891  016032  005737  001100                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
1892  016036  001406                          BEQ     1$              ;;    INHIBIT ITERATIONS
1893  016040  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
1894  016044  023737  001210  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
1895  016052  002021                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
1896  016054  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
1897  016062  013737  016132  001210          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
1898  016070  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
1899  016074  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
1900  016100  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
1901  016104  005037  001212                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
1902  016110  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
1903  016116  013777  001102  163014  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
1904  016124  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
1905  016130  000002                          RTI                     ;;FIXES PS
1906  016132  000040                  $MXCNT: 40                      ;;MAX. NUMBER OF ITERATIONS
1907                                  ;****************************************************************
1908
1909                                  .SBTTL  ERROR HANDLER ROUTINE
1910
1911                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
1912                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
1913                                  ;*AND GO TO $ERRTYP ON ERROR
1914                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1915                                  ;*SW15=1         HALT ON ERROR
1916                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
1917                                  ;*SW10=1         BELL ON ERROR
1918                                  ;*SW09=1         LOOP ON ERROR
1919                                  ;*CALL
1920                                  ;*      ERROR   N         ;;ERROR=EMT AND N=ERROR ITEM NUMBER
1921
1922  016134                          $ERROR:
1923  016134  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
1924  016140  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
1925  016142  013777  001102  162770          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
1926  016150  032777  002000  162760          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
1927  016156  001402                          BEQ     1$              ;;NO - SKIP
1928  016160  104400  001214                  TYPE    ,$BELL          ;;RING BELL
1929  016164  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
1930  016170  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
1931  016174  162737  000002  001116          SUB     #2,$ERRPC
1932  016202  117737  162710  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
```

```
1933  016210  032777  020000  162720        BIT     #BIT13,@SWR      ;;SKIP TYPEOUT IF SET
1934  016216  001004                         BNE     20$              ;;SKIP TYPEOUTS
1935  016220  004737  016302                 JSR     PC,$ERRTYP       ;;GO TO USER ERROR ROUTINE
1936  016224  104400  001221                 TYPE    ,$CRLF
1937  016230                          20$:
1938  016230  005777  162702          2$:    TST     @SWR             ;;HALT ON ERROR
1939  016234  100001                         BPL     6$               ;;SKIP IF CONTINUE
1940  016236  000000                         HALT                     ;;HALT ON ERROR!
1941  016240  022737  015636  000042  6$:    CMP     #$FNDAD,@#42     ;;ACT-11 AUTO-ACCEPT?
1942  016246  001001                         BNE     3$               ;;BRANCH IF NO
1943  016250  000000                         HALT                     ;;YES
1944  016252  032777  001000  162656  3$:    BIT     #BIT09,@SWR      ;;LOOP ON ERROR SWITCH SET?
1945  016260  001402                         BEQ     4$               ;;BR IF NO
1946  016262  013716  001110                 MOV     $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
1947  016266  005737  001212          4$:    TST     $ESCAPE          ;;CHECK FOR AN ESCAPE ADDRESS
1948  016272  001402                         BEQ     5$               ;;BR IF NONE
1949  016274  013716  001212                 MOV     $ESCAPE,(SP)     ;;FUDGE RETURN ADDRESS FOR ESCAPE
1950  016300                          5$:
1951  016300  000002                         RTI                      ;;RETURN
1952                                   ;**************************************************************
1953
1954                                   .SBTTL   ERROR MESSAGE TYPEOUT ROUTINE
1955
1956                                   ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
1957                                   ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
1958                                   ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
1959
1960  016302                          $ERRTYP:
1961  016302  104400  001221                 TYPE    ,$CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
1962  016306  010046                         MOV     R0,-(SP)         ;;SAVE R0
1963  016310  005000                         CLR     R0               ;;PICKUP THE ITEM INDEX
1964  016312  153700  001114                 BISB    @#$ITEMB,R0
1965  016316  001004                         BNE     1$               ;;IF ITEM NUMBER IS ZERO, JUST
1966                                                                   ;;TYPE THE PC OF THE ERROR
1967  016320  013746  001116                 MOV     $ERRPC,-(SP)     ;;SAVE $ERRPC FOR TYPEOUT
1968                                                                   ;;ERROR ADDRESS
1969  016324  104401                         TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1970  016326  000426                         BR      6$               ;;GET OUT
1971  016330  005300                  1$:    DEC     R0               ;;ADJUST THE INDEX SO THAT IT WILL
1972  016332  006300                         ASL     R0               ;;       WORK FOR THE ERROR TABLE
1973  016334  006300                         ASL     R0
1974  016336  006300                         ASL     R0
1975  016340  062700  001224                 ADD     #$ERRTB,R0       ;;FORM TABLE POINTER
1976  016344  012037  016354                 MOV     (R0)+,2$         ;;PICKUP "ERROR MESSAGE" POINTER
1977  016350  001404                         BEQ     3$               ;;SKIP TYPEOUT IF NO POINTER
1978  016352  104400                         TYPE                     ;;TYPE THE "ERROR MESSAGE"
1979  016354  000000                  2$:    .WORD   0                ;;"ERROR MESSAGE" POINTER GOES HERE
1980  016356  104400  001221                 TYPE    ,$CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
1981  016362  012037  016372          3$:    MOV     (R0)+,4$         ;;PICKUP "DATA HEADER" POINTER
1982  016366  001404                         BEQ     5$               ;;SKIP TYPEOUT IF 0
1983  016370  104400                         TYPE                     ;;TYPE THE "DATA HEADER"
1984  016372  000000                  4$:    .WORD   0                ;;"DATA HEADER" POINTER GOES HERE
1985  016374  104400  001221                 TYPE    ,$CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
1986  016400  011000                  5$:    MOV     (R0),R0          ;;PICKUP "DATA TABLE" POINTER
1987  016402  001004                         BNE     7$               ;;GO TYPE THE DATA
1988  016404  012600                  6$:    MOV     (SP)+,R0         ;;RESTORE R0
```

J 4

UNIBUS EXERCISER   MACY11 30A(1052) 03-MAY-78 14:28 PAGE 43
CZKUAD.P11   26-APR-78 17:02   ERROR MESSAGE TYPEOUT ROUTINE   SEQ 0048

```
1989  016406  104400  001221              TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
1990  016412  000207                       RTS     PC              ;;RETURN
1991  016414                       7$:
1992  016414  013046                        MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
1993  016416  104401                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1994  016420  005710                        TST     (R0)            ;;IS THERE ANOTHER NUMBER?
1995  016422  001770                        BEQ     6$              ;;BR IF NO
1996  016424  104400  016432                TYPE    ,8$             ;;TYPE TWO(2) SPACES
1997  016430  000771                        BR      7$              ;;LOOP
1998  016432  020040  0C0        8$:        .ASCIZ  / /             ;;TWO(2) SPACES
1999          016436                         .EVEN
2000                              ;**************************************************************
2001
2002                              .SBTTL   TTY INPUT ROUTINE
2003
2004                              ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2005                              ;*CALL:
2006                              ;*       RDCHR                    ;;INPUT A SINGLE CHARACTER FROM THE TTY
2007                              ;*       RETURN HERE              ;;CHARACTER IS ON THE STACK
2008                              ;
2009
2010  016436  011646             $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
2011  016440  016666  000004  000002       MOV     4(SP),2(SP)     ;;SAVE THE PS
2012  016446  105777  162470     1$:        TSTB    @$TKS           ;;WAIT FOR
2013  016452  100375                        BPL     1$              ;;A CHARACTER
2014  016454  117766  162464  000004        MOVB    @$TKB,4(SP)     ;;READ THE TTY
2015  016462  042766  177600  000004        BIC     # C<177>,4(SP) ;;GET RID OF JUNK IF ANY
2016  016470  000002                        RTI                     ;;GO BACK TC USER
2017                              ;***************************************************************
2018                              ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2019                              ;*CALL:
2020                              ;*       RDLIN                    ;;INPUT A STRING FROM THE TTY
2021                              ;*       RETURN HERE              ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2022                              ;*                                ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2023
2024  016472  010346             $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
2025  016474  012703  016600     1$:        MOV     #$TTYIN,R3      ;;GET ADDRESS
2026  016500  022703  016616     2$:        CMP     #$TTYIN+16,R3   ;;BUFFER FULL?
2027  016504  101405                        BLOS    4$              ;;BR IF YES
2028  016506  104405                        RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2029  016510  112613                        MOVB    (SP)+,(R3)      ;;GET CHARACTER
2030  016512  122713  000177                CMPB    #177,(R3)       ;;IS IT A RUBOUT
2031  016516  001003                        BNE     3$              ;;SKIP IF NOT
2032  016520  104400  001220     4$:        TYPE    ,$QUES          ;;TYPE A '?'
2033  016524  000763                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
2034  016526  111337  016576     3$:        MOVB    (R3),9$         ;;ECHO THE CHARACTER
2035  016532  104400  016576                TYPE    ,9$
2036  016536  122723  000015                CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2037  016542  001356                        BNE     2$              ;;LOOP IF NOT RETURN
2038  016544  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2039  016550  104400  001222                TYPE    ,$LF            ;;TYPE A LINE FEED
2040  016554  012603                        MOV     (SP)+,R3        ;;RESTORE R3
2041  016556  011646                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2042  016560  016666  000004  000002        MOV     4(SP),2(SP)     ;;     FIRST ASCII CHARACTER ON IT
2043  016566  012766  016600  000004        MOV     #$TTYIN,4(SP)
2044  016574  000002                        RTI                     ;;RETURN
```

```
2045  016576    000                    9$:     .BYTE   0                   ;;STORAGE FOR ASCII CHAR. TO TYPE
2046  016577    000                            .BYTE   0                   ;;TERMINATOR
2047  016600    000016                 $TTYIN: .BLKB   16                  ;;RESERVE 16 BYTES FOR TTY INPUT
2048                                    ;**********************************************************************
2049
2050                                    .SBTTL  ROUTINE TO SIZE MEMORY
2051
2052                                    ;*CALL:
2053                                    ;*      JSR     PC,$SIZE
2054                                    ;*      RETURN
2055                                    ;*$LSTAD WILL CONTAIN:
2056                                    ;*         WITH KT11 OPTION      -- LAST VIRTUAL ADDRESS OF THE LAST BANK
2057                                    ;*         WITHOUT KT11 OPTION   -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
2058                                    ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
2059                                    ;*$KT11 IS THE MEMORY MANAGEMENT KEY
2060                                    ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
2061                                    ;*         MUST BE SETUP BEFORE THE CALL
2062                                    ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
2063                                    ;*         DETERMINED BY ROUTINE
2064
2065  016616    010046                 $SIZE:  MOV     R0,-(SP)            ;;SAVE R0 ON THE STACK
2066  016620    010146                         MOV     R1,-(SP)            ;;SAVE R1 ON THE STACK
2067  016622    010246                         MOV     R2,-(SP)            ;;SAVE R2 ON THE STACK
2068  016624    010346                         MOV     R3,-(SP)            ;;SAVE R3 ON THE STACK
2069  016626    013746    000004               MOV     @#ERRVEC,-(SP)      ;;SAVE PRESENT ERROR VECTOR PS & PC
2070  016632    013746    000006               MOV     @#ERRVEC+2,-(SP)
2071  016636    010600                         MOV     SP,R0               ;;SAVE THE STACK POINTER
2072  016640    013746    000034               MOV     @#34,-(SP)          ;;SETUP THE TRAP VECTOR
2073  016644    012737    016654    000034     MOV     #1$,@#34            ;; TO GET THE PS
2074  016652    104400                         TRAP
2075  016654    016637    000002    000006 1$: MOV     2(SP),@#ERRVEC+2 ;;SET ERRVEC PS
2076  016662    012716    016670               MOV     #$SIZE1,(SP)        ;; TO PRESENT PS
2077  016666    000002                         RTI
2078  016670    012637    000034     $SIZE1: MOV     (SP)+,@#34          ;;RESTORE TRAP VECTOR
2079  016674    012701    003776               MOV     #3776,R1            ;;SETUP ADDRESS
2080  016700    105727                         TSTB    (PC)+               ;;USE MEMORY MANAGEMENT?
2081  016702    000200                 $KT11:  .WORD   200                 ;;SET TO USE MEMORY MANAGEMENT
2082  016704    100063                         BPL     $CORE               ;;BR IF NO
2083  016706    012737    017046    000004     MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
2084  016714    005737    177572               TST     @#SR0               ;;KT11 ARE YOU THERE?
2085  016720    052737    100000    016702     BIS     #100000,$KT11       ;;YES--SET KT11 KEY
2086  016726    005046                         CLR     -(SP)               ;;INITIALIZE FOR "PAR" LOADING
2087  016730    012702    172340               MOV     #KIPAR0,R2          ;;ADDRESS OF FIRST "PAR"
2088  016734    012703    000010               MOV     # D8,R3             ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
2089  016740    012762    077406    177740 1$: MOV     #77406,-40(R2)      ;;PDR = 4K, UP, READ/WRITE
2090  016746    011622                         MOV     (SP),(R2)+          ;;LOAD "PAR"
2091  016750    062716    000200               ADD     #200,(SP)           ;;UPDATE FOR NEXT "PAR"
2092  016754    077307                         SOB     R3,1$               ;;LOOP UNTIL ALL EIGHT ARE LOADED
2093  016756    012742    177600               MOV     #177600,-(R2)       ;;SETUP KIPAR7 FOR I/O
2094  016762    005042                         CLR     -(R2)               ;;SETUP KIPAR6 FOR TESTING
2095  016764    012737    000006    000004     MOV     #6,      @#4        ;;SET FOR TIMEOUT
2096  016772    012737    000002    000006     MOV     #2,      @#6        ;;IF SR3 DOESNT EXIST
2097  017000    012737    000020    172516     MOV     #20,@#SR3           ;;ENABLE 22 BIT MODE
2098  017006    005237    177572               INC     @#SR0               ;;TURN ON MEMORY MANAGEMENT
2099  017012    012737    017036    000004     MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
2100  017020    005737    143776     2$:       TST     @#143776            ;;TRAP ON NON-EX-MEM
```

```
2101  017024  062712  000040                  ADD     #40,(R2)          ;;MAKE A 1K STEP
2102  017030  023712  172356                  CMP     @#KIPAR7,(R2)     ;;LAST ONE?
2103  017034  101371                          BHI     2$                ;;NO--TRY IT
2104  017036  011202          $KTOUT: MOV     (R2),R2           ;;GET LAST BANK+1
2105  017040  005037  177572                  CLR     @#SR0             ;;TURN OFF MEMORY MANAGEMENT
2106  017044  000421                          BR      $SIZEX
2107  017046  042737  100000  016702  $KTNEX: BIC     #100000,$KT11     ;;KT11 NON-EXISTENT
2108  017054  012737  017104  000004  $CORE:  MOV     #$CROUT,@#ERRVEC  ;;SET FOR TIMEOUT
2109  017062  005002                          CLR     R2                ;;SET UP BANK
2110  017064  062701  004000          1$:     ADD     #4000,R1          ;;INCREMENT BY 1K
2111  017070  062702  000040                  ADD     #40,R2            ;;1K STEP
2112  017074  005711                          TST     (R1)              ;;TRAP ON TIME OUT
2113  017076  022701  177776                  CMP     #177776,R1        ;;LAST ONE
2114  017102  001370                          BNE     1$                ;;NO--TRY AGAIN
2115  017104  162701  004000          $CROUT: SUB     #4000,R1
2116  017110  162702  000040          $SIZEX: SUB     #40,R2            ;;DROP BACK
2117  017114  010006                          MOV     R0,SP             ;;RESTORE THE STACK
2118  017116  012637  000006                  MOV     (SP)+,@#ERRVEC+2  ;;RESTORE ERROR VECTOR
2119  017122  012637  000004                  MOV     (SP)+,@#ERRVEC
2120  017126  010137  017150                  MOV     R1,$LSTAD         ;;LAST ADDRESS
2121  017132  010237  017152                  MOV     R2,$LSTBK         ;;LAST BANK
2122  017136  012603                          MOV     (SP)+,R3          ;;RESTORE R3
2123  017140  012602                          MOV     (SP)+,R2          ;;RESTORE R2
2124  017142  012601                          MOV     (SP)+,R1          ;;RESTORE R1
2125  017144  012600                          MOV     (SP)+,R0          ;;RESTORE R0
2126  017146  000207                          RTS     PC
2127  017150  000000          $LSTAD: .WORD   0                 ;;CONTAINS THE LAST ADDRESS
2128  017152  000000          $LSTBK: .WORD   0                 ;;CONTAINS THE LAST BANK
2129                                  ;**************************************************************
2130
2131                                  .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
2132
2133                                  ;*SAVE R0-R5
2134                                  ;*CALL:
2135                                  ;*      SAVREG
2136                                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
2137                                  ;*
2138                                  ;*TOP---(+16)
2139                                  ;* +2---(+18)
2140                                  ;* +4---R5
2141                                  ;* +6---R4
2142                                  ;* +8---R3
2143                                  ;*+10---R2
2144                                  ;*+12---R1
2145                                  ;*+14---R0
2146
2147  017154                  $SAVREG:
2148  017154  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
2149  017156  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
2150  017160  010246                          MOV     R2,-(SP)          ;;PUSH R2 ON STACK
2151  017162  010346                          MOV     R3,-(SP)          ;;PUSH R3 ON STACK
2152  017164  010446                          MOV     R4,-(SP)          ;;PUSH R4 ON STACK
2153  017166  010546                          MOV     R5,-(SP)          ;;PUSH R5 ON STACK
2154  017170  016646  000022                  MOV     22(SP),-(SP)      ;;SAVE PS OF MAIN FLOW
2155  017174  016646  000022                  MOV     22(SP),-(SP)      ;;SAVE PC OF MAIN FLOW
2156  017200  016646  000022                  MOV     22(SP),-(SP)      ;;SAVE PS OF CALL
```

```
2157  017204  016646  000022              MOV      22(SP),-(SP)     ;;SAVE PC OF CALL
2158  017210  000002                      RTI
2159
2160                            ;*RESTORE R0-R5
2161                            ;*CALL:
2162                            ;*      RESREG
2163  017212                    $RESREG:
2164  017212  012666  000022             MOV      (SP)+,22(SP)     ;;RESTORE PC OF CALL
2165  017216  012666  000022             MOV      (SP)+,22(SP)     ;;RESTORE PS OF CALL
2166  017222  012666  000022             MOV      (SP)+,22(SP)     ;;RESTORE PC OF MAIN FLOW
2167  017226  012666  000022             MOV      (SP)+,22(SP)     ;;RESTORE PS OF MAIN FLOW
2168  017232  012605                     MOV      (SP)+,R5         ;;POP STACK INTO R5
2169  017234  012604                     MOV      (SP)+,R4         ;;POP STACK INTO R4
2170  017236  012603                     MOV      (SP)+,R3         ;;POP STACK INTO R3
2171  017240  012602                     MOV      (SP)+,R2         ;;POP STACK INTO R2
2172  017242  012601                     MOV      (SP)+,R1         ;;POP STACK INTO R1
2173  017244  012600                     MOV      (SP)+,R0         ;;POP STACK INTO R0
2174  017246  000002                     RTI
2175                            ;*****************************************************************
2176
2177                            .SBTTL   TYPE ROUTINE
2178
2179                            ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2180                            ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2181                            ;*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2182                            ;*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2183                            ;*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2184                            ;*
2185                            ;*CALL:
2186                            ;*1) USING A TRAP INSTRUCTION
2187                            ;*      TYPE     ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2188                            ;*OR
2189                            ;*      TYPE
2190                            ;*      MESADR
2191                            ;*
2192
2193  017250  105737  001155    $TYPE:    TSTB     $TPFLG           ;;IS THERE A TERMINAL?
2194  017254  100002                      BPL      1$               ;;BR IF YES
2195  017256  000000                      HALT                      ;;HALT HERE IF NO TERMINAL
2196  017260  000407                      BR       3$               ;;LEAVE
2197  017262  010046            1$:       MOV      R0,-(SP)         ;;SAVE R0
2198  017264  017600  000002             MOV      @2(SP),R0        ;;GET ADDRESS OF ASCIZ STRING
2199  017270  112046            2$:       MOVB     (R0)+,-(SP)      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
2200  017272  001005                      BNE      4$               ;;BR IF IT ISN'T THE TERMINATOR
2201  017274  005726                      TST      (SP)+            ;;IF TERMINATOR POP IT OFF THE STACK
2202  017276  012600            60$:      MOV      (SP)+,R0         ;;RESTORE R0
2203  017300  062716  000002    3$:       ADD      #2,(SP)          ;;ADJUST RETURN PC
2204  017304  000002                      RTI                       ;;RETURN
2205  017306  122716  000011    4$:       CMPB     #THT,(SP)        ;;BRANCH IF <HT>
2206  017312  001426                      BEQ      8$
2207  017314  122716  000200             CMPB     #TCRLF,(SP)      ;;BRANCH IF NOT <CRLF>
2208  017320  001004                      BNE      5$
2209  017322  005726                      TST      (SP)+            ;;POP  <CR><LF> EQUIV
2210  017324  104400                      TYPE                      ;;TYPE A CR AND LF
2211  017326  001221                      $CRLF
2212  017330  000757                      BR       2$               ;;GET NEXT CHARACTER
```

```
2213   017332   004737   017414          5$:     JSR     PC,$TYPEC      ;;GO TYPE THIS CHARACTER
2214   017336   123726   001154          6$:     CMPB    $FILLC,(SP)+   ;;IS IT TIME FOR FILLER CHARS.?
2215   017342   001352                           BNE     2$             ;;IF NO GO GET NEXT CHAR.
2216   017344   013746   001152                  MOV     $NULL,-(SP)    ;;GET # OF FILLER CHARS. NEEDED
2217                                                                    ;;AND THE NULL CHAR.
2218   017350   105366   000001          7$:     DECB    1(SP)          ;;DOES A NULL NEED TO BE TYPED?
2219   017354   002770                           BLT     6$             ;;BR IF NO--GO POP THE NULL OFF OF STACK
2220   017356   004737   017414                  JSR     PC,$TYPEC      ;;GO TYPE A NULL
2221   017362   105337   017460                  DECB    $CHARCNT       ;;DO NOT COUNT AS A COUNT
2222   017366   000770                           BR      7$             ;;LOOP
2223
2224                                      ;HORIZONTAL TAB PROCESSOR
2225
2226   017370   112716   000040          8$:     MOVB    #40,(SP)       ;;REPLACE TAB WITH SPACE
2227   017374   004737   017414          9$:     JSR     PC,$TYPEC      ;;TYPE A SPACE
2228   017400   132737   000007   017460         BITB    #7,$CHARCNT    ;;BRANCH IF NOT AT
2229   017406   001372                           BNE     9$             ;;TAB STOP
2230   017410   005726                           TST     (SP)+          ;;POP SPACE OFF STACK
2231   017412   000726                           BR      2$             ;;GET NEXT CHARACTER
2232   017414   105777   161526          $TYPEC: TSTB    @$TPS          ;;WAIT UNTIL PRINTER IS READY
2233   017420   100375                           BPL     $TYPEC
2234   017422   116677   000002   161520         MOVB    2(SP),@$TPB    ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2235   017430   122766   000015   000002         CMPB    #15,2(SP)      ;;BRANCH IF
2236   017436   001003                           BNE     1$             ;;NOT <CR>
2237   017440   105037   017460                  CLRB    $CHARCNT       ;;
2238   017444   000406                           BR      $TYPEX         ;;EXIT
2239   017446   122766   000012   000002  1$:    CMPB    #12,2(SP)      ;;BRANCH IF
2240   017454   002002                           BGE     $TYPEX         ;;<LF>
2241   017456   105227                           INCB    (PC)+          ;;INC SPACE
2242   017460   000000                  $CHARCNT:.WORD   0              ;;COUNT
2243   017462   000207                  $TYPEX: RTS     PC
2244                                             ;;      EQUATES
2245            000011                   THT=11
2246            000200                   TCRLF=200
2247
2248                                     ;***********************************************************************
2249
2250                                     .SBTTL   BINARY TO OCTAL (ASCII) AND TYPE
2251
2252                                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2253                                     ;*OCTAL (ASCII) NUMBER AND TYPE IT.
2254                                     ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2255                                     ;*CALL:
2256                                     ;*      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
2257                                     ;*      TYPOS                  ;;CALL FOR TYPEOUT
2258                                     ;*      .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2259                                     ;*      .BYTE   M              ;;M=1 OR 0
2260                                     ;*                                     ;;1=TYPE LEADING ZEROS
2261                                     ;*                                     ;;0=SUPPRESS LEADING ZEROS
2262                                     ;*
2263                                     ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2264                                     ;*$TYPOS OR $TYPOC
2265                                     ;*CALL:
2266                                     ;*      MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
2267                                     ;*      TYPON                  ;;CALL FOR TYPEOUT
2268                                     ;*
```

```
2269                                    ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
2270                                    ;*CALL:
2271                                    ;*       MOV       NUM,-(SP)       ;;NUMBER TO BE TYPED
2272                                    ;*       TYPOC                     ;;CALL FOR TYPEOUT
2273
2274   017464   017646   000000   $TYPOS:  MOV    a(SP),-(SP)     ;;PICKUP THE MODE
2275   017470   116637   000001   017707    MOVB  1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
2276   017476   112637   017711             MOVB  (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
2277   017502   062716   000002             ADD   #2,(SP)         ;;ADJUST RETURN ADDRESS
2278   017506   000406                       BR   $TYPON
2279   017510   112737   000001   017707   $TYPOC:  MOVB  #1,$OFILL      ;;SET THE ZERO FILL SWITCH
2280   017516   112737   000006   017711    MOVB  #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
2281   017524   112737   000005   017706   $TYPON:  MOVB  #5,$OCNT      ;;SET THE ITERATION COUNT
2282   017532   010346                       MOV  R3,-(SP)       ;;SAVE R3
2283   017534   010446                       MOV  R4,-(SP)       ;;SAVE R4
2284   017536   010546                       MOV  R5,-(SP)       ;;SAVE R5
2285   017540   113704   017711             MOVB  $OMODE+1,R4    ;;GET THE NUMBER OF DIGITS TO TYPE
2286   017544   005404                       NEG  R4
2287   017546   062704   000006             ADD   #6,R4          ;;SUBTRACT IT FOR MAX. ALLOWED
2288   017552   110437   017710             MOVB  R4,$OMODE       ;;SAVE IT FOR USE
2289   017556   113704   017707             MOVB  $OFILL,R4      ;;GET THE ZERO FILL SWITCH
2290   017562   016605   000012             MOV   12(SP),R5      ;;PICKUP THE INPUT NUMBER
2291   017566   005003                       CLR  R3            ;;CLEAR THE OUTPUT WORD
2292   017570   006105             1$:       ROL  R5            ;;ROTATE MSB INTO "C"
2293   017572   000404                        BR  3$            ;;GO DO MSB
2294   017574   006105             2$:       ROL  R5            ;;FORM THIS DIGIT
2295   017576   006105                       ROL  R5
2296   017600   006105                       ROL  R5
2297   017602   010503                       MOV  R5,R3
2298   017604   006103             3$:       ROL  R3            ;;GET LSB OF THIS DIGIT
2299   017606   105337   017710             DECB  $OMODE         ;;TYPE THIS DIGIT?
2300   017612   100016                       BPL  7$            ;;BR IF NO
2301   017614   042703   177770             BIC   #177770,R3     ;;GET RID OF JUNK
2302   017620   001002                       BNE  4$            ;;TEST FOR 0
2303   017622   005704                       TST  R4            ;;SUPPRESS THIS 0?
2304   017624   001403                       BEQ  5$            ;;BR IF YES
2305   017626   005204             4$:       INC  R4            ;;DON'T SUPPRESS ANYMORE 0'S
2306   017630   052703   000060             BIS   #'0,R3         ;;MAKE THIS DIGIT ASCII
2307   017634   052703   000040   5$:       BIS   #' ,R3         ;;MAKE ASCII IF NOT ALREADY
2308   017640   110337   017704             MOVB  R3,8$          ;;SAVE FOR TYPING
2309   017644   104400   017704             TYPE  .8$            ;;GO TYPE THIS DIGIT
2310   017650   105337   017706   7$:       DECB  $OCNT          ;;COUNT BY 1
2311   017654   003347                       BGT  2$            ;;BR IF MORE TO DO
2312   017656   002402                       BLT  6$            ;;BR IF DONE
2313   017660   005204                       INC  R4            ;;INSURE LAST DIGIT ISN'T A BLANK
2314   017662   000744                        BR  2$            ;;GO DO THE LAST DIGIT
2315   017664   012605             6$:       MOV  (SP)+,R5       ;;RESTORE R5
2316   017666   012604                       MOV  (SP)+,R4       ;;RESTORE R4
2317   017670   012603                       MOV  (SP)+,R3       ;;RESTORE R3
2318   017672   016666   000002   000004    MOV   2(SP),4(SP)    ;;SET THE STACK FOR RETURNING
2319   017700   012616                       MOV  (SP)+,(SP)
2320   017702   000002                       RTI                 ;;RETURN
2321   017704   000               8$:       .BYTE  0            ;;STORAGE FOR ASCII DIGIT
2322   017705   000                         .BYTE  0            ;;TERMINATOR FOR TYPE ROUTINE
2323   017706   000               $OCNT:    .BYTE  0            ;;OCTAL DIGIT COUNTER
2324   017707   000               $OFILL:   .BYTE  0            ;;ZERO FILL SWITCH
```

```
2325   017710  000000                    $OMODE: .WORD   0                      ;;NUMBER OF DIGITS TO TYPE
2326                                      ;***********************************************************
2327
2328                                      .SBTTL   CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
2329
2330                                      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
2331                                      ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
2332                                      ; NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
2333                                      ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
2334                                      ;*R PLACED WITH SPACES.
2335                                      ;*CALL:
2336                                      ;*       MOV     NUM,-(SP)              ;;PUT THE BINARY NUMBER ON THE STACK
2337                                      ;*       TYPDS                          ;;GO TO THE ROUTINE
2338
2339   017712                            $TYPDS:
2340   017712  010046                             MOV     R0,-(SP)               ;;PUSH R0 ON STACK
2341   017714  010146                             MOV     R1,-(SP)               ;;PUSH R1 ON STACK
2342   017716  010246                             MOV     R2,-(SP)               ;;PUSH R2 ON STACK
2343   017720  010346                             MOV     R3,-(SP)               ;;PUSH R3 ON STACK
2344   017722  010546                             MOV     R5,-(SP)               ;;PUSH R5 ON STACK
2345   017724  012746  020200               MOV     #20200,-(SP)           ;;SET BLANK SWITCH AND SIGN
2346   017730  016605  000020               MOV     20(SP),R5              ;;GET THE INPUT NUMBER
2347   017734  100004                             BPL     1$                     ;;BR IF INPUT IS POS.
2348   017736  005405                             NEG     R5                     ;;MAKE THE BINARY NUMBER POS.
2349   017740  112766  000055  000001       MOVB    #'-,1(SP)              ;;MAKE THE ASCII NUMBER NEG.
2350   017746  005000                    1$:      CLR     R0                     ;;ZERO THE CONSTANTS INDEX
2351   017750  012703  020126               MOV     #$DBLK,R3              ;;SETUP THE OUTPUT POINTER
2352   017754  112723  000040               MOVB    #' ,(R3)+              ;;SET THE FIRST CHARACTER TO A BLANK
2353   017760  005002                    2$:      CLR     R2                     ;;CLEAR THE BCD NUMBER
2354   017762  016001  020116               MOV     $DTBL(R0),R1          ;;GET THE CONSTANT
2355   017766  160105                    3$:      SUB     R1,R5                  ;;FORM THIS BCD DIGIT
2356   017770  002402                             BLT     4$                     ;;BR IF DONE
2357   017772  005202                             INC     R2                     ;;INCREASE THE BCD DIGIT BY 1
2358   017774  000774                             BR      3$
2359   017776  060105                    4$:      ADD     R1,R5                  ;;ADD BACK THE CONSTANT
2360   020000  005702                             TST     R2                     ;;CHECK IF BCD DIGIT=0
2361   020002  001002                             BNE     5$                     ;;FALL THROUGH IF 0
2362   020004  105716                             TSTB    (SP)                   ;;STILL DOING LEADING 0'S?
2363   020006  100407                             BMI     7$                     ;;BR IF YES
2364   020010  106316                    5$:      ASLB    (SP)                   ;;MSD?
2365   020012  103003                             BCC     6$                     ;;BR IF NO
2366   020014  116663  000001  177777       MOVB    1(SP),-1(R3)          ;;YES--SET THE SIGN
2367   020022  052702  000060             6$:      BIS     #'0,R2                 ;;MAKE THE BCD DIGIT ASCII
2368   020026  052702  000040             7$:      BIS     #' ,R2                 ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
2369   020032  110223                             MOVB    R2,(R3)+               ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
2370   020034  005720                             TST     (R0)+                  ;;JUST INCREMENTING
2371   020036  020027  000010               CMP     R0,#10                ;;CHECK THE TABLE INDEX
2372   020042  002746                             BLT     2$                     ;;GO DO THE NEXT DIGIT
2373   020044  003002                             BGT     8$                     ;;GO TO EXIT
2374   020046  010502                             MOV     R5,R2                  ;;GET THE LSD
2375   020050  000764                             BR      6$                     ;;GO CHANGE TO ASCII
2376   020052  105726                    8$:      TSTB    (SP)+                  ;;WAS THE LSD THE FIRST NON-ZERO?
2377   020054  100003                             BPL     9$                     ;;BR IF NO
2378   020056  116663  177777  177776       MOVB    -1(SP),-2(R3)         ;;YES--SET THE SIGN FOR TYPING
2379   020064  105013                    9$:      CLRB    (R3)                   ;;SET THE TERMINATOR
2380   020066  012605                             MOV     (SP)+,R5               ;;POP STACK INTO R5
```

```
2381   020070  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
2382   020072  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2383   020074  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2384   020076  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2385   020100  104400  020126                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
2386   020104  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
2387   020112  012616                          MOV     (SP)+,(SP)
2388   020114  000002                          RTI                     ;;RETURN TO USER
2389   020116  023420          $DTBL:  10000.
2390   020120  001750                  1000.
2391   020122  000144                  100.
2392   020124  000012                  10.
2393   020126  000004          $DBLK:  .BLKW    4
2394                           ;*******************************************************************
2395
2396                           .SBTTL  TRAP DECODER
2397
2398                           ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2399                           ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2400                           ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2401                           ;*GO TO THAT ROUTINE.
2402
2403   020136  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2404   020140  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
2405   020144  005740                  TST     -(R0)           ;;BACKUP BY 2
2406   020146  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2407   020150  006300                  ASL     R0              ;;POSITION FOR INDEXING
2408   020152  016000  020160          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2409   020156  000200                  RTS     R0              ;;GO TO ROUTINE
2410
2411
2412                           .SBTTL  TRAP TABLE
2413
2414                           ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2415                           ;*BY THE "TRAP" INSTRUCTION.
2416
2417                           ;       ROUTINE
2418                           ;       -------
2419   020160          $TRPAD:
2420   020160  017250          $TYPE   ;;CALL=TYPE       TRAP+0(104400)  TTY TYPEOUT ROUTINE
2421   020162  017510          $TYPOC  ;;CALL=TYPOC      TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2422   020164  017464          $TYPOS  ;;CALL=TYPOS      TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2423   020166  017524          $TYPON  ;;CALL=TYPON      TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2424   020170  017712          $TYPDS  ;;CALL=TYPDS      TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
2425   020172  016436          $RDCHR  ;;CALL=RDCHR      TRAP+5(104405)  TTY TYPEIN CHARACTER ROUTINE
2426   020174  016472          $RDLIN  ;;CALL=RDLIN      TRAP+6(104406)  TTY TYPEIN STRING ROUTINE
2427   020176  017154          $SAVREG ;;CALL=SAVREG     TRAP+7(104407)  SAVE R0-R5 ROUTINE
2428   020200  017212          $RESREG ;;CALL=RESREG     TRAP+10(104410) RESTORE R0-R5 ROUTINE
2429                           ;*******************************************************************
2430
2431                           .SBTTL  POWER DOWN AND UP ROUTINES
2432
2433                           ;POWER DOWN ROUTINE
2434   020202  012737  0<0324  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2435   020210  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2436   020216  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
```

E 5

UNIBUS EXERCISER   MACY11 30A(1052) 03-MAY-78 14:28 PAGE 51
CZKUAD.P11   26-APR-78 17:02   POWER DOWN AND UP ROUTINES   SEQ 0056

```
2437  020220  010146                              MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2438  020222  010246                              MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2439  020224  010346                              MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2440  020226  010446                              MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2441  020230  010546                              MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2442  020232  010637  020330                      MOV     SP,$SAVR6       ;;SAVE SP
2443  020236  012737  020250  000024              MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
2444  020244  000000                              HALT
2445  020246  000776                              BR      .-2             ;;HANG UP
2446
2447                                      ;POWER UP ROUTINE
2448  020250  013706  020330      $PWRUP: MOV     $SAVR6,SP               ;;GET SP
2449  020254  005037  020330              CLR     $SAVR6                  ;;WAIT LOOP FOR THE TTY
2450  020260  005237  020330      1$:     INC     $SAVR6                  ;;WAIT FOR THE INC
2451  020264  001375                      BNE     1$                      ;;OF  WORD
2452  020266  012605                      MOV     (SP)+,R5                ;;POP STACK INTO R5
2453  020270  012604                      MOV     (SP)+,R4                ;;POP STACK INTO R4
2454  020272  012603                      MOV     (SP)+,R3                ;;POP STACK INTO R3
2455  020274  012602                      MOV     (SP)+,R2                ;;POP STACK INTO R2
2456  020276  012601                      MOV     (SP)+,R1                ;;POP STACK INTO R1
2457  020300  012600                      MOV     (SP)+,R0                ;;POP STACK INTO R0
2458  020302  012737  020202  000024      MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
2459  020310  012737  000340  000026      MOV     #340,@#PWRVEC+2 ;;PRIO:7
2460  020316  104400                      TYPE                    ;REPORT THE POWER FAILURE
2461  020320  020332      $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
2462  020322  000002                      RTI
2463  020324  000000      $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
2464  020326  000776                      BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
2465  020330  000000      $SAVR6: 0                       ;;PUT THE SP HERE
2466  020332  005015  047520  042527 $POWER: .ASCIZ  <15><12>"POWER"
2467  020340  000122
2468                                      .EVEN
2469                              ;*******************************************************************
2470                              ;*******************************************************************
2471                              ;*******************************************************************
2472  020342                      ATEND:
2473          000001              .END
```

```
ADD20   002644        611    620#
ADLI    011074       1088   1090   1093   1096   1762#
ALLERR  001574        465#   954
ATEND   020342        771    804    838    872    906    935    975    982   1181   1277   1286   1295   1303
                     1324   1347   1443   1447   1458   1467   1477   1503   1537   1556   1699   1780   1782
                     2472#
BE1BA   001604        471#   771*   804*   838*   872*   906*   935*   982*  1110   1181*  1245   1286*  1412
                     1447*  1699*  1716   1766*
BE1CC   001602        470#   772*   776    779    805*   809    8'2    839*   843    846    873*   877    880
                      907*   911    914    934*   981*  1182*  1287*  1448*  1700*  1703   1715   1765*
BE1CLR  001610        473#  1105*
BE1CR1  001606        472#   747*   748*   773*   778    806*   811    840*   845    874*   879    908*   913
                      941*   947    984*   987   1017*  1108   1135*  1288*  1315   1415   1452*  1701*  1717
                     1767*
BE1CR2  001612        474#   940*   944*   945    948    954    983*   989*   996   1089*  1091*  1092*  1094*
                     1103   1109   1137*  1140*  1153*  1183*  1184   1202*  1451*  1502*  1609   1624
BE1DB   001600        469#   512    580    639    692    990    992   1051   1714
BE1PSW  001662        494#   733*   770*   803*   837*   871*   905*   939*   980*  1016*  1086*  1134*  1285*
                     1450*
BE1VEC  001660        493#   581    732*   769*   802*   836*   870*   904*   938*   979*  1015*  1085*  1133*
                     1178*  1284*  1449*
BE2BA   001620        477#  1250   1295*  1419   1458*
BE2CC   001616        476#  1294*  1459*
BE2CLR  001624        479#
BE2CR1  001622        478#  1022*  1296*  1422   1460*
BE2CR2  001626        480#
BE2DB   001614        475#  1056
BE2PSW  001666        496#  1021*  1293*  1457*
BE2VEC  001664        495#  1020*  1292*  1456*
BE3BA   001634        483#  1255   1303*  1467*
BE3CC   001632        482#  1302*  1468*
BE3CLR  001640        485#
BF3CR1  001636        484#  1027*  1304*  1366   1427   1470*
BE3CR2  001642        486#  1469*  1536*
BE3DB   001630        481#  1061   1370   1372   1465*
BE3PSW  001672        498#  1026*  1301*  1464*
BE3VEC  001670        497#  1025*  1300*  1466*
BE4BA   001650        489#  1260   1477*
BE4CC   001646        488#  1311*  1478*
BE4CLR  001654        491#
BE4CR1  001652        490#  1033*  1313*  1386   1433   1479*
BE4CR2  001656        492#  1312*  1394*
BE4DB   001644        487#  1066   1474*
BE4PSW  001676        500#  1032*  1310*  1475*
BE4VEC  001674        499#  1031*  1309*  1476*
BGIN    002736        585    612    652#
BIT0  = 000001        118#
BIT00 = 000001        108#   118   1677   1685
BIT01 = 000002        107#   117
BIT02 = 000004        106#   116
BIT03 = 000010        105#   115    940    944
BIT04 = 000020        104#   114   1137   1140   1153   1612
BIT05 = 000040        103#   113    748   1642
BIT06 = 000100        102#   112   1168   1646
BIT07 = 000200        101#   111    553    564    659
BIT08 = 000400        100#   110   1650   1872
```

```
BIT09 = 001000            99#     109    1654    1882    1944
BIT1  = 000002           117#
BIT10 = 002000            98#    1658    1926
BIT11 = 004000            97#     747     996    1638    1889
BIT12 = 010000            96#    1183
BIT13 = 020000            95#    1933
BIT14 = 040000            94#     983     989    1312    1394    1451    1469    1502    1536    1858
BIT15 = 100000            93#
BIT2  = 000004           116#
BIT3  = 000010           115#
BIT4  = 000020           114#
BIT5  = 000040           113#
BIT6  = 000100           112#
BIT7  = 000200           111#
BIT8  = 000400           110#
BIT9  = 001000           109#
BKEX    011026           1737    1740#
BKGD    010754           1316    1723#
BPTVEC= 000014            125#
BRK     005604           1085    1100#
BR4TST  004436            898#
BR5TST  004276            864#
BR6TST  004136            830#
BR7TST  003776            796#
CHEX    007466           1417    1424    1429    1435#
CHKERR  010432           1631#
CLRREG  002700            636#     655     729     933     974    1012    1083    1216    1281
CLRRTN  001724            511#     575
CMPARE  005340           1039#
CNTR    011046            943    1038    1138    1754#   1768
DATA1   001712            504#   1245*   1246*   1326    1412*   1413*   1507    1511
DATA2   001714            505#   1250*   1251*   1348    1419*   1420*
DATA3   001716            506#   1255*   1256*   1538
DATA4   001720            507#   1260*   1261*   1557
DDISP = 177570            40#     260
DEVCNT  001700            501#     590*    613     646     679     713     716    1018    1023    1028    1039    1220    1289
                         1297    1305    1338    1357    1376    1453    1461    1471    1491    1525    1527    1547    1579
                         1615
DEVS    001702            502#   1014    1041    1042    1043    1044
DH1     011452            310     459    1788#
DH10    012257            346    1788#
DH11    012503            352    1788#
DH12    012642            357    1788#
DH13    012766            362     367    1788#
DH15    013235            373     379     384     389     394    1788#
DH2     011616            316    1788#
DH22    014005            399     404     409     414     419     424     429    1788#
DH3     011740            321     326     331     336     341    1788#
DH31    014560            435     442     447     453    1788#
DISPLA  001140            260#    549*   1903*   1925*
DISPRE  000174            199#     549
DOUP    011030            977    1279    1445    1743#
DO14    005532           1087#
DSWR  = 177570            39#     259
DT1     015262            311     460    1788#
DT10    015330            348    1788#
```

```
DT11      015346        353    1788#
DT12      015362        358    1788#
DT13      015374        363     368    1788#
DT15      015402        375     380     385     390     395    1788#
DT2       015272        317    1788#
DT22      015420        400     405     410     415     420     425     430    1788#
DT3       015312        322     327     332     337     342    1788#
DT31      015432        437     443     448     454    1788#
EMTVEC=   000030        128#    531*    532*
EM1       011404        309    1788#
EM10      012211        345    1788#
EM11      012415        351    1788#
EM12      012554        356    1788#
EM13      012703        361    1788#
EM14      013007        366    1788#
EM15      013072        371    1788#
EM16      013347        378    1788#
EM17      013435        383    1788#
EM2       011503        314    1788#
EM20      013523        388    1788#
EM21      013617        393    1788#
EM22      013713        398    1788#
EM23      014047        403    1788#
EM24      014120        408    1788#
EM25      014166        413    1788#
EM26      014220        418    1788#
EM27      014262        423    1788#
EM3       011706        320    1788#
EM30      014331        428    1788#
EM31      014420        433    1788#
EM32      014653        440    1788#
EM33      014737        446    1788#
EM34      015011        451    1788#
EM35      015165        457    1788#
EM4       012021        325    1788#
EM5       012057        330    1788#
EM6       012115        335    1788#
EM7       012153        340    1788#
ENDMEM    001722        508#    514
ERRCHK    004730        732     769     802     836     870     904     938     953#    979    1133
ERRS      010714       1710    1713#
ERRTN     010406        958    1070    1112    1197    1266    1437    1602    1622#
ERRVEC=   000004        121#    577*    587*    589*    734*    937*   1863    1864*   1866*   1869*   2069    2070    2075*
                       2083*   2099*   2108*   2118*   2119*
EXAD      011140       1770    1773#
EXBRK     005664       1106    1113#
EXNO      002660        614     623#
EXNOG     010712       1708    1711#
EXS       007476       1416    1423    1428    1434    1438#
FAILCK    006004       1141#
FOR4      011277        718    1788#
GNS    = ****** U       198     568     572     617     673     677     685     689     698     706    1815    1822    2420
                       2421    2422    2423    2424    2425    2426    2427    2428
GO        005320       1019    1024    1029    1034#
INK       006540       1248    1253    1258    1263#
INTRTN    005470       1053    1058    1063    1068#
```

I 5

UNIBUS EXERCISER   MACY11 30A(1052) 03-MAY-78 14:28 PAGE 56
CZKUAD.P11   26-APR-78 17:02   CROSS REFERENCE TABLE -- USER SYMBOLS   SEQ 0060

```
INTR1    005402        1015     1049#
INTR2    005420        1020     1054#
INTR3    005436        1025     1059#
INTR4    005454        1031     1064#
IOTVEC=  000020         126#     529*     530*
KIPAR0=  172340         182#    2055     2087
KIPAR1=  172342         183#
KIPAR2=  172344         184#
KIPAR3=  172346         185#
KIPAR4=  172350         186#
KIPAR5=  172352         187#
KIPAR6=  172354         188#
KIPAR7=  172356         189#    2102
KIPDR0=  172300         171#
KIPDR1=  172302         172#
KIPDR2=  172304         173#
KIPDR3=  172306         174#
KIPDR4=  172310         175#
KIPDR5=  172312         176#
KIPDR6=  172314         177#
KIPDR7=  172316         178#
LEEV     010546        1637     1659     1661#
LOAD1    005154        1013#
LOAD2    005212        1020#
LOAD3    005244        1025#
LOAD4    005276        1030#
LODDEV   002426         582#     606      622
MMVEC =  000250         138#
MOVREG   002462         591#     596
MOVVEC   002526         601#
MULT1    007500        1219     1228     1237     1442#
NG       003514         730#
NODEV    002544         587      609#
NODO     006270        1170     1205#
NOG      010636         737      740      743      746     1696#
NPRTST   003636         763#
NXT      006202        1176     1187#    1204
PBERR    006234        1178     1193#
PBPSW =  000116         468#    1180*    1188     1189*
PBRTN    006256        1179     1201#
PBVEC =  000114         467#    1179*    1188*
PIRQ  =  177772          38#
PIRQVE=  000240         132#
PR0   =  000000          55#     785      818      852      886      920      985     1037     1283
PR1   =  000040          56#
PR2   =  000100          57#
PR3   =  000140          58#     899      942     1084     1177     1778
PR4   =  000200          59#     744      865      903      915
PR5   =  000240          60#     741      831      869      881
PR6   =  000300          61#     738      797      835      847
PR7   =  000340          62#     563      577      733      735      764      768      770      780      801      803      813      837
                        871      905      936      939      973      980     1011     1016     1021     1026     1032     1086     1132
                       1134     1171     1285     1293     1301     1310     1450     1457     1464     1475     1576
PS    =  177776          35#      36
PSW   =  177776          36#     768*     774*     801*     807*     835*     841*     869*     875*     903*     909*     936*     942*
                        973*     985*    1011*    1037*    1084*    1132*    1177*    1283*    1698*    1778*
```

```
PWRFAL  010342         653     1139    1606#
PWRVEC= 000024         127#    535*    536*    653*    1136*   1139*   2434*   2435*   2443*   2458*   2459*
QNO     011174         555     1788#
Q1      010276         1572    1590#
Q2      010306         1593#
Q3      010316         1596#
Q4      010326         1599#
RDCHR = 104405         2028    2425#
RDLIN = 104406         2426#
RESREG= 104410         1664    2428#
RESVEC= 000010         122#
ROTATE  010600         1227    1236    1676#
SAVREG= 104407         1623    2427#
SERV1   006422         1244#   1449
SERV2   006446         1249#   1456
SERV3   006472         1254#   1466
SERV4   006516         1259#   1476
SIMLGO  001576         466#    1036*   1481*
SR0   = 177572         142#    2084    2098*   2105*
SR1   = 177574         143#
SR2   = 177576         144#
SR3   = 172516         145#    2097*
STACK = 001100         32#     528
START   001756         203     220     523#
STKLMT= 177774         37#
STVEC   010240         654     1570#
SWR     001136         259#    546     548*    553     564     659     1168    1858    1872    1874    1882    1889    1926
                       1933    1938    1944
SWREG   000176         200#    548
SW0   = 000001         90#
SW00  = 000001         80#     90
SW01  = 000002         79#     89
SW02  = 000004         78#     88
SW03  = 000010         77#     87
SW04  = 000020         76#     86
SW05  = 000040         75#     85
SW06  = 000100         74#     84
SW07  = 000200         73#     83
SW08  = 000400         72#     82
SW09  = 001000         71#     81
SW1   = 000002         89#
SW10  = 002000         70#
SW11  = 004000         69#
SW12  = 010000         68#
SW13  = 020000         67#
SW14  = 040000         66#
SW15  = 100000         65#
SW2   = 000004         88#
SW3   = 000010         87#
SW4   = 000020         86#
SW5   = 000040         85#
SW6   = 000100         84#
SW7   = 000200         83#
SW8   = 000400         82#
SW9   = 001000         81#
S1      007344         1284    1410#
```

K 5

UNIBUS EXERCISER   MACY11 30A(1052)   03-MAY-78 14:28   PAGE 58
CZKUAD.P11   26-APR-78 17:02   CROSS REFERENCE TABLE -- USER SYMBOLS   SEQ 0062

```
S2       007376           1292     1418#
S3       007430           1300     1425#
S4       007446           1309     1430#
TBITVE= 000014            123#
TCRLF = 000200            2207     2246#
THRU0   010562            562      1669#
THT   = 000011            2205     2245#
TKVEC = 000060            130#
TMPPWR  006026            1136     1151#
TPVEC = 000064            131#
TRAPVE= 000034            129#     533*     534*
TRTVEC= 000014            124#
TSTOVR  011142            1280     1446     1777#
TST1    003506            559      728#     1839
TST10   004760            946      950      971#
TST11   005140            997      999      1010#
TST12   005502            1040     1046     1081#
TST13   005666            1097     1123#
TST14   006046            1126     1148     1167#
TST15   006272            1169     1192     1215#
TST16   006556            1221     1226     1235     1240     1276#
TST17   007314            1339     1358     1377     1398#
TST2    003634            761#
TST3    003774            767      795#
TST4    004134            800      829#
TST5    004274            834      863#
TST6    004434            868      897#
TST7    004574            902      932#
TYMOUT  002662            589      627#     734      937
TYPDS = 104404            670      1819     1826     2424#
TYPE  = 104400            555      566      570      615      667      671      675      683      687      696      704      715      718
                          1813     1820     1827     1928     1936     1961     1978     1980     1983     1985     1989     1996     2032
                          2035     2039     2210     2309     2385     2420#    2460
TYPOC = 104401            1969     1993     2421#
TYPON = 104403            2423#
TYPOS = 104402            680      701      709      2422#
UIPAR0= 177640            160#
UIPAR1= 177642            161#
UIPAR2= 177644            162#
UIPAR3= 177646            163#
UIPAR4= 177650            164#
UIPAR5= 177652            165#
UIPAR6= 177654            166#
UIPAR7= 177656            167#
UIPDR0= 177600            149#
UIPDR1= 177602            150#
UIPDR2= 177604            151#
UIPDR3= 177606            152#
UIPDR4= 177610            153#
UIPDR5= 177612            154#
UIPDR6= 177614            155#
UIPDR7= 177616            156#
XQ      010334            1592     1595     1598     1601#
XTST    006016            1144     1146#
$BDADR  001122            253#
$BDDAT  001126            255#     526
```

```
$BELL    001214          285#    1928    1952
$CHARC   017460          2221*   2228    2237*   2242#
$CMTAG   001100          241#    524     531     537     538     539
$CM1   = 000006          271#    272#    273#    274#    275#    276#    277#        .
$CM2   = 000014          271#    272#    273#    274#    275#    276#    277#
$CM3   = 000006          269#    271
$CM4   = 000006          277#    278#    279#    280#    281#    282#    283#
$CORE    017054          2082    2108#
$CRLF    001221          287#    667     715     1936    1952    1961    1980    1985    1989    2048    2211    2248
$CROUT   017104          2108    2115#
$DBLK    020126          2351    2385    2393#
$DOAGN   015646          1809    1832    1838#
$DTBL    020116          2354    2389#
$ENDAD   015636          228     1834#   1941
$ENDCT   015502          537     1811#
$ENULL   015652          1840#
$EOP     015446          1406    1801#
$EOPCT   015474          537*    1808#   1812
$ERFLG   001103          244#    766     799     833     867     901     1225    1234    1848    1878    1880    1886*   1907
                         1923*   1952
$ERMAX   001115          250#    540*    766     799     833     867     901     1225    1234    1880    1902*   1907
$ERROR   016134          531     1922#
$ERRPC   001116          251#    1788    1930*   1931*   1932    1952    1967
$ERRTB   001224          305#    1975
$ERRTY   016302          1935    1960#
$ERTTL   001112          248#    1824    1828*   1929*   1952
$ESCAP   001212          284#    539*    19u1*   1947    1949    1952
$FILLC   001154          267#    2214    2248
$FILLS   001153          266#    2248
$GDADR   001120          252#
$GDDAT   001124          254#
$GET42   015626          1829#
$HD    = 000001          14      15
$ICNT    001104          245#    1788    1893*   1894    1896*   1906
$ILLUP   020324          2434    2463#
$ITEMB   001114          249#    1932*   1952    1964
$KTNEX   017046          2083    2107#
$KTOUT   017036          2099    2104#
$KT11    016702          2081#   2085*   2107*
$LF      001222          288#    1952    2039    2048    2248
$LPADR   001106          246#    541*    1884*   1899*   1904    1906
$LPERR   001110          247#    542*    1884    1900*   1906    1946
$LSTAD   017150          2120*   2127#
$LSTBK   017152          658     2121*   2128#
$MAIL  = ****** U        553     729     762     796     830     864     898     933     972     1011    1082    1125    1168
                         1216    1277    1400    1899    1938    2199
$MXCNT   016132          1897    1906#
$NULL    001152          265#    1142    1152    1827    2216    2248
$NWTST = 000001          722#    724     752#    754     788#    790     822#    824     856#    858     890#    892     924#
                         926     963#    965     1002#   1004    1074#   1076    1116#   1118    1157#   1159    1208#   1210
                         1270#   1272    1395#
$OCNT    017706          2281*   2310*   2323#
$OMODE   017710          2276*   2280*   2285    2288*   2299*   2325#
$OVER    016116          1859    1877    1885    1895    1903#
$PASS    001100          242#    1805*   1806*   1817    1840    1891    1907
$POWER   020332          1147    2461    2466#
```

```
$PWRDN  020202        535    1154   1619   2434#   2458
$PWRMG  020320       1142    1147*  1152*  2461#
$PWRUP  020250       2443    2448#
$QUES   001220        286#   1952   2032   2048    2248
$RDCHR  016436       2010#   2425
$RDDEC= ****** U     2427
$RDLIN  016472       2024#   2426
$RDOCT= ****** U     2427
$RDSZ = 000016       2017#
$REGAD  001156        269#
$REG0   001160        271#    516    657*   662*    664    695*    708    778*    811*    845*    879*    913*    947*
                     1334*   1353*  1521*  1544*  1563*  1714*   1788
$REG1   001162        272#    779*   812*   846*    880*   914*    948*   992*   1041*   1108*   1333*   1352*   1372*
                     1390*   1487*  1488   1520*  1543*  1562*   1715*  1724*   1725   1788
$REG2   001164        273#    629*   630*   780*    813*   847*    881*   915*    993*   1042*   1109*   1264*   1325*
                     1328    1330*  1335   1373*  1391*  1491    1670*  1671*   1716*  1738*   1779*   1788
$REG3   001166        274#    781*   814*   848*    882*   916*   1043*  1110*   1335*  1354*   1522*   1542*   1561*
                     1717*   1788
$REG4   001170        275#   1044*  1308*  1388    1390   1431*   1718*  1755*   1756*  1757    1788
$REG5   001172        276#    592*   956*  1069*   1101*  1194*   1265*  1436*   1788
$RESRE  017212       2163#   2428
$R2A  = ****** U     2429
$SAVRE  017154       2147#   2427
$SAVR6  020330       2442*   2448   2449*  2450*   2465#
$SCOPE  015656        529    1857#
$SETUP= 000037        191#    529    531    533     535    537     538    539     541   1803    1941
$SIZE   016616        576    2065#
$SIZEX  017110       2106    2116#
$SIZE1  016670       2076    2078#
$STUP = 177777        191#
$SVLAD  016070       1867    1898#
$SVPC = 000232        226#    231
$SWR  = 167400          2#     14     20     21      22     23      24     25      26    283     284     285     538
                      539     541    542    729     762    796     830    864     898    933     972    1011    1082
                     1124    1168   1216   1277    1399   1798    1804   1830    1839   1840    1849    1850    1851
                     1852    1853   1858   1870    1872   1873    1878   1879    1880   1887    1888    1889    1900
                     1903    1906   1914   1915    1916   1917    1918   1926    1933   1938    1944    1952    2462
$SWRMK= 000300          3#     26     27   1853    1854   1874    1875
$TIMES  001210        283#    538*  1124*  1399*   1804*  1887*   1894   1897*   1906
$TKB    001144        262#   1401   2004   2014
$TKS    001142        261#   2004   2012
$TMP0   001174        202*    219*   277#   557     594*   595     597*   656*    658*   663*    666*    668     691*
                      694*    700    713
$TMP1   001176        278#   1135*
$TMP2   001200        279#
$TMP3   001202        280#   1172*  1174*  1175
$TMP4   001204        281#    957*  1052*  1057*   1062*  1067*   1102*  1131*   1196*  1247*   1252*   1257*   1262*
                     1414*   1421*  1426*  1432*   1591*  1594*   1597*  1600*   1629   1788
$TMP5   001206        282#    518
$TN  =  000020         14#    722    729#   752     762#   767     788    796#    800    822     830#    834     856
                      864#    868    890    898#    902    924     933#   946     950    963     972#    997     999
                     1002    1011#  1040   1046    1074   1082#   1097   1116    1124#  1126    1148    1157    1168#
                     1192    1208   1216#  1221    1226   1235    1240   1270    1277#  1339    1358    1377    1395
                     1399#
$TPB    001150        264#   2234*  2248
$TPFLG  001155        268#   2193   2248
```

```
$TPS    001146        263#    2232    2248
$TRAP   020136        533     2403#
$TRP  = 000011        2411#   2421#   2422#   2423#   2424#   2425#   2426#   2427#   2428#   2429#
$TRPAD  020160        2408    2419#
$TSTNM  001102        243#    1788    1803*   1848    1876    1898*   1903    1907    1925    1952
$TTYIN  016600        2025    2026    2043    2047#
$TYPBN= ****** U      2425
$TYPOS  017712        2339#   2424
$TYPE   017250        2193#   2411    2420
$TYPEC  017414        2213    2220    2227    2232#   2233
$TYPEX  017462        2238    2240    2243#
$TYPOC  017510        2279#   2421
$TYPON  017524        2278    2281#   2423
$TYPOS  017464        2274#   2422
$XTSTR  015666        1861#
$OFILL  017707        2275*   2279*   2289    2324#
$40CAT= ****** U      1858    1935
.     = 020342        194#    198#    201#    218#    226     227#    229#    231#    240#    289    527    541    542
                      566     569#    570     615     618#    671     675     683     686#    687    690#   696    699#
                      704     1788#   1813    1816#   1820    1840    1841#   1906    1907    1952   1999#  2004   2047#
                      2048    2248    2393#   2445    2464
```

B 6

UNIBUS EXERCISER   MACY11 30A(1052) 03-MAY-78 14:28 PAGE 63
CZKUAD.P11   26-APR-78 17:02   CROSS REFERENCE TABLE -- MACRO NAMES   SEQ 0066

BE1	762#	765	798	832	866	900									
COMMEN	1#	133#	205	1317	1340	1359	1378	1482	1494	1529	1549				
ENDCOM	1#	133#	214	1321	1344	1363	1382	1486	1499	1533	1553				
ERROR	33#	631	782	815	849	883	917	949	994	998	1045	1111	1145	1195	1336
	1355	1374	1392	1523	1545	1564	1634	1640	1644	1648	1652	1656	1660	1672	1719
	1739														
ESCAPE	1#	133#													
GETPRI	1#														
MSSG	721#	724	752#	754	787#	790	821#	824	855#	858	889#	892	923#	926	962#
	965	1001#	1004	1073#	1076	1115#	1118	1156#	1159	1207#	1210	1270#	1272		
MULT	1#	133#													
NEWTST	1#	133#	722	752	788	822	856	890	924	963	1002	1074	1116	1157	1208
	1270	1395													
NOGRNT	751#	1697													
POP	1#	133#	2168	2380	2452										
PUSH	1#	133#	2148	2339	2436										
REPORT	1#														
SCOPE	34#	728	761	795	829	863	897	932	971	1010	1081	1123	1167	1215	1276
	1398	1802													
SETPRI	1#														
SETTRA	2411#	2421	2422	2423	2424	2425	2426	2427	2428						
SETUP	1#	133#	524												
SKIP	1#	133#	767	800	834	868	902	946	950	997	999	1040	1046	1097	1126
	1148	1192	1221	1226	1235	1240	1339	1358	1377						
SLASH	1#	133#	649	1567											
SPACE	133#														
STARS	1#	133#	222	233	289	306	462	509	521	607	625	633	722	727	752
	760	788	794	822	828	856	862	890	896	922	924	931	951	963	970
	1002	1009	1047	1074	1080	1098	1116	1122	1149	1157	1166	1199	1208	1214	1241
	1268	1270	1275	1395	1397	1407	1409	1440	1493	1582	1604	1620	1666	1674	1694
	1712	1721	1722	1741	1751	1760	1775	1785	1788	1791	1842	1907	1952	2000	2017
	2048	2129	2175	2248	2326	2394	2429	2469							
SWRSU	1#	543#													
TRMTRP	2411#														
TYPBIN	1#	133#													
TYPDEC	1#	133#	668	1817	1824										
TYPNAM	1#														
TYPNUM	1#	133#													
TYPOCS	1#	133#	679	700	708										
TYPOCT	1#	133#	1967	1991											
TYPTXT	1#	133#	566	570	615	671	675	683	687	696	704	1813	1820		
$SCMRE	233#	271	272	273	274	275	276								
$SCMTM	233#	277	278	279	280	281	282								
$SESCA	1#	133#													
$SNEWT	1#	133#	722	752	788	822	856	890	924	963	1002	1074	1116	1157	1208
	1270	1395													
$SSET	2411#	2421	2422	2423	2424	2425	2426	2427	2428						
$SSKIP	1#	133#	767	800	834	868	902	946	950	997	999	1040	1046	1097	1126
	1148	1192	1221	1226	1235	1240	1339	1358	1377						
.EQUAT	1#	2#	28												
.HEADE	1#	2#	4												
.KT11	1#	2#	133												
.SETUP	1#	2#	191												
.SWRHI	1#	2#	15												
.SWRLO	2#	27#													
.SACT1	1#	2#	222												

```
.$APTB     1#
.$APTH     1#
.$APTY     1#
.$ASTA     1#
.$CATC     1#        2#     191
.$CMTA     1#        2#     233
.$DB2D     1#
.$DB2O     1#
.$DIV      1#
.$EOP      1#        2#    1791
.$ERRO     1#        2#    1907
.$ERRT     1#        2#    1952
.$MULT     1#
.$POWE     1#        2#    2429
.$RAND     1#
.$RDDE     1#
.$RDOC     1#
.$READ     1#        2#    2000
.$R2AZ     1#
.$SAVE     1#        2#    2129
.$SB2D     1#
.$SB2O     1#
.$SCOP     1#        2#    1842
.$SIZE     1#        2#    2048
.$SUPR     1#
.$TRAP     1#        2#    2394
.$TYPB     1#
.$TYPD     1#        2#    2326
.$TYPE     1#        2#    2175
.$TYPO     1#        2#    2248
.$40CA     1#


. ABS.  020342      000


ERRORS DETECTED:  0

CZKUAD.BIN,CZKUAD.LST/CRF/SOL/NL:TOC=DSKZ:CZKUAD.SML,DSKZ:CZKUAD.P11
RUN-TIME: 13 16 1 SECONDS
RUN-TIME RATIO: 263/31-8.3
CORE USED:  28K  (55 PAGES)
```