File created: 29-Sep-85 18:11:53 **{ERIS}<LISPCORE>BUSMASTER>KOTO>TEST2>PCMEMTEST.;7**

changes to: (FNS ShowResults StartBltTest PCMEM.MAKETEST BMTSetValue BMTChangeDirection FastTestBltIn QuietTestBltIn QuietTestBltOut TestBltOut TestBltIn RecordError)

previous date: 18-Sep-85 14:58:03 {ERIS}<LISPCORE>BUSMASTER>KOTO>TEST2>PCMEMTEST.;1

Read Table: OLD-INTERLISP-FILE

Package: INTERLISP

Format: XCCS


```
(RPAQQ PCMEMTESTCOMS
       [(* PC memory checkout and test tools)
        (FNS PCMEM.CHECKOUT RecordError SetupTestArray ShowErrors ShowResults ShowWords StartBltTest StopBltTest
             TestBltIn TestBltOut FastTestBltIn QuietTestBltIn QuietTestBltOut)
        (VARS PCMEM.READTESTPATS PCMEM.THRESH PCMEM.WRITETESTPATS)
        (FNS PCMEM.MAKETEST BMTRead BMTChangeDirection BMTSetValue DoTB)
        [VARS (BMTArray1)
              (BMTArray2)
              (BMTInPatternSpecs ' ((zeros 0)
                                    (ones 65535)
                                    (alt0 NIL NIL "fixed pattern of alternating 0s and 1s, starting with 0")
                                    (alt1 NIL NIL "fixed pattern of alternating 0s and 1s, starting with 1")
                                    (rand NIL NIL "random words")
                                    ((TogMenuValue)
                                     (BMTRead "Pattern")
                                     NIL "will prompt and read" Other)))
              (BMTOutPatternSpecs ' ((zeros 0)
                                     (ones 65535)
                                     (alt0 NIL NIL "fixed pattern of alternating 0s and 1s, starting with 0")
                                     (alt1 NIL NIL "fixed pattern of alternating 0s and 1s, starting with 1")
                                     (alt NIL NIL "0 -1 ... alternating every other pass with -1 0 ...")
                                     (altAll NIL NIL "zeros alternating every other pass with ones")
                                     (newRand NIL NIL "random words, new each pass")
                                     (fixedRand NIL NIL "random words, same each pass")
                                     ((TogMenuValue)
                                      (BMTRead "Pattern")
                                      NIL "will prompt and read" Other)))
              (BMTTestTogMenuSpecs ' (("Direction" (In (BMTChangeDirection 'In))
                                              (Out (BMTChangeDirection 'Out))
                                              (FastIn (BMTChangeDirection 'FastIn))
                                              (FastOut (BMTChangeDirection 'FastOut)))
                                      ("Mode" (Straight (NILL))
                                              (Swapped SWAP))
                                      (Pattern)
                                      ("Save results" (No (BMTSetValue 'save NIL))
                                              (Yes (BMTSetValue 'save T)))
                                      ("Show every error" (No (BMTSetValue 'show NIL))
                                              (Yes (BMTSetValue 'show T)))
                                      ("Summarize every" (1 (BMTSetValue 'sumEvery 1))
                                              (10 (BMTSetValue 'sumEvery 10))
                                              ((TogMenuValue)
                                               (BMTSetValue 'sumEvery (BMTRead "Summarize every"))
                                               NIL "will prompt and read" Other))
                                      ("Type of summary" (%. (BMTSetValue 'sumType '.))
                                              (Full (BMTSetValue 'sumType 'Full)))
                                      ("# passes" (Forever 2147483647)
                                              ((TogMenuValue)
                                               (BMTRead "# passes")
                                               NIL "will prompt and read" Other))
                                      ("Size of block" 100 1000 5000 10000 32768 ((TogMenuValue)
                                                                                  (BMTRead "Size of block")
                                                                                  NIL "will prompt and read"
                                                                                  Other))
                                      ("Dismiss" (Yes (BMTSetValue 'block? T))
                                              (No (BMTSetValue 'block? NIL)
                                                  NIL "Enable/disable blocking - No is dangerous!"]
        (P (PUTASSOC 'Pattern BMTInPatternSpecs BMTTestTogMenuSpecs))
        (BITMAPS BusmasterIcon)
        (FILES (SYSLOAD)
               BUSMASTER)
        (DECLARE: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY (P (RESETSAVE DWIMIFYCOMPFLG T))
               (GLOBALVARS BMTArray1 BMTArray2)
               (FILES (LOADCOMP)
                      BUSEXTENDER.DCOM))
        (DECLARE: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA)
                                                                              (NLAML DoTB)
                                                                              (LAMA])
```

(* * PC memory checkout and test tools)

```
(DEFINEQ
```

(PCMEM.CHECKOUT

```
  [LAMBDA (maxPageNumber)                                   (* ht: " 4-Jun-85 10:28")
```

(* * run some simple tests to validate the 1109-Busmaster-PCmemory paths and contents - assumes that BUS.CHECKOUT and PCRCVR.CHECKOUT have been run successfully)

```
    (PROG NIL
          (BUS.RESET)
          (BUSDMA.INIT)
          (LET ((pages (for i from 0 to (IMAX (OR maxPageNumber 0)
                                              15)
                          when (AND (PROGN (PCBUS.WRITEHL i 0 0)
                                           (PCBUS.READHL i 0)
                                           =0)
                                    (PROGN (PCBUS.WRITEHL i 0 255)
                                           (PCBUS.READHL i 0)
                                           =255))
                          collect i)))
               (if pages=NIL
                   then (printout T "There does not appear to be any memory connected to the PC in page address
                                    range 0 - " (IMAX (OR maxPageNumber 0)
                                                      15)
                                    "." T "Please check that the busmaster and PC are powered up and running
                                    correctly by calling (BUS.CHECKOUT) and (PCRCVR.CHECKOUT), and check that there
                                    is at least one memory board installed in the PC.  If that doesn't help, check
                                    the page address switches on the memory board to see that they are in the
                                    indicated range." T)
                        (RETURN)
                 else (printout T "There is memory on the PC at page address(es) " .PPVTL pages "." T))
               (if [AND (BUSDMA.READADDRESS 0)
                        =
                        (PROGN (DISMISS 1)
                               (BUSDMA.READADDRESS 0))
                        (NOT (AND (BUSDMA.READTCBIT 0 T)
                                  (PROGN (DISMISS 50)
                                         (BUSDMA.READTCBIT 0 T]
                   then (printout T "The memory refresh DMA is not running." T)
                 else (printout T "Memory refresh DMA OK" T))
               (if (NOT (ARRAYP BMTArray1))
                   then BMTArray1_ (ARRAY 32768 'WORD))
               (if (NOT (ARRAYP BMTArray2))
                   then BMTArray2_ (ARRAY 32768 'WORD))
               (bind result for p in pages
                  do (if result_ (for pat in PCMEM.READTESTPATS
                                    thereis (if (GREATERP (QuietTestBltIn p 'STRAIGHT pat 5 32768 BMTArray1 BMTArray2)
                                                          PCMEM.THRESH)
                                                else (PRIN1 "+" T)
                                                NIL))
                                then (printout T "Page " p " read test errors for pattern " result T)
                              else (printout T "Page " p " read test OK" T))
                         (if result_ (for pat in PCMEM.WRITETESTPATS
                                        thereis (if (GREATERP (QuietTestBltOut p 'STRAIGHT pat 2 32768 BMTArray1 BMTArray2
                                                                                                                    )
                                                              PCMEM.THRESH)
                                                    else (PRIN1 "+" T)
                                                    NIL))
                                    then (printout T "Page " p " write test errors for pattern " result T)
                                  else (printout T "Page " p " write test OK" T])
```

(RecordError

```
  [LAMBDA (aList value firstIndex secondIndex)                  (* edited: "29-Sep-85 17:17")
    (LET ((topEntry (CDR (ASSOC firstIndex aList)))
          subEntry)
         (if (NOT topEntry)
             then (PUTASSOC firstIndex (SETQ topEntry (LIST NIL))
                            aList))
         (if (SETQ subEntry (CDR (ASSOC secondIndex topEntry)))
             then (if (NOT (FMEMB value subEntry))
                      then (NCONC1 subEntry value))
           else (PUTASSOC secondIndex (LIST value)
                          topEntry])
```

(SetupTestArray

```
  [LAMBDA (array1 pattern n)                                    (* ht: " 7-Apr-85 09:31")
    (for i from 1 to n do (SETA array1 i (SELECTQ pattern
                                          (0 0)
                                          ((NIL rand)
                                              (RAND 0 65535))
                                          (alt0 (if (EVENP i)
                                                    then 65535
                                                  else 0))
                                          (alt1 (if (EVENP i)
                                                    then 0
```

```
                                                    else 65535))
                                                (if (NUMBERP pattern)
                                                    then pattern
                                                  else (SHOULDNT "not a valid pattern" pattern])
```

(ShowErrors

```
[LAMBDA (aList)                                                    (* ht: " 7-Apr-85 14:47")
  (RESETFORM (RADIX 16)
         (for topEntry in aList do (printout NIL 20 .I4.16 topEntry:1 #
                                             (for subEntry in topEntry::2
                                                do (printout NIL 25 .I4.16 subEntry:1 , .PARA 30 50 (SORT
                                                                                                      subEntry::1
                                                                                                       ])
```

(ShowResults

```
[LAMBDA (j n pattern mode c1 c2 c3 ln lp mn mp sn sp out?)     (* ht: "12-Apr-85 14:05")
  (printout T T j " passes * " n " words = " (TIMES j n)
         (if out?
             then " writes"
           else " reads")
         " of pattern " # (RESETFORM (RADIX 16)
                                     (PRIN1 pattern))
         " in "
         (OR mode (QUOTE STRAIGHT))
         " mode." T)
  (if (OR lp mp)
      then (printout T 20 c1 25 c2 30 c3 T))
  (printout T ln # (if (NEQ ln 0)
                       then (printout NIL " (" (FQUOTIENT ln (QUOTIENT (TIMES j n)
                                                                       1000.0))
                                    " per thousand)"))
         (if lp
             then " read errors: "
           else " read errors,"))
  (if lp
      then (ShowErrors lp))
  (printout T T mn # (if (NEQ mn 0)
                         then (printout NIL " (" (FQUOTIENT mn (QUOTIENT (TIMES j n)
                                                                         1000.0))
                                      " per thousand)"))
         (if mp
             then (if out?
                      then " write errors: "
                    else " memory decays: ")
           else (if out?
                    then " write errors,"
                  else " memory decays,")))
  (if mp
      then (ShowErrors mp))
  (if sp
      then (printout T T 20 "1st" 25 "2nd" 30 "addrs"))
  (printout T T sn # (if (NEQ sn 0)
                         then (printout NIL " (" (FQUOTIENT sn (QUOTIENT (PLUS ln mn)
                                                                         1000.0))
                                      " per thousand)"))
         (if sp
             then " slow faults: "
           else " slow faults."))
  (if sp
      then (ShowErrors sp))
  (TERPRI T])
```

(ShowWords

```
[LAMBDA (inRadix outRadix offset)                                  (* edited: " 3-Jun-85 17:08")
  (RESETFORM (RADIX (OR outRadix 16))
         (bind addr until (PEEKC T) = 's do (printout T (PCBUS.READWORD 1 (OR offset -1)
                                                                        +
                                                                        (TrueRadixRead (OR inRadix 16)))
                                                T)
            finally (READC T])
```

(StartBltTest

```
[LAMBDA NIL                                                        (* ht: "14-Apr-85 16:28")
  (LET* [(mw (MAINWINDOW $$TogWindow$$ T))
         (menus (WINDOWPROP mw (QUOTE BMTMenus]
        (WINDOWPROP mw (QUOTE PROCESS)
               (ADD.PROCESS [LIST (QUOTE DoTB)
                                  (SELECTQ (TogMenuValue (CAR menus))
                                       (In (QUOTE TestBltIn))
                                       (Out 'TestBltOut)
                                       (FastIn 'FastTestBltIn)
                                       (FastOut 'FastTestBltOut)
                                       (SHOULDNT))
```

```
                                    (NCONC (for m in (CDR menus) collect (TogMenuValue m))
                                           (WINDOWPROP mw (QUOTE BMTArrays]
                            (QUOTE WINDOW)
                            mw])
```

(StopBltTest

```
  [LAMBDA NIL                                                    (* ht: "10-Apr-85 15:21")
    (LET* ((mw (MAINWINDOW $$TogWindow$$ T))
           (proc (WINDOWPROP mw 'PROCESS)))
          (printout mw T "Stopping...")
          (PROCESS.EVAL proc '  (SETQ $Stop$ T])
```

(TestBltIn

```
  [LAMBDA (mode pattern save show sumEvery sumType numPasses n block? array1 array2)
                                                                 (* edited: "29-Sep-85 17:32")
    (DECLARE (SPECVARS save show sumEvery sumType block?))
    (if (NOT n)
        then (SETQ n (ARRAYSIZE array1)))
    (SetupTestArray array1 pattern n)
    (if block?
        then (BLOCK))
    (BUSDMA.INIT)
    (PCBUS.WRITEARRAY array1 0 n mode)
    (if block?
        then (BLOCK))
    (printout T "Pattern stored, " (for i from 1 to n unless (EQ (PCBUS.READWORD 1 (DIFFERENCE i 1)
                                                                              mode)
                                                                     (ELT array1 i))
                                                 count (PCBUS.WRITEWORD 1 (DIFFERENCE i 1)
                                                                 (ELT array1 i)
                                                                 mode))
           " errors," T "starting test." T)
    (PROG ((losses 0)
           (memFaults 0)
           (slowFaults 0)
           (lossPairs (LIST NIL))
           (memPairs (LIST NIL))
           (slowPairs (LIST NIL))
           type $Stop$ true)
          (declare (SPECVARS $Stop$) for j from 1 to (OR numPasses MAX.SMALLP)
             until (OR $Stop$ (KEYDOWNP (QUOTE STOP)))
             do
             (PCBUS.READARRAY array2 0 n mode)
             (if block?
                 then (BLOCK))
             [for i from 1 to n unless (EQ (ELT array1 i)
                                           (ELT array2 i))
                do (bind prev do (SETQ true (PCBUS.READWORD 1 (DIFFERENCE i 1)
                                                            mode))
                      repeatuntil (if (EQ true (ELT array1 i))
                                      then (SETQQ type r)
                                           [if save
                                               then (SELECTQ pattern
                                                            (rand (RecordError lossPairs (ELT array2 i)
                                                                               i
                                                                               (ELT array1 i)))
                                                            (RecordError lossPairs i (ELT array1 i)
                                                                         (ELT array2 i]
                                           (add losses 1)
                                    elseif prev
                                      then (if (EQ prev true)
                                               then (SETQQ type m)
                                                    (add memFaults 1)
                                                    [if save
                                                        then (SELECTQ pattern
                                                                      (rand (RecordError memPairs (ELT array1 i)
                                                                                         i
                                                                                         (ELT array2 i)))
                                                                      (RecordError memPairs i (ELT array1 i)
                                                                                   (ELT array2 i]
                                                    (PCBUS.WRITEWORD 1 (DIFFERENCE i 1)
                                                                     (ELT array1 i))
                                             else (add slowFaults 1)
                                                  (if save
                                                      then (RecordError slowPairs i prev true))
                                                  (if show
                                                      then (RESETFORM (RADIX 16)
                                                                      (printout T "s" i 6 prev , true T)))
                                                  (SETQ prev true)
                                                  NIL)
                                    else (SETQ prev true)
                                         NIL))
                   (if show
                       then (RESETFORM (RADIX 16)
                                       (printout T type i 6 (ELT array1 i)
```

```
                                    12
                                    (ELT array2 i)
                                    18 true T]
            (if block?
                then (BLOCK))
            (if (EQ (IMOD j (OR sumEvery 10))
                    0)
                then (SELECTQ sumType
                          (%. (PRIN1 "." T))
                          (printout T losses , "read errors, " memFaults " memory decays, " slowFaults " slow
                                faults." T)))
            finally (SELECTQ pattern
                        (rand (ShowResults (DIFFERENCE j 1)
                                    n pattern mode "addr" "val" "errs" losses (SORT (CDR lossPairs)
                                                                                    T)
                                    memFaults
                                    (SORT (CDR memPairs)
                                          T)
                                    slowFaults
                                    (CDR slowPairs)))
                              (ShowResults (DIFFERENCE j 1)
                                    n pattern mode "val" "err" "addrs" losses (CDR lossPairs)
                                    memFaults
                                    (CDR memPairs)
                                    slowFaults
                                    (CDR slowPairs])
```

(TestBltOut

```
  [LAMBDA (mode pattern save show sumEvery sumType numPasses n block? array1 array2)
                                                                    (* edited: "29-Sep-85 17:43")
    (DECLARE (SPECVARS save show sumEvery sumType block?))
    (if (NOT n)
        then (SETQ n (ARRAYSIZE array1)))
    (SELECTQ pattern
         ((alt altAll newRand))
         (fixedRand (SetupTestArray array1 'rand n))
         (SetupTestArray array1 pattern n))
    (if block?
        then (BLOCK))
    (printout T "Pattern initialized, starting test." T)
    (PROG ((losses 0)
           (memFaults 0)
           (slowFaults 0)
           (lossPairs (LIST NIL))
           (memPairs (LIST NIL))
           (slowPairs (LIST NIL))
           type $Stop$ true)
          (declare (SPECVARS $Stop$) for j from 1 to (OR numPasses MAX.SMALLP)
             until (OR $Stop$ (KEYDOWNP (QUOTE STOP)))
             do
             (SELECTQ pattern
                  (newRand (SetupTestArray array1 pattern n))
                  (alt (SetupTestArray array1 (if (EVENP j)
                                                  then (QUOTE alt0)
                                                else (QUOTE alt1))
                               n))
                  (altAll (SetupTestArray array1 (if (EVENP j)
                                                     then 0
                                                   else 65535)
                                n))
                  NIL)
             (if block?
                 then (BLOCK))
             (PCBUS.WRITEARRAY array1 0 n)
             (if block?
                 then (BLOCK))
             (PCBUS.READARRAY array2 0 n)
             (if block?
                 then (BLOCK))
             [for i from 1 to n unless (EQ (ELT array1 i)
                                           (ELT array2 i))
                do (bind prev do (SETQ true (PCBUS.READWORD 1 (DIFFERENCE i 1)
                                                            mode))
                      repeatuntil (if (EQ true (ELT array1 i))
                                      then (SETQQ type r)
                                           [if save
                                               then (SELECTQ pattern
                                                         (rand (RecordError lossPairs (ELT array2 i)
                                                                            i
                                                                            (ELT array1 i)))
                                                               (RecordError lossPairs i (ELT array1 i)
                                                                      (ELT array2 i]
                                           (add losses 1)
                                    elseif prev
                                      then (if (EQ prev true)
                                               then (SETQQ type w)
```

```
                                        (add memFaults 1)
                                        [if save
                                            then (SELECTQ pattern
                                                         (rand (RecordError memPairs (ELT array1 i)
                                                                       i
                                                                       (ELT array2 i)))
                                                         (RecordError memPairs i (ELT array1 i)
                                                                      (ELT array2 i]
                                        (PCBUS.WRITEWORD 1 (DIFFERENCE i 1)
                                                         (ELT array1 i))
                                    else (add slowFaults 1)
                                         (if save
                                             then (RecordError slowPairs i prev true))
                                         (if show
                                             then (RESETFORM (RADIX 16)
                                                             (printout T "s" i 6 prev , true T)))
                                         (SETQ prev true)
                                         NIL)
                               else (SETQ prev true)
                                    NIL))
                    (if show
                        then (RESETFORM (RADIX 16)
                                        (printout T type i 6 (ELT array1 i)
                                                  12
                                                  (ELT array2 i)
                                                  18 true T]
               (if block?
                   then (BLOCK))
               (if (EQ (IMOD j (OR sumEvery 10))
                       0)
                   then (SELECTQ sumType
                                 (%. (PRIN1 "." T))
                                 (printout T losses , "read errors, " memFaults " write errors, " slowFaults " slow
                                           faults." T)))
               finally (SELECTQ pattern
                                (rand (ShowResults (DIFFERENCE j 1)
                                                   n pattern mode "addr" "val" "errs" losses (SORT (CDR lossPairs)
                                                                                                   T)
                                                   memFaults
                                                   (SORT (CDR memPairs)
                                                         T)
                                                   slowFaults
                                                   (CDR slowPairs)
                                                   T))
                                      (ShowResults (DIFFERENCE j 1)
                                                   n pattern mode "val" "err" "addrs" losses (CDR lossPairs)
                                                   memFaults
                                                   (CDR memPairs)
                                                   slowFaults
                                                   (CDR slowPairs)
                                                   T])
```

(FastTestBltIn

```
  [LAMBDA (mode pattern save show sumEvery sumType numPasses n block? array1 array2)
                                                                    (* edited: "29-Sep-85 17:50")
    (DECLARE (SPECVARS save show sumEvery sumType block?))
    (if (NOT n)
        then n_ (ARRAYSIZE array1))
    (SetupTestArray array1 pattern n)
    (BUSDMA.INIT)
    (PCBUS.WRITEARRAY array1 0 n mode)
    (printout T "Pattern stored, " (for i from 1 to n unless (PCBUS.READWORD 1 (DIFFERENCE i 1)
                                                                             mode)
                                                                    =
                                                                    (ELT array1 i)
                                                  count (PCBUS.WRITEWORD 1 (DIFFERENCE i 1)
                                                                           (ELT array1 i)
                                                                           mode))
           " errors," T "starting test." T)
    (PROG ((losses 0)
           (memFaults 0)
           (slowFaults 0)
           (lossPairs (LIST NIL))
           (memPairs (LIST NIL))
           (slowPairs (LIST NIL))
           type true)
          (declare for j from 1 to (OR numPasses MAX.SMALLP) until (KEYDOWNP 'STOP)
             do
             (bind (top _ n)
                   firstBad until (OR top=0 firstBad_ (PCBUS.TESTARRAY array1 0 top mode)
                                      =0)
                do (bind prev do true_ (PCBUS.READWORD 1 firstBad-1 mode)
                      repeatuntil (if true= (ELT array1 firstBad)
                                      then (type_ 'r)
                                           (if save
                                               then (SELECTQ pattern
```

```
                                                (rand (RecordError lossPairs NIL firstBad (ELT array1
                                                                                                 firstBad)))
                                                (RecordError lossPairs firstBad (ELT array1 firstBad)
                                                       NIL)))
                                        (add losses 1)
                                   elseif prev
                                     then (if prev=true
                                              then (type_ 'm)
                                                   (add memFaults 1)
                                                   (if save
                                                       then (SELECTQ pattern
                                                                  (rand (RecordError memPairs (ELT array1 firstBad)
                                                                                firstBad NIL))
                                                                  (RecordError memPairs firstBad (ELT array1 firstBad)
                                                                         NIL)))
                                                   (PCBUS.WRITEWORD 1 (DIFFERENCE firstBad 1)
                                                          (ELT array1 firstBad))
                                            else (add slowFaults 1)
                                                 (if save
                                                     then (RecordError slowPairs firstBad prev true))
                                                 (if show
                                                     then (RESETFORM (RADIX 16)
                                                                  (printout T "s" firstBad 6 prev , true T)))
                                                 (prev_true)
                                                NIL)
                                   else (prev_true)
                                        NIL))
                          (if show
                              then (RESETFORM (RADIX 16)
                                       (printout T type firstBad 6 (ELT array1 firstBad)
                                              12 true T)))
                          (top_firstBad-1))
                    (if (IMOD j (OR sumEvery 10))
                       =0
                        then (SELECTQ sumType
                                  (%. (PRIN1 "." T))
                                  (printout T losses , "read errors, " memFaults " memory decays, " slowFaults " slow
                                         faults." T)))
                    finally (SELECTQ pattern
                                 (rand (ShowResults (DIFFERENCE j 1)
                                                    n pattern mode "addr" "val" "errs" losses (SORT lossPairs::1 T)
                                                    memFaults
                                                    (SORT memPairs::1 T)
                                                    slowFaults slowPairs::1))
                                 (ShowResults (DIFFERENCE j 1)
                                        n pattern mode "val" "err" "addrs" losses lossPairs::1 memFaults memPairs::1
                                        slowFaults slowPairs::1])
```

(QuietTestBltIn

```
  [LAMBDA (page mode pattern numPasses n array1 array2)          (* edited: "29-Sep-85 17:51")
    (LET ((total 0))
         (if (NOT n)
             then n_ (ARRAYSIZE array1))
         (SetupTestArray array1 pattern n)
         (BUS.RESET)
         (BUSDMA.INIT)
         (PCBUS.WRITEARRAY array1 0 n mode 1 page)
         (add total (for i from 1 to n unless (PCBUS.READWORD page (DIFFERENCE i 1)
                                                     mode)
                                              =
                                              (ELT array1 i)
                       count (PCBUS.WRITEWORD page (DIFFERENCE i 1)
                                    (ELT array1 i)
                                    mode)))
         (PROG ((losses 0)
                (memFaults 0)
                (slowFaults 0)
                type true)
               (for j from 1 to (OR numPasses MAX.SMALLP) until (KEYDOWNP 'STOP)
                  do (PCBUS.READARRAY array2 0 n mode 1 page)
                     (for i from 1 to n unless (ELT array1 i) = (ELT array2 i)
                        do (bind prev do true_ (PCBUS.READWORD page (DIFFERENCE i 1)
                                                      mode)
                                repeatuntil (if true= (ELT array1 i)
                                                then (type_ 'r)
                                                     (add losses 1)
                                             elseif prev
                                               then (if prev=true
                                                        then (type_ 'm)
                                                             (add memFaults 1)
                                                             (PCBUS.WRITEWORD page (DIFFERENCE i 1)
                                                                    (ELT array1 i))
                                                      else (add slowFaults 1)
                                                           (prev_true)
                                                           NIL)
                                             else (prev_true)
```

```
                                                NIL)))
                        (PRIN1 "." T)
                    finally total_ (PLUS total losses memFaults slowFaults)))
            total])
```

(QuietTestBltOut

```
  [LAMBDA (page mode pattern numPasses n array1 array2)          (* edited: "29-Sep-85 17:52")
    (if (NOT n)
        then n_ (ARRAYSIZE array1))
    (SELECTQ pattern
         ((alt altAll newRand))
         (fixedRand (SetupTestArray array1 'rand n))
         (SetupTestArray array1 pattern n))
    (BUS.RESET)
    (BUSDMA.INIT)
    (PROG ((losses 0)
           (memFaults 0)
           (slowFaults 0)
           type true)
          (for j from 1 to (OR numPasses MAX.SMALLP) until (KEYDOWNP 'STOP)
             do (SELECTQ pattern
                      (newRand (SetupTestArray array1 pattern n))
                      (alt (SetupTestArray array1 (if (EVENP j)
                                                      then 'alt0
                                                    else 'alt1)
                                  n))
                      (altAll (SetupTestArray array1 (if (EVENP j)
                                                         then 0
                                                       else 65535)
                                  n))
                      NIL)
                (PCBUS.WRITEARRAY array1 0 n mode 1 page)
                (PCBUS.READARRAY array2 0 n mode 1 page)
                (for i from 1 to n unless (ELT array1 i) = (ELT array2 i)
                   do (bind prev do true_ (PCBUS.READWORD page i-1 mode)
                         repeatuntil (if true= (ELT array1 i)
                                         then (type_ 'r)
                                              (add losses 1)
                                       elseif prev
                                         then (if prev=true
                                                  then (type_ 'w)
                                                       (add memFaults 1)
                                                       (PCBUS.WRITEWORD page i-1 (ELT array1 i))
                                                else (add slowFaults 1)
                                                     (prev_true)
                                                     NIL)
                                       else (prev_true)
                                            NIL)))
                (PRIN1 "." T)
             finally (RETURN losses+memFaults+slowFaults])

)

(RPAQQ PCMEM.READTESTPATS (0 65535 rand))

(RPAQQ PCMEM.THRESH 15)

(RPAQQ PCMEM.WRITETESTPATS (altAll fixedRand))

(DEFINEQ
```

(PCMEM.MAKETEST

```
  [LAMBDA (array1 array2)                                        (* edited: "29-Sep-85 18:04")
    (LET ((mw (CREATEW NIL "PC Memory Test Window"))
          mwRight controlW controlM reg)
         [SETQ mwRight (PLUS (fetch LEFT of (WINDOWPROP mw (QUOTE REGION)))
                             (fetch WIDTH of (WINDOWPROP mw (QUOTE REGION]
         [WINDOWPROP mw (QUOTE BMTArrays)
                (LIST [OR array1 BMTArray1 (SETQ BMTArray1 (ARRAY 32768 (QUOTE WORD]
                      (OR array2 BMTArray2 (SETQ BMTArray2 (ARRAY 32768 (QUOTE WORD]
         (DSPSCROLL (QUOTE ON)
                mw)
         (WINDOWPROP mw (QUOTE ICON)
                BusmasterIcon)
         (WINDOWPROP mw (QUOTE BMTMenus)
                (bind (nw _ mw)
                      tm aw tw ttw for tms in BMTestTogMenuSpecs
                   collect (SETQ ttw (TogMenu (SETQ tm (MakeTogMenu (CDR tms)))
                                              (CAR tms)
                                              NIL 0 0 T))
                           (if (EQ (CAR tms)
                                   (QUOTE Pattern))
                               then
```

(* * widen it to take the biggest name in the out patterns)

```
                                    [SETQ reg (APPEND (WINDOWPROP ttw (QUOTE REGION]
                                    [replace WIDTH of reg with (WIDTHIFWINDOW (STRINGWIDTH "fixedRand"
                                                                              (DSPFONT NIL ttw]
                                    (SHAPEW ttw reg)
                                    (WINDOWPROP mw (QUOTE PatternMenu)
                                          tm))
                              (if (OR (NULL aw)
                                      (AND tw (IGREATERP [PLUS (fetch LEFT of (WINDOWPROP tw (QUOTE REGION)))
                                                               (fetch WIDTH of (WINDOWPROP tw (QUOTE REGION)))
                                                               (fetch WIDTH of (WINDOWPROP ttw (QUOTE REGION]
                                                         mwRight)))
                                  then (SETQ aw nw)
                                       (SETQ tw NIL)
                                       (SETQ nw ttw))
                              (if tw
                                  then (ATTACHWINDOW ttw tw (QUOTE RIGHT))
                                else (ATTACHWINDOW ttw aw (QUOTE TOP)
                                                  (QUOTE LEFT)))
                              (SETQ tw ttw)
                              (REDISPLAYW ttw)
                              tm))
        (ATTACHWINDOW (SETQ controlW (TogMenu [SETQ controlM (MakeTogMenu (QUOTE ((Start NIL (StartBltTest))
                                                                                  (Stop NIL (StopBltTest]
                                                "Control" NIL 0 0 T))
              mw
              (QUOTE RIGHT)
              (QUOTE TOP))
        (WINDOWPROP mw (QUOTE ControlMenu)
              controlM)
        (REDISPLAYW controlW)
        mw])
```

(BMTRead

```
  [LAMBDA (message)                                          (* edited: "12-Apr-85 11:05")
    (RESETFORM (TTYDISPLAYSTREAM (GETPROMPTWINDOW (MAINWINDOW $$TogWindow$$ T)))
           (printout NIL message ": ")
           (PROG1 (READ)
               (CLOSEW (GETPROMPTWINDOW (MAINWINDOW $$TogWindow$$ T)))))])
```

(BMTChangeDirection

```
  [LAMBDA (direction)                                        (* ht: "14-Apr-85 16:31")
    [if (BOUNDP (QUOTE $$TogWindow$$))
        then (TogMenuReset (WINDOWPROP (MAINWINDOW $$TogWindow$$ T)
                                       (QUOTE PatternMenu))
                           NIL
                           (SELECTQ direction
                                ((In FastIn)
                                    BMTInPatternSpecs)
                                ((Out FastOut)
                                    BMTOutPatternSpecs)
                                (SHOULDNT]
    direction])
```

(BMTSetValue

```
  [LAMBDA (var value)                                        (* edited: "12-Apr-85 10:48")
    [if (BOUNDP (QUOTE $$TogWindow$$))
        then (LET (proc)
                  (SETQ proc (WINDOWPROP (MAINWINDOW $$TogWindow$$ T)
                                         (QUOTE PROCESS)))
                  (if (AND proc (PROCESS.APPLY proc (FUNCTION BOUNDP)
                                               (LIST var)
                                               T))
                      then (PROCESS.APPLY proc (FUNCTION SET)
                                          (LIST var value]
    value])
```

(DoTB

```
  [NLAMBDA (fn args)                                         (* ht: "14-Apr-85 16:26")
    (RESETFORM (TTYDISPLAYSTREAM (PROCESSPROP (THIS.PROCESS)
                                              'WINDOW))
           (APPLY fn args))
    (TogMenuReset (WINDOWPROP (PROCESSPROP (THIS.PROCESS)
                                           'WINDOW)
                              'ControlMenu])
)

(RPAQQ BMTArray1 NIL)

(RPAQQ BMTArray2 NIL)

(RPAQ BMTInPatternSpecs '((zeros 0)
                          (ones 65535)
```

```
                          (alt0 NIL NIL "fixed pattern of alternating 0s and 1s, starting with 0")
                          (alt1 NIL NIL "fixed pattern of alternating 0s and 1s, starting with 1")
                          (rand NIL NIL "random words")
                          ((TogMenuValue)
                           (BMTRead "Pattern")
                           NIL "will prompt and read" Other)))

(RPAQ BMTOutPatternSpecs ' ((zeros 0)
                            (ones 65535)
                            (alt0 NIL NIL "fixed pattern of alternating 0s and 1s, starting with 0")
                            (alt1 NIL NIL "fixed pattern of alternating 0s and 1s, starting with 1")
                            (alt NIL NIL "0 -1 ... alternating every other pass with -1 0 ...")
                            (altAll NIL NIL "zeros alternating every other pass with ones")
                            (newRand NIL NIL "random words, new each pass")
                            (fixedRand NIL NIL "random words, same each pass")
                            ((TogMenuValue)
                             (BMTRead "Pattern")
                             NIL "will prompt and read" Other)))

(RPAQ BMTestTogMenuSpecs ' (("Direction" (In (BMTChangeDirection 'In))
                                         (Out (BMTChangeDirection 'Out))
                                         (FastIn (BMTChangeDirection 'FastIn))
                                         (FastOut (BMTChangeDirection 'FastOut)))
                            ("Mode" (Straight (NILL))
                                    (Swapped SWAP))
                            (Pattern)
                            ("Save results" (No (BMTSetValue 'save NIL))
                                    (Yes (BMTSetValue 'save T)))
                            ("Show every error" (No (BMTSetValue 'show NIL))
                                    (Yes (BMTSetValue 'show T)))
                            ("Summarize every" (1 (BMTSetValue 'sumEvery 1))
                                    (10 (BMTSetValue 'sumEvery 10))
                                    ((TogMenuValue)
                                     (BMTSetValue 'sumEvery (BMTRead "Summarize every"))
                                     NIL "will prompt and read" Other))
                            ("Type of summary" (%. (BMTSetValue 'sumType '.))
                                    (Full (BMTSetValue 'sumType 'Full)))
                            ("# passes" (Forever 2147483647)
                                    ((TogMenuValue)
                                     (BMTRead "# passes")
                                     NIL "will prompt and read" Other))
                            ("Size of block" 100 1000 5000 10000 32768 ((TogMenuValue)
                                                                        (BMTRead "Size of block")
                                                                        NIL "will prompt and read" Other))
                            ("Dismiss" (Yes (BMTSetValue 'block? T))
                                    (No (BMTSetValue 'block? NIL)
                                        NIL "Enable/disable blocking - No is dangerous!"))))

(PUTASSOC 'Pattern BMTInPatternSpecs BMTestTogMenuSpecs)

(RPAQ BusmasterIcon (READBITMAP))


(64 64 "OOOOOOOOOOOOOOOO" "OOOOOOOOOOOOOOOO" "LBBBBBBBBBBBBB@C" "LBBBBBBNCOOOOOHC" "L@HHHHHLKOOOOOHC"
    "L@HLIHHLKOOOOOHC" "LBBNCJBNCOOOOOHC" "LBBNCJBNCOOOOOHC" "L@HLIHHLKOOOOOHC" "L@HNKHHLKOOOOOHC"
    "LBBOOJBNCOOOOOHC" "LBBGOBBNCOOOOOHC" "L@HKNHHLHHIOHHHC" "L@HILHHLHHIOHHHC" "LBBCNBBNBBCOBB@C"
    "LBBCNBBNBBCOJB@C" "L@HKLHHLHHIOHHHC" "L@HILHHLHHIOHHHC" "LBBCNBBNBBCOBB@C" "LBBCNBBNBBCOBB@C"
    "L@HILHHLHHIOHHHC" "L@HILHHLHHIOHHHC" "LBBCNBBNBBCOBB@C" "LBBCNBBNBBCOBB@C" "L@HILHHLHHIOHHHC"
    "L@HILHHLHHIOHHHC" "LBBCNBBNBBCOBB@C" "LBBCNBBNBBCOBB@C" "L@HILHHLHHIOHHHC" "L@HILHHLHHIOHHHC"
    "LBBCNBBNBBCOBB@C" "LBBCNBBNBBCOBB@C" "L@HILHHLHHIOHHHC" "L@HILHINHHIOHHHC" "LBBCNBCOBBCOBB@C"
    "LBBCNBGOJBCOBB@C" "L@HILHOOHHIOHHHC" "L@HILHOOHHIOHHHC" "LBBCNBGOJBCOBB@C" "LBBCNBGOJBCOBB@C"
    "L@HILHOOHHIOHHHC" "L@HKNHOOHHIOHHHC" "LBBCOBGOJBCOBB@C" "LBBOOJGOJBCOBB@C" "L@HOOHOOHHIOHHHC"
    "L@HNKHOOHHIOHHHC" "LBBNCJGOJBCOBB@C" "LBBNCJGOJBCOBB@C" "L@HLIHKOHHIOHHHC" "L@HLIHINHHHHHHHC"
    "L@@@@@@@@@@@@@@C" "L@@@@@@@@@@@@@@C" "L@@@@@@@@@@@@@@C" "LO@O@B@H@@@@@@@C" "LHI@HCAHLEKAHJIC"
    "LHI@@CAIBFMBDNIC" "LOA@@BJINDIBDHEC" "LHA@HBJI@DIBDHFC" "LH@O@BDHNDIAHHBC" "L@@@@@@@@@@@@@BC"
    "L@@@@@@@@@@@@@LC" "L@@@@@@@@@@@@@@C" "L@@@@@@@@@@@@@@C" "OOOOOOOOOOOOOOOO")

(FILESLOAD (SYSLOAD)
       BUSMASTER)

(DECLARE: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY

(RESETSAVE DWIMIFYCOMPFLG T)

(DECLARE: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS BMTArray1 BMTArray2)
)

(FILESLOAD (LOADCOMP)
       BUSEXTENDER.DCOM)
)

(DECLARE: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS

(ADDTOVAR NLAMA )
```

```
(ADDTOVAR NLAML DoTB)

(ADDTOVAR LAMA )
)

(PUTPROPS PCMEMTEST COPYRIGHT ("Speech Input Project, Univ. of Edinburgh" 1985))
```

FUNCTION INDEX

BMTChangeDirection 9
BMTRead 9
BMTSetValue 9
DoTB 9
FastTestBltIn 6
PCMEM.CHECKOUT 2
PCMEM.MAKETEST 8
QuietTestBltIn 7
QuietTestBltOut 8
RecordError 2
SetupTestArray 2
ShowErrors 3
ShowResults 3
ShowWords 3
StartBltTest 3
StopBltTest 4
TestBltIn 4
TestBltOut 5

VARIABLE INDEX

BMTArray1 9
BMTArray2 9
BMTestTogMenuSpecs 10
BMTInPatternSpecs 9
BMTOutPatternSpecs 10
BusmasterIcon 10
PCMEM.READTESTPATS 8
PCMEM.THRESH 8
PCMEM.WRITETESTPATS 8