File created: 11-May-2023 21:39:25 **{DSK}<cygdrive>c>Users>Larry>home>il>MEDLEY>SOURCES>PMAP.;2**

edit by: lmm

changes to: (VARS PMAPCOMS)

previous date: 19-Jul-2022 23:17:41 {DSK}<cygdrive>c>Users>Larry>home>il>MEDLEY>SOURCES>PMAP.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
(RPAQQ PMAPCOMS
       (                                                    ; Page mapping primitives.  This file is shared with VAX.
        (FNS ADDMAPBUFFER \ALLOCMAPBUFFER CHECKBUFFERREFVAL \WRITEOUTBUFFERS \CLEARMAP DOPMAP FINDPTRSBUFFER
             FORGETPAGES \GETMAPBUFFER LOCKMAP MAPAFTERCLOSE MAPBUFFERCOUNT MAPPAGE MAPWORD \RELEASEBUFFER
             RELEASINGVMEMPAGE RESTOREMAP UNLOCKMAP \MAPPAGE \COLLECTDIRTYBUFS \SETIODIRTY)
        (FNS WORDCONTENTS SETWORDCONTENTS /SETWORDCONTENTS WORDOFFSET)
        (EXPORT (PROP BYTEMACRO WORDCONTENTS SETWORDCONTENTS WORDOFFSET))
        (COMS (ADDVARS (DEFAULTMAPFILE)
                       (SYSTEMBUFFERLIST)
                       (MAPEMPTYBUFFERLIST))
              (GLOBALVARS SYSTEMBUFFERLIST MAPEMPTYBUFFERLIST DEFAULTMAPFILE))
        [COMS                                               ; Functions for page-mapped devices
              (DECLARE%: DONTCOPY (EXPORT (MACROS \RELEASECPAGE)))
              (FNS \MAKE.PMAP.DEVICE \PAGEDBACKFILEPTR \PAGEDSETFILEPTR \PAGED.INCFILEPTR \PAGEDGETFILEPTR
                   \PAGEDGETEOFPTR \PAGEDREADP \PAGEDEOFP \PAGED.GETNEXTBUFFER \PAGED.FORCEOUTPUT \UPDATEOF
                   \READPAGES \WRITEPAGES)
              (FNS \SETEOF \PAGED.SETEOFPTR \NEWLENGTHIS)
              (DECLARE%: DONTEVAL@LOAD DOCOPY               ; For TEXTOFD
                     (P (PUTD '\PAGEDBIN (GETD '\BUFFERED.BIN)
                              T)
                        (PUTD '\PAGEDPEEKBIN (GETD '\BUFFERED.PEEKBIN)
                              T]
        (FNS PPBUFS)
        (DECLARE%: DONTCOPY (RECORDS BUFFER)
               EVAL@COMPILE
               (MACROS GETBUFFERPTR CHECKBUFFERREF CPBUFFERP BUFFERINUSEP UNDIRTY DIRTYP)
               (I.S.OPRS INBUFS))
        (INITRECORDS BUFFER)
        (LOCALVARS . T)))

;; Page mapping primitives.  This file is shared with VAX.

(DEFINEQ

(ADDMAPBUFFER
  [LAMBDA (TEMP ERRORFLG)                                   (* rrb "16-DEC-79 15:54")
    ;; old entry left arond for compatibility
    NIL])

(\ALLOCMAPBUFFER
  [LAMBDA NIL                                               (* lmm "10-MAR-83 23:19")
    ;; allocates a new buffer.  The new buffer will be put on SYSTEMBUFFERLIST which is used by the GC when releasing a buffer.
                                                            ; This should be the only function that creates BUFFERs.
    (SETQ SYSTEMBUFFERLIST (create BUFFER
                                   VMEMPAGE _ (NCREATE 'VMEMPAGEP)
                                   SYSNEXT _ SYSTEMBUFFERLIST])

(CHECKBUFFERREFVAL
  [LAMBDA (BUFF)                                            (* lmm "10-MAR-83 23:23")
    ;; checks the reference bit of a buffer descriptor and sets it if it is off.  Also returns the value of the buffer page ptr so that it will be on the stack and
    ;; therefore not be reset if a gc occurs.
    (UNINTERRUPTABLY
        (COND
           ((fetch NOREFERENCE of BUFF)
            (\DELREF (fetch VMEMPAGE of BUFF))
            (replace NOREFERENCE of BUFF with NIL)))
        (fetch VMEMPAGE of BUFF))])

(\WRITEOUTBUFFERS
  [LAMBDA (BUFFER STREAM)                                   (* bvm%: "16-May-84 14:32")
    ;; writes the contents of a buffer back out to the file they are mapped from.  BUFFER can be a single buffer or a list of buffers containing ascending
    ;; contiguous pages
    (COND
       ((LISTP BUFFER)
        (\WRITEPAGES STREAM (fetch FILEPAGE# of (CAR BUFFER))
               (for BUF in BUFFER collect (CHECKBUFFERREFVAL BUF)))
        (for BUF in BUFFER do (UNDIRTY BUF STREAM)))
```

```
       (T (\WRITEPAGES STREAM (fetch FILEPAGE# of BUFFER)
                 (CHECKBUFFERREFVAL BUFFER))                    ; reset dirty bit.
          (UNDIRTY BUFFER STREAM])
```

(\CLEARMAP

```
[LAMBDA (STREAM PAGES USERFLG)                                  ; Edited 12-Jul-88 13:53 by bvm
```

;; clears pages from an ofd writing them out if they are dirty. PAGES is a page# or a list of page#s or NIL. USERFLG is T for user calls and if ;; PAGES is NIL, causes all usermapped pages to get written out.

```
(COND
   ((DIRTYABLE STREAM)
```

;; first write out any buffers that are dirty.

```
    (FDEVOP 'FORCEOUTPUT (fetch DEVICE of STREAM)
            STREAM)))
(if (NULL PAGES)
    then (UNINTERRUPTABLY
```

;; Since we're about to throw the buffers away, flush the current page. In the case of output stream, the forceoutput method ;; already did this with a \releasecpage

```
            (replace CBUFSIZE of STREAM with 0)
            (replace CBUFPTR of STREAM with NIL)))
(PROG ((BUFFER (fetch BUFFS of STREAM))
       PREVBUFFER)
  LP   (COND
          ((NULL BUFFER)
           (RETURN))
          ((COND
              ((NULL PAGES)
               (COND
                  (USERFLG                                      ; User is asking for all mapped pages to be cleared, Is this a
                                                                ; usermapped page?
                          (fetch USERMAPPED of BUFFER))
                  (T                                            ; system call, clear all pages
                     T)))
              ((NLISTP PAGES)
               (EQ PAGES (fetch FILEPAGE# of BUFFER)))
              ((FMEMB (fetch FILEPAGE# of BUFFER)
                      PAGES)))                                  ; found a page to clear.
```

;; this may cause extra IO system buffers to get unallocated. this is ok in that they will get reallocated up to the standard number but ;; not ok in that if the file was opened specifying more that the standard number, the extras will get lost.

```
           (\RELEASEBUFFER (PROG1 BUFFER
                                  [COND
                                     [PREVBUFFER                ; This isn't the first buffer on list.
                                            (replace BUFFERNEXT of PREVBUFFER with (SETQ BUFFER
                                                                                          (fetch BUFFERNEXT of BUFFER]
                                     (T                         ; deleting the first buffer, change the STREAM
                                        (replace BUFFS of STREAM with (SETQ BUFFER (fetch BUFFERNEXT of BUFFER])
                                  STREAM)
           (GO LP))
          (T (SETQ PREVBUFFER BUFFER)
             (SETQ BUFFER (fetch BUFFERNEXT of BUFFER))
             (GO LP])
```

(DOPMAP

```
[LAMBDA (PAGE# STREAM VMEMPAGE)                                 (* rmk%: "25-OCT-83 19:57")
```

;; reads a page from a file into a block of storage. If the protection bits are ever implemented in hardware, this should set them from a new ;; argument.

```
(\READPAGES STREAM PAGE# VMEMPAGE)
```

;; We return the page pointer to ensure that it remains on the stack to guard against inclement garbage collections

```
VMEMPAGE])
```

(FINDPTRSBUFFER

```
[LAMBDA (PTR NOERRORFLG)                                        (* lmm "10-MAR-83 23:20")
```

;; given a pointer to a mapped location, return the buffer which contains that pointer. Causes error if no such buffer (thus this is used as a checking ;; function too)

```
(COND
   [(bind (B _ SYSTEMBUFFERLIST) while B do (COND
                                               ((EQ PTR (fetch VMEMPAGE of B))
                                                (RETURN B))
                                               (T (SETQ B (fetch SYSNEXT of B]
   (NOERRORFLG NIL)
   (T (ERROR PTR "not a MAPPAGE pointer"])
```

(FORGETPAGES

```
[LAMBDA (STREAM FROMPAGE TOPAGE)                                (* bvm%: "12-NOV-83 16:51")
```

;; cleans pages out of the map. Used only by truncate file to throw away any trancated pages that might be mapped. Pages FROMPAGE to ;; TOPAGE inclusive are forgotten. If FROMPAGE is NIL uses 0, if TOPAGE is NIL, uses last page.

```
    (COND
       ((OR (NULL TOPAGE)
            (NULL FROMPAGE)
            (IGEQ TOPAGE FROMPAGE))
        (PROG (REFFED (BUFFER (fetch BUFFS of STREAM))
                      PREVBUFFER)
          LP  (COND
                 ((NULL BUFFER)
                  (RETURN REFFED))
                 ((AND (OR (NULL FROMPAGE)
                           (IGEQ (fetch FILEPAGE# of BUFFER)
                                 FROMPAGE))
                       (OR (NULL TOPAGE)
                           (ILEQ (fetch FILEPAGE# of BUFFER)
                                 TOPAGE)))                    ; this is a BUFFER to process
                  [COND
                     ((BUFFERINUSEP BUFFER STREAM)            ; if buffer is still referenced, note to return that fact.
                      (SETQ REFFED (CONS (fetch FILEPAGE# of BUFFER)
                                         REFFED]
                  [COND
                     (PREVBUFFER (replace BUFFERNEXT of PREVBUFFER with (fetch BUFFERNEXT of BUFFER)))
                     (T (replace BUFFS of STREAM with (fetch BUFFERNEXT of BUFFER]
                                                              ; MAPOUTBUFFER changes the structure of BUFFER so make
                                                              ; change to BUFFS before it is called.
                  (\RELEASEBUFFER BUFFER STREAM)
                  [SETQ BUFFER (COND
                                  (PREVBUFFER (fetch BUFFERNEXT of PREVBUFFER))
                                  (T (fetch BUFFS of STREAM]
                  (GO LP))
                 (T (SETQ PREVBUFFER BUFFER)
                    (SETQ BUFFER (fetch BUFFERNEXT of BUFFER))
                    (GO LP])
```

(\GETMAPBUFFER

```
  [LAMBDA NIL                                                 (* bvm%: "12-NOV-83 16:54")
```

;; gets a map buffer from the free list or creates a new one. Some of the ones on the free list may still be referenced by user structure and hence
;; can't be used. the reference counts will actually be behind the real ones because PMAPs only gets updated when a garbage collection occurs.
;; A possible strategy before allocating a new one or if none can be allocated is to force a garbage collection.

```
    (COND
       [(AND MAPEMPTYBUFFERLIST (COND
                                   [(OR (NOT (fetch USERMAPPED of MAPEMPTYBUFFERLIST))
                                        (fetch NOREFERENCE of MAPEMPTYBUFFERLIST))
                                                              ; is first empty buffer unreferenced or has it never been user
                                                              ; mapped?
                                    (replace IODIRTY of MAPEMPTYBUFFERLIST with NIL)
                                    (replace USERMAPPED of MAPEMPTYBUFFERLIST with NIL)
                                    (PROG1 MAPEMPTYBUFFERLIST
                                        (SETQ MAPEMPTYBUFFERLIST (fetch BUFFERNEXT of MAPEMPTYBUFFERLIST)))]
                                   (T                         ; find the first not referenced one and return it.
                                      (PROG ((PREV MAPEMPTYBUFFERLIST)
                                             (BUF MAPEMPTYBUFFERLIST))
                                        LP  (COND
                                               ((NULL (SETQ BUF (fetch BUFFERNEXT of BUF)))
                                                (RETURN NIL))
                                               ((OR (NOT (fetch USERMAPPED of BUF))
                                                    (fetch NOREFERENCE of BUF))
                                                              ; buffer is not referenced or was never user mapped.
                                                (replace BUFFERNEXT of PREV with (fetch BUFFERNEXT of BUF))
                                                (replace IODIRTY of BUF with NIL)
                                                (replace USERMAPPED of BUF with NIL)
                                                (RETURN BUF))
                                               (T (SETQ PREV BUF)
                                                  (GO LP]
       (T                                                     ; if there isn't one that's not referenced, create a new one.
          (\ALLOCMAPBUFFER])
```

(LOCKMAP

```
  [LAMBDA (PTR)                                               (* rrb "15-SEP-79 18:17")
```

;; is a noop on the dorado all buffers are locked until no longer referenced.

```
    PTR])
```

(MAPAFTERCLOSE

```
  [LAMBDA (STREAM)                                            (* rmk%: "25-OCT-83 20:08")
```

;; this function is called after closing a file.

```
    (\CLEARMAP STREAM)
    (AND DEFAULTMAPFILE (EQ STREAM (\GETSTREAM DEFAULTMAPFILE))
         (SETQ DEFAULTMAPFILE NIL])
```

(MAPBUFFERCOUNT

```
  [LAMBDA (AVAILFLG)                                          (* rrb " 2-JAN-80 15:47")
```

```
    ;; counts either the total number of buffers or the number available for use now.

    (bind (B _ SYSTEMBUFFERLIST) while B count (PROG1 (OR (NOT AVAILFLG)
                                                          (fetch NOREFERENCE of B)
                                                          (NOT (fetch USERMAPPED of B)))
                                                      (SETQ B (fetch SYSNEXT of B)))])
```

(MAPPAGE

```
  [LAMBDA (PAGE# FILE READONLY)                                    (* rmk%: "25-OCT-83 19:55")

    ;; establishes a buffer for a page of a file and (since semantics of 10 require it) checks to make sure file is open for reading.

    ;; must set the eof pointer if this page is past the current eof and the file is writable, unless user says READONLY in which case we don't guarantee
    ;; that (accidental) changes to the buffer will get saved in the file.

    (PROG ((STREAM (\GETSTREAM FILE)))
          (OR (fetch PAGEMAPPED of (fetch DEVICE of STREAM))
              (ERROR STREAM "not page-mappable"))
          (RETURN (SELECTQ (fetch ACCESS of STREAM)
                           (INPUT (\MAPPAGE PAGE# STREAM T))
                           (BOTH (PROG1 (\MAPPAGE PAGE# STREAM T)
                                        [OR READONLY (COND
                                                          ((ILEQ (fetch EPAGE of STREAM)
                                                                 PAGE#)

                                                              ;; user is mapping for write the last page or a page beyond the last one, set the EOF to the zeroth
                                                              ;; byte of the next page.  This assumes that BOUT keeps at least the page part of the EOF up to date
                                                              ;; with its output.

                                                              (\SETEOF STREAM (ADD1 PAGE#)
                                                                       0)))
                                        (ERROR STREAM "must be open for input to map."])
```

(MAPWORD

```
  [LAMBDA (FILEADR FILE)                                           (* lmm "10-MAR-83 23:33")

    ;; changed to contain dorado standard page size constants.

    (WORDOFFSET (MAPPAGE (FOLDLO FILEADR WORDSPERPAGE)
                         FILE)
                (MOD FILEADR BYTESPERPAGE])
```

(\RELEASEBUFFER

```
  [LAMBDA (BUFFER STREAM)                                          (* bvm%: "12-NOV-83 16:51")

    ;; releases a buffer by moving it from the STREAM to the free list.  it will not be taken off the free list if it is still referenced and it has been
    ;; usermapped.

    (replace BUFFERNEXT of BUFFER with MAPEMPTYBUFFERLIST)         ; put the buffer on the free list.
    (SETQ MAPEMPTYBUFFERLIST BUFFER])
```

(RELEASINGVMEMPAGE

```
  [LAMBDA (PTR)                                                    (* bvm%: "24-JUN-82 17:01")

    ;; this function is called by the garbage collector when it determines that PTR is a VMEMPAGE to which there are no pointers.  If this function
    ;; returns T, PTR will not be put on the free list.  This function checks to see if PTR is a buffer and if so, marks that buffer's descriptor as available.
    ;; If not, the user has created and used PTR so zero it before it goes onto free list.

    (COND
       ((SETQ PTR (FINDPTRSBUFFER PTR T))
        (replace NOREFERENCE of PTR with T)
        T])
```

(RESTOREMAP

```
  [LAMBDA (STREAM PAGES)                                           (* bvm%: "12-NOV-83 16:51")

    ;; This function is called by LOGOUT after it has returned on any file that has been found to be changed.  It remaps any pages that are referenced
    ;; (LOGOUT calls RECLAIM) and returns a list of their page numbers.

    (PROG ((STRM (\GETSTREAM STREAM))
           (BUFFER (fetch BUFFS of STREAM))
           PREVBUFFER REFFED)
      LP   [COND
              ((NULL BUFFER)
               (RETURN REFFED))
              ([OR (NULL PAGES)
                   (for P inside PAGES thereis (EQ P (fetch FILEPAGE# of BUFFER]

               ;; found a page to restore.  If page is not referenced, don't bother to remap it.  If it is referenced, map it and return its page number.

               (COND
                  ((BUFFERINUSEP BUFFER STRM)

                   ;; if r/w bits are ever made accessible to LISP, they should be gotten from the ofd and passed to DOPMAP.

                   (DOPMAP (fetch FILEPAGE# of BUFFER)
                           STRM
                           (fetch VMEMPAGE of BUFFER))
                   (SETQ REFFED (CONS (fetch FILEPAGE# of BUFFER)
                                      REFFED)))
                  (T
```

```
                    ;; this may cause extra IO system buffers to get unallocated.  this is ok in that they will get reallocated up to the standard
                    ;; number but not ok in that if the file was opened specifying more that the standard number, the extras will get lost.
                    (\RELEASEBUFFER (PROG1 BUFFER
                                           [COND
                                              [PREVBUFFER          ; This isn't the first buffer on list.
                                                     (replace BUFFERNEXT of PREVBUFFER
                                                        with (SETQ BUFFER (fetch BUFFERNEXT of BUFFER]
                                              (T                   ; deleting the first buffer, change the STRM
                                                 (replace BUFFS of STRM with (SETQ BUFFER (fetch BUFFERNEXT
                                                                                             of BUFFER])
                                           STRM)
                                    (GO LP]
                 (SETQ PREVBUFFER BUFFER)
                 (SETQ BUFFER (fetch BUFFERNEXT of BUFFER))
                 (GO LP])
```

(UNLOCKMAP

```
  [LAMBDA (PTR)                                                    (* rrb "15-SEP-79 18:18")
```

;; is a noop on the dorado all buffers are locked until no longer referenced.

```
  PTR])
```

(\MAPPAGE

```
  [LAMBDA (FILEPAGE# STREAM USERFLG)                               (* bvm%: "17-May-84 10:39")
```

;; maps a page of a file into a buffer. Assumes its arg is an STREAM and has been checked. Currently mapped pages are maintained in the
;; STREAM. The STREAM specifies a fixed number of buffers which are cycled through the sequential IO and more are added if the user calls
;; MAPPAGE. The oldest available buffer is used for the new page and more are allocated if none is available.

```
  (PROG ((BUF (fetch BUFFS of STREAM))
         %#IOBUFFS PREV PREVAVAIL MOREBUFS)
        [COND
           ((NULL BUF)                                             ; no buffers yet
            (SETQ BUF (\GETMAPBUFFER))
            (GO DOPMAP))
           ((EQ (fetch FILEPAGE# of BUF)
                FILEPAGE#)                                         ; if usermapped, then set bit in buffer.
            (COND
               (USERFLG (replace USERMAPPED of BUF with T)))
            (CHECKBUFFERREF BUF)                                   ; page is already on top
            (RETURN (fetch VMEMPAGE of BUF]
```

;; not on top -- walk thru the list, looking for the page and noting the last available buffer in case it is not found.

```
        (SETQ %#IOBUFFS (COND
                           ((fetch USERMAPPED of BUF)
                            0)
                           (T 1)))                                 ; Counts number of non-usermapped buffers
        (SETQ PREV BUF)
     LP [COND
           ((NULL (SETQ BUF (fetch BUFFERNEXT of BUF)))            ; not found
            [COND
               ((OR (NULL PREVAVAIL)
                    (ILEQ %#IOBUFFS (fetch MAXBUFFERS of STREAM)))
                                                                   ; Fewer than the specified max exist so far, so create a new
                                                                   ; buffer
                (SETQ BUF (\GETMAPBUFFER)))
               (T (SETQ BUF (fetch BUFFERNEXT of PREVAVAIL))       ; write out the old buffer if necessary and remove it from its place
                                                                   ; in the list
                  (COND
                     ((AND (DIRTYABLE STREAM)
                           (OR (fetch USERMAPPED of BUF)
                               (DIRTYP BUF STREAM)))
                      (\WRITEOUTBUFFERS (COND
                                           ((AND (fetch MULTIBUFFERHINT of STREAM)
                                                 (SETQ MOREBUFS (\COLLECTDIRTYBUFS (fetch FILEPAGE#
                                                                                      of BUF)
                                                                                   STREAM)))
                                                                   ; This device likes multiple buffers, so write out as much as we
                                                                   ; can
                                            (CONS BUF MOREBUFS))
                                           (T BUF))
                                        STREAM)))
                  (replace BUFFERNEXT of PREVAVAIL with (fetch BUFFERNEXT of BUF]
                                                                   ; BUF is not a buffer to be used.  If interrupted here a buffer
                                                                   ; could get dropped.
            (GO DOPMAP))
           ((EQ (fetch FILEPAGE# of BUF)
                FILEPAGE#)                                         ; found the page, move it to front.
            (CHECKBUFFERREF BUF)
            (UNINTERRUPTABLY
                (replace BUFFERNEXT of PREV with (fetch BUFFERNEXT of BUF))
                (replace BUFFERNEXT of BUF with (fetch BUFFS of STREAM))
                (replace BUFFS of STREAM with BUF))
            (RETURN (GETBUFFERPTR BUF)))
           ((OR (NULL (fetch USERMAPPED of BUF))
                (fetch NOREFERENCE of BUF))                        ; BUF is available
```

```
            (SETQ PREVAVAIL PREV)
            (SETQ %#IOBUFFS (ADD1 %#IOBUFFS]                    ; advance to next buffer on list.
        (SETQ PREV BUF)
        (GO LP)
    DOPMAP
        (RETURN (PROG1 (DOPMAP FILEPAGE# STREAM (CHECKBUFFERREFVAL BUF))
                                                                ; PROG1 holds page pointer
                    (replace FILEPAGE# of BUF with FILEPAGE#)
                    (replace BUFFERNEXT of BUF with (fetch BUFFS of STREAM))
                                                                ; move to front of buffer list
                    (replace BUFFS of STREAM with BUF)
                    (replace USERMAPPED of BUF with USERFLG))])
```

(\COLLECTDIRTYBUFS

```
  [LAMBDA (FIRSTPAGE STREAM)                                    (* bvm%: "16-May-84 14:38")
```

;;; Returns a list of buffers that contain contiguously ascending dirty pages in STREAM immediately beyond FIRSTPAGE

```
    (bind NEXTBUF (LASTPAGE _ (ADD1 FIRSTPAGE)) while [SETQ NEXTBUF
                                                       (find B inbufs (fetch BUFFS of STREAM)
                                                          suchthat (AND (EQ (fetch FILEPAGE# of B)
                                                                            LASTPAGE)
                                                                        (OR (fetch USERMAPPED of B)
                                                                            (DIRTYP B STREAM]
       collect (PROGN (add LASTPAGE 1)
                      NEXTBUF])
```

(\SETIODIRTY

```
  [LAMBDA (STREAM PAGENUMBER)                                   (* rmk%: "25-OCT-83 20:00")
```

;; marks a buffer descriptor as dirty.

```
    (for BUF inbufs (fetch BUFFS of STREAM) when (EQ (fetch FILEPAGE# of BUF)
                                                     PAGENUMBER)
       do (replace IODIRTY of BUF with T)
          (RETURN BUF)
       finally (SHOULDNT)                                       ; It better be there somewhere
              ])

)

(DEFINEQ
```

(WORDCONTENTS

```
  [LAMBDA (PTR)                                                 (* lmm "28-FEB-82 23:24")
    (CHECK (FINDPTRSBUFFER PTR T))
    (\GETBASE PTR 0])
```

(SETWORDCONTENTS

```
  [LAMBDA (PTR N)                                               (* lmm "28-FEB-82 23:21")
```

;; stores into a word in a buffer. Does error checking which is not done by macro.

```
    (OR (FINDPTRSBUFFER PTR T)
        (ERROR PTR "not a PMAP buffer."))
    (\PUTBASE PTR 0 N])
```

(/SETWORDCONTENTS

```
  [LAMBDA (PTR N)                                               (* lmm "18-SEP-78 00:26")
    [AND LISPXHIST (UNDOSAVE (LIST (FUNCTION /SETWORDCONTENTS)
                                   PTR
                                   (WORDCONTENTS PTR]
    (SETWORDCONTENTS PTR N])
```

(WORDOFFSET

```
  [LAMBDA (PTR N)                                               (* lmm "28-FEB-82 23:22")
    (CHECK (FINDPTRSBUFFER PTR T))
    (\ADDBASE PTR N])

)

;; FOLLOWING DEFINITIONS EXPORTED

(PUTPROPS WORDCONTENTS BYTEMACRO ((PTR)
                                  (\GETBASE PTR 0)))

(PUTPROPS SETWORDCONTENTS BYTEMACRO ((PTR N)
                                     (\PUTBASE PTR 0 N)))

(PUTPROPS WORDOFFSET BYTEMACRO ((PTR N)
                                (\ADDBASE PTR N)))

;; END EXPORTED DEFINITIONS
```

```
(ADDTOVAR DEFAULTMAPFILE )

(ADDTOVAR SYSTEMBUFFERLIST )

(ADDTOVAR MAPEMPTYBUFFERLIST )

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS SYSTEMBUFFERLIST MAPEMPTYBUFFERLIST DEFAULTMAPFILE)
)
```

;; Functions for page-mapped devices

```
(DECLARE%: DONTCOPY
```

;; FOLLOWING DEFINITIONS EXPORTED

```
(DECLARE%: EVAL@COMPILE

(PUTPROPS \RELEASECPAGE MACRO ((STREAM)
                               (PROGN                          ; Must be under an UNINTERRUPTABLY !
                                      (COND
                                         ((fetch CBUFDIRTY of STREAM)
                                          (\SETIODIRTY STREAM (fetch CPAGE of STREAM))
                                          (replace CBUFDIRTY of STREAM with NIL)))
                                      (replace CBUFSIZE of STREAM with 0)
                                      (replace CBUFPTR of STREAM with NIL))))
)
)
```

;; END EXPORTED DEFINITIONS

```
(DEFINEQ
```

(\MAKE.PMAP.DEVICE

```
  [LAMBDA (DEVICE)                                    (* bvm%: "10-Jul-84 13:54")
```

;;; Installs the device ops needed to make DEVICE a pagemapped device. Returns DEVICE

```
    [with FDEV DEVICE (SETQ FDBINABLE T)
          (SETQ FDBOUTABLE T)
          (SETQ FDEXTENDABLE T)
          (SETQ RESETABLE T)
          (SETQ RANDOMACCESSP T)
          (SETQ PAGEMAPPED T)
          (SETQ BUFFERED T)
          (SETQ BIN (FUNCTION \BUFFERED.BIN))
          (SETQ BOUT (FUNCTION \BUFFERED.BOUT))
          (SETQ PEEKBIN (FUNCTION \BUFFERED.PEEKBIN))
          (SETQ READP (FUNCTION \PAGEDREADP))
          (SETQ BACKFILEPTR (FUNCTION \PAGEDBACKFILEPTR))
          (SETQ SETFILEPTR (FUNCTION \PAGEDSETFILEPTR))
          (SETQ GETFILEPTR (FUNCTION \PAGEDGETFILEPTR))
          (SETQ GETEOFPTR (FUNCTION \PAGEDGETEOFPTR))
          (SETQ SETEOFPTR (FUNCTION \PAGED.SETEOFPTR))
          (SETQ EOFP (FUNCTION \PAGEDEOFP))
          (SETQ BLOCKIN (FUNCTION \BUFFERED.BINS))
          (SETQ BLOCKOUT (FUNCTION \BUFFERED.BOUTS))
          (SETQ GETNEXTBUFFER (FUNCTION \PAGED.GETNEXTBUFFER))
          (COND
             ((OR (NULL FORCEOUTPUT)
                  (EQ FORCEOUTPUT (FUNCTION NILL)))
              (SETQ FORCEOUTPUT (FUNCTION \PAGED.FORCEOUTPUT]
    DEVICE])
```

(\PAGEDBACKFILEPTR

```
  [LAMBDA (STREAM)                                    ; Edited 13-Jun-2021 11:21 by rmk:
                                                      ; also see similar function \DRIBBACKFILEPTR

    [COND
       ((APPENDONLY STREAM)
        (LISPERROR "ILLEGAL ARG" (fetch (STREAM FULLNAME) of STREAM]
                                                      ; Checks done separately so we dont take an error with
                                                      ; interrupts off
    (\UPDATEOF STREAM)
    (COND
       ((NOT (AND (EQ (fetch (STREAM COFFSET) of STREAM)
                      0)
                  (EQ (fetch (STREAM CPAGE) of STREAM)
                      0)))
        (UNINTERRUPTABLY
            [replace (STREAM COFFSET) of STREAM with (COND
                                                        ((EQ (fetch (STREAM COFFSET) of STREAM)
                                                             0)
                                                         (\RELEASECPAGE STREAM)
```

```
                                                    (add (fetch (STREAM CPAGE) of STREAM)
                                                         -1)
                                                    (SUB1 BYTESPERPAGE))
                                                   (T (SUB1 (fetch (STREAM COFFSET) of STREAM]
          [replace (STREAM CHARPOSITION) of STREAM with (IMAX 0 (SUB1 (fetch (STREAM CHARPOSITION) of STREAM]))])
```

(\PAGEDSETFILEPTR

```
[LAMBDA (STREAM INDX)                                   ; Edited 24-Jun-87 18:18 by bvm:
  (\UPDATEOF STREAM)                                    ; Update the EOF in case we have writen thru it
  (PROG ((NEWPAGE (fetch (BYTEPTR PAGE) of INDX))
         (NEWOFF (fetch (BYTEPTR OFFSET) of INDX)))
        (UNINTERRUPTABLY
            (COND
               ([OR (NEQ NEWPAGE (fetch CPAGE of STREAM))
                    (AND (EQ NEWPAGE (fetch EPAGE of STREAM))
                         (> NEWOFF (fetch COFFSET of STREAM]
                ;; Force page release if (1) ptr is moving to a different page, (2) new ptr is past eof.  We permit setting ptr past eof--if the next op
                ;; is a BIN, an eof error occurs, while if the next op is a write, the end of file gets moved.  In order for this to work, we have the
                ;; convention that whenever CBUFPTR is non-nil, eof is the greater of the old eof or the current file pointer.

                ;; This clause also used to test for backing up on an APPEND-only stream, but that's nonsense--we should probably prohibit it
                ;; altogether.

                (\RELEASECPAGE STREAM)
                (replace CPAGE of STREAM with NEWPAGE)))
            (replace COFFSET of STREAM with NEWOFF))])
```

(\PAGED.INCFILEPTR

```
[LAMBDA (STREAM AMOUNT)                                 ; Edited 29-Feb-88 17:22 by bvm
  ;; Increment file pointer of stream by AMOUNT, which may be negative.  The only reason this function currently exists is to give fast performance to
  ;; FFILEPOS -- it avoids the boxing that would occur on large file pointers.
  (UNINTERRUPTABLY
      (PROG ((NEWOFF (+ (fetch COFFSET of STREAM)
                        AMOUNT))
             (NEWPAGE (fetch CPAGE of STREAM)))
       ;; SETFILEPTR sets CHARPOSITION to zero, but callers of \INCFILEPTR don't care, by fiat
            (COND
               ((>= NEWOFF BYTESPERPAGE)                ; New page
                (SETQ NEWPAGE (+ NEWPAGE (fetch (BYTEPTR PAGE) of NEWOFF)))
                (SETQ NEWOFF (fetch (BYTEPTR OFFSET) of NEWOFF)))
               [(< NEWOFF 0)                            ; New page going backward
                (SETQ NEWPAGE (- NEWPAGE (fetch (BYTEPTR PAGE) of (SETQ NEWOFF (SUB1 (- BYTESPERPAGE NEWOFF]
                (COND
                   ((< NEWPAGE 0)                       ; Probably shouldn't happen;  should it be an error?
                    (SETQ NEWPAGE 0)))
                (SETQ NEWOFF (SUB1 (- BYTESPERPAGE (fetch (BYTEPTR OFFSET) of NEWOFF]
               ([COND
                   ((< AMOUNT 0)                        ; Backing up, may have to set the eof if we have been writing
                    (\UPDATEOF STREAM)
                    T)
                   (T                                   ; Moving forward, make sure we don't move past the eof
                      (AND (fetch CBUFPTR of STREAM)
                           (<= NEWOFF (fetch CBUFSIZE of STREAM]
                                                        ; easy case, no page turn
                (replace COFFSET of STREAM with NEWOFF) ; Just bump COFFSET and we're done
                (RETURN))
               (T                                       ; Moving forward past eof, might as well let this fall thru to
                                                        ; general case, since we need to make sure current buffer is
                                                        ; released.
                  ))
            (\UPDATEOF STREAM)
            (\RELEASECPAGE STREAM)
            (replace CPAGE of STREAM with NEWPAGE)
            (replace COFFSET of STREAM with NEWOFF)))])
```

(\PAGEDGETFILEPTR

```
[LAMBDA (STREAM)                                        (* rmk%: " 2-JUL-82 13:07")
  (create BYTEPTR
          PAGE _ (fetch CPAGE of STREAM)
          OFFSET _ (fetch COFFSET of STREAM])
```

(\PAGEDGETEOFPTR

```
[LAMBDA (STREAM)                                        (* bvm%: "26-DEC-81 15:48")
  (\UPDATEOF STREAM)                                    ; If we have been writing the EOF may not be current
  (create BYTEPTR
          PAGE _ (fetch EPAGE of STREAM)
          OFFSET _ (fetch EOFFSET of STREAM])
```

(\PAGEDREADP

```
[LAMBDA (STREAM FLG)                                    ; Edited 19-Jul-2022 23:17 by rmk
                                                        ; Edited  7-Aug-2021 12:45 by rmk:
```

```
;; If FLG is NIL, a single EOL as the last character of the file doesn't count.  This is a character operation, not a byte operation.
(AND (NOT (\PAGEDEOFP STREAM))
     (OR (NOT (NULL FLG))
         (NEQ EOL.TC (\SYNCODE \PRIMTERMSA (\PEEKCCODE.EOLC STREAM)))
         (PROGN ;; We peeked at an EOL.  Is there anything beyond that?
                (OR (ILESSP (ffetch CPAGE of STREAM)
                            (ffetch EPAGE of STREAM))
                    (PROGN ;; Yes if we aren't on the last page.
                           ;; If on the last page, we know we are not at the end, because the just-peeked EOL is there.  But we also don't
                           ;; know how many bytes the EOL occupied. So at this point we have to read the EOL, check to see if we are
                           ;; then at the EOF, and then back out the EOL
                           (\INCCODE.EOLC STREAM)
                           (PROG1 (NOT (\PAGEDEOFP STREAM))
                                  (\BACKCCODE.EOLC STREAM])
```

(\PAGEDEOFP

```
[LAMBDA (STREAM)                                                    ; Edited 15-Jun-87 15:06 by jds
```

;;; Determines if a paged file is at EOF.

```
    (OR (READONLY STREAM)
        (\UPDATEOF STREAM))
    (LET* [(CUROFFSET (fetch COFFSET of STREAM))
           (CURPAGE (IPLUS (fetch CPAGE of STREAM)
                           (COND
                              ((AND (fetch CBUFPTR of STREAM)
                                    (IEQP CUROFFSET (fetch CBUFSIZE of STREAM)))
                               (SETQ CUROFFSET 0)
                               1)
                              (T 0]
          ;; CURPAGE is current page, allowing for the fact that COFFSET can be at end of the prior page which is equivalent to being at 0 on the
          ;; next page.
          (COND
             ((IGREATERP CURPAGE (fetch EPAGE of STREAM))           ; We're on a page that's past the last one in the file.
              T)
             ((ILESSP CURPAGE (fetch EPAGE of STREAM))
              ;; Not on last page yet, so not eof.  Need to figure in the COFFSET because it is possible for COFFSET to be BYTESPERPAGE
              ;; before the page is turned
              NIL)
             ((IGEQ CUROFFSET (fetch EOFFSET of STREAM))           ; We're on the last page, so check the buffer offset.
              T])
```

(\PAGED.GETNEXTBUFFER

```
[LAMBDA (STREAM WHATFOR NOERRORFLG)                                 ; Edited 13-Jun-2021 11:24 by rmk:
    ;; Advances STREAM to a new page.  Leaves the current page pointer NIL as the new page may never be written, so must update eof.  Returns T
    ;; on success;  any other return is a value to use by \BIN
    (PROG ((CPAGE# (fetch (STREAM CPAGE) of STREAM))
           (COFF (fetch (STREAM COFFSET) of STREAM))
           EPAGE# BUF)
          [COND
             ((NOT (OPENED STREAM))
              (LISPERROR "FILE NOT OPEN" (fetch (STREAM FULLNAME) of STREAM]
          (COND
             ((AND (ILESSP COFF (SELECTQ WHATFOR
                                    (READ (fetch (STREAM CBUFSIZE) of STREAM))
                                    BYTESPERPAGE))
                   (fetch (STREAM CBUFPTR) of STREAM))           ; Is ok, why were we called?
              (RETURN T)))                                       ; Buffer exhausted or empty
          (UNINTERRUPTABLY                                       ; Clean up current page
              (\RELEASECPAGE STREAM)
              (if (EQ COFF BYTESPERPAGE)
                  then                                           ; Change to be first byte of next page instead of beyond last byte
                                                                 ; of previous page
                       (replace (STREAM COFFSET) of STREAM with (SETQ COFF 0))
                       (replace (STREAM CPAGE) of STREAM with (add CPAGE# 1))))
          [COND
             ([AND (IGEQ CPAGE# (SETQ EPAGE# (fetch (STREAM EPAGE) of STREAM)))
                   (OR (NOT (IEQP CPAGE# EPAGE#))
                       (IGEQ COFF (fetch (STREAM EOFFSET) of STREAM]
                                                                 ; Current file pointer is at or past end of file
              (SELECTQ WHATFOR
                   (READ (RETURN (AND (NULL NOERRORFLG)
                                      (\EOF.ACTION STREAM))))
                   (WRITE (UNINTERRUPTABLY
                              (replace (STREAM EPAGE) of STREAM with (SETQ EPAGE# CPAGE#))
                              (replace (STREAM EOFFSET) of STREAM with COFF)))
                   (SHOULDNT]
```

;; Now fill the buffer -- map in current page

```
          (SETQ BUF (\MAPPAGE CPAGE# STREAM))                    ; This is interruptable
          (UNINTERRUPTABLY
              (replace (STREAM CBUFSIZE) of STREAM with (COND
                                                          ((ILESSP CPAGE# EPAGE#)
                                                                  ; Full page
                                                           BYTESPERPAGE)
                                                          ((IEQP CPAGE# EPAGE#)
                                                                  ; Last page
                                                           (fetch (STREAM EOFFSET) of STREAM))
                                                          (T      ; Beyond EOF so no data
                                                             0)))
              (replace (STREAM CBUFMAXSIZE) of STREAM with BYTESPERPAGE)
              (replace (STREAM CBUFPTR) of STREAM with BUF))
          (RETURN T])
```

(**PAGED.FORCEOUTPUT**

```
  [LAMBDA (STREAM)                                                (* bvm%: "22-Aug-84 12:44")
```

;; Flushes the contents of any dirty pages back into the file but leaves them available to LISP. As there is no way to know whether or not a
;; usermapped page has been changed, such pages will be written out again when the ofd is closed.

```
    (SETQ STREAM (\GETSTREAM STREAM 'OUTPUT))
    (COND
       ((DIRTYABLE STREAM)
        (\UPDATEOF STREAM)
        (UNINTERRUPTABLY
            (\RELEASECPAGE STREAM))
        (PROG [(BUFFERS (SORT (for B inbufs (fetch BUFFS of STREAM) when (OR (fetch USERMAPPED of B)
                                                                            (DIRTYP B))
                                  collect B)
                              (FUNCTION (LAMBDA (X Y)
                                        (IGREATERP (fetch FILEPAGE# of Y)
                                                   (fetch FILEPAGE# of X]
                                                                  ; Write out any dirty pages, in ascending order.
              (while BUFFERS do (\WRITEOUTBUFFERS (PROG1 BUFFERS; Write out as many contiguous ones as possible
                                                         (bind (B _ BUFFERS)
                                                               (N _ (fetch FILEPAGE# of (CAR BUFFERS)))
                                                            while (AND (CDR B)
                                                                       (EQ (fetch FILEPAGE# of (CADR B))
                                                                           (ADD1 N)))
                                                            do (SETQ B (CDR B))
                                                               (add N 1)
                                                            finally (SETQ BUFFERS (CDR B))
                                                                    (RPLACD B NIL)))
                                                  STREAM)))
              (\TRUNCATEFILE STREAM)                              ; Adjusts length on device

        ))
    STREAM])
```

(**UPDATEOF**

```
  [LAMBDA (STREAM)                                                (* bvm%: " 7-Jun-84 16:53")
```

;; The EOF needs updating if we have written past the EOF. We check CBUFPTR to detect phony file positions from SETFILEPTR and
;; TURNPAGE that were never actually written thru

```
    (AND (fetch CBUFPTR of STREAM)
         (PROGN
```

;; Determines if the current file ptr is BEYOND the end of file. Since page is loaded, we can test against the CBUFSIZE. As we
;; are ignoring the equal case, we dont need the test for page numbers used by FASTEOF.

```
                (IGREATERP (fetch COFFSET of STREAM)
                           (fetch CBUFSIZE of STREAM)))
         (\SETEOF STREAM (fetch CPAGE of STREAM)
                 (fetch COFFSET of STREAM])
```

(**READPAGES**

```
  [LAMBDA (STREAM FIRSTPAGE BUFFERLIST)                           (* bvm%: "26-DEC-81 15:44")
```

;; Read data from the file specified by open file descriptor OFD, starting with FIRSTPAGE into the buffers given in BUFFERLIST. If BUFFERLIST
;; is not a list, then it is assumed to be a pointer to a buffer into which a single page is read.

```
    (FDEVOP 'READPAGES (fetch DEVICE of STREAM)
           STREAM FIRSTPAGE BUFFERLIST])
```

(**WRITEPAGES**

```
  [LAMBDA (STREAM FIRSTPAGE BUFFERLIST)                           (* bvm%: "26-DEC-81 15:44")
```

;; Write data into the file specified by open file descriptor OFD, starting with FIRSTPAGE from the buffers given in BUFFERLIST. If BUFFERLIST
;; is not a list, then it is assumed to be a pointer to a buffer from which a single page is written.

```
    (\UPDATEOF STREAM)                                            ; Make EOF current
    (FDEVOP 'WRITEPAGES (fetch DEVICE of STREAM)
           STREAM FIRSTPAGE BUFFERLIST])

)

(DEFINEQ
```

(\SETEOF

```
  [LAMBDA (STREAM EP EO)                                    (* bvm%: "30-Sep-84 15:12")
    ;; Sets the end of file.  If new end of file is on the current page, resets the character count if necessary.
    [COND
       ((IGEQ EO BYTESPERPAGE)
        (add EP (fetch (BYTEPTR PAGE) of EO))
        (SETQ EO (fetch (BYTEPTR OFFSET) of EO]
    (UNINTERRUPTABLY
        (replace EPAGE of STREAM with EP)
        (replace EOFFSET of STREAM with EO)
        (COND
           ((NULL (fetch CBUFPTR of STREAM))                ; nothing mapped, so no fuss

            )
           ((EQ EP (fetch CPAGE of STREAM))
            (replace CBUFSIZE of STREAM with EO))
           ((IGREATERP (fetch CPAGE of STREAM)
                   EP)
            (\RELEASECPAGE STREAM)                          ; Page no longer exists

            )
           (T ;; If there's a page mapped in, it must not be the last page now, so make sure its CBUFSIZE is maximal.  Otherwise we lose when EO
              ;; was 512
              (replace CBUFSIZE of STREAM with BYTESPERPAGE)))
       NIL)])
```

(\PAGED.SETEOFPTR

```
  [LAMBDA (STREAM NBYTES)                                   (* bvm%: "30-Oct-86 17:44")
    (LET ((NEWEP (fetch (BYTEPTR PAGE) of NBYTES))
          (NEWEO (fetch (BYTEPTR OFFSET) of NBYTES)))
         (SELECTQ (\NEWLENGTHIS STREAM NEWEP NEWEO)
             (SHORTER (COND
                          ((OVERWRITEABLE STREAM)
                           (FORGETPAGES STREAM (ADD1 NEWEP)
                                  (PROG1 (fetch EPAGE of STREAM)   ; Remember the old last page
                                         (\SETEOF STREAM NEWEP NEWEO)
                                                                   ; Shorten the OFD's view of the file

                                  ))
                           ;; FORGETPAGES tells PMAP to throw away the extra pages.  The \SETEOF is done first so that an interrupt will
                           ;; not leave STREAM pointing to old and possibly partially overwritten pages.
                           (\CLEARBYTES (\MAPPAGE NEWEP STREAM)
                                  NEWEO
                                  (- BYTESPERPAGE NEWEO))          ; Zero out the trailing fragment of the last page
                           (\SETIODIRTY STREAM NEWEP)              ; Note that its dirty
                           (\TRUNCATEFILE STREAM NEWEP NEWEO)      ; Shorten the real file
                           T)))
             (SAME                                                 ; Nothing to do
                   T)
             (LONGER (if (APPENDABLE STREAM)
                         then (\SETEOF STREAM NEWEP NEWEO)
                              T))
             (SHOULDNT])
```

(\NEWLENGTHIS

```
  [LAMBDA (STREAM PGE OFF)                                  (* bvm%: "13-Feb-85 23:32")
    ;; Computes whether PGE OFF pair is longer or shorter than the current end of file
    (\UPDATEOF STREAM)                                       ; Before comparing, make it current
    (PROG ((TMP (IDIFFERENCE (fetch EPAGE of STREAM)
                      PGE)))
          (RETURN (COND
                     ((ILESSP TMP 0)
                      'LONGER)
                     [(EQ TMP 0)
                      (SETQ TMP (IDIFFERENCE (fetch EOFFSET of STREAM)
                                             OFF))
                      (COND
                         ((ILESSP TMP 0)
                          'LONGER)
                         ((EQ TMP 0)
                          'SAME)
                         (T 'SHORTER]
                     (T 'SHORTER])

)

(DECLARE%: DONTEVAL@LOAD DOCOPY

(PUTD '\PAGEDBIN (GETD '\BUFFERED.BIN)
      T)
```

```
(PUTD '\PAGEDPEEKBIN (GETD '\BUFFERED.PEEKBIN)
      T)
)

(DEFINEQ

(PPBUFS
  [LAMBDA (BUF0)                                                    (* rmk%: " 7-APR-81 20:53")
                                                                    ; Displays a buffer chain
    (for B inbufs BUF0 do (printout T "[" (fetch FILEPAGE# of B)
                                  ": " B "] ")
       finally (TERPRI T])

)

(DECLARE%: DONTCOPY

(DECLARE%: EVAL@COMPILE

(DATATYPE BUFFER (FILEPAGE# (VMEMPAGE XPOINTER)
                        BUFFERNEXT SYSNEXT (NOREFERENCE FLAG)
                        (USERMAPPED FLAG)
                        (IODIRTY FLAG)))
)

(/DECLAREDATATYPE 'BUFFER '(POINTER XPOINTER POINTER POINTER FLAG FLAG FLAG)
       ;; ---field descriptor list elided by lister---
       '8)


(DECLARE%: EVAL@COMPILE

(PUTPROPS GETBUFFERPTR MACRO ((BUFF)
                              (fetch VMEMPAGE of BUFF)))

(PUTPROPS CHECKBUFFERREF MACRO [OPENLAMBDA (BUFF)                   (* bvm%: "24-JUN-82 17:03")
                                   ;; checks the reference field of a buffer descriptor and if no one is referencing it, it creates a reference
                                   ;; and changes the flag.  The flag is set by the garbage collector when there are no longer any
                                   ;; references to the buffer it describes.
                                   (UNINTERRUPTABLY
                                       (COND
                                          ((fetch NOREFERENCE of BUFF)
                                           ;; this is a page the reference to which has been dropped, zero its reference count before
                                           ;; returning it.
                                           (\DELREF (fetch VMEMPAGE of BUFF))
                                           (replace NOREFERENCE of BUFF with NIL)))])

(PUTPROPS CPBUFFERP MACRO ((BUFFER STREAM)
                           (EQ (fetch CBUFPTR of STREAM)
                               (fetch VMEMPAGE of BUFFER))))

(PUTPROPS BUFFERINUSEP MACRO [OPENLAMBDA (BUFFER STREAM)
                                 (AND (NULL (fetch NOREFERENCE of BUFFER))
                                      (OR (fetch USERMAPPED of BUFFER)
                                          (CPBUFFERP BUFFER STREAM])

(PUTPROPS UNDIRTY MACRO [OPENLAMBDA (BUFFER STREAM)
                           (replace IODIRTY of BUFFER with NIL)
                           (COND
                              ((CPBUFFERP BUFFER STREAM)
                               (replace CBUFDIRTY of STREAM with NIL])

(PUTPROPS DIRTYP MACRO [OPENLAMBDA (BUFFER STREAM)                  (* rmk%: "25-OCT-83 19:57")
                           ;; determines if this buffer has been dirtied by the IO system.  It can't determine if the user has done a putbase into
                           ;; the page if he got it from MAPPAGE.
                           (OR (fetch IODIRTY of BUFFER)
                               (AND STREAM (CPBUFFERP BUFFER STREAM)
                                    (fetch CBUFDIRTY of STREAM])
)

(DECLARE%: EVAL@COMPILE

[I.S.OPR 'INBUFS NIL '(first (SETQ I.V. BODY) by (fetch BUFFERNEXT of I.V.) until (NULL I.V.]
)
)

(/DECLAREDATATYPE 'BUFFER '(POINTER XPOINTER POINTER POINTER FLAG FLAG FLAG)
       ;; ---field descriptor list elided by lister---
       '8)

(DECLARE%: DOEVAL@COMPILE DONTCOPY
```

```
(LOCALVARS . T)
)
```

FUNCTION INDEX

MACRO INDEX

VARIABLE INDEX

I.S.OPR INDEX

RECORD INDEX