File created: 12-Oct-93 14:29:06 **{Pele:mv:envos}<LispCore>Sources>CLTL2>LLBASIC.;2**

changes to: (OPTIMIZERS \ATOMCELL GETPROPLIST SETPROPLIST)
(VARS LLBASICCOMS)
(FUNCTIONS ATOM)
(FNS COPYATOM)

previous date: 5-Aug-93 10:20:24 {Pele:mv:envos}<LispCore>Sources>LLBASIC.;35

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
;;
;; Copyright (c) 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1990, 1991, 1992, 1993 by Venue & Xerox Corporation.  All rights reserved.

(RPAQQ LLBASICCOMS
       ((FNS LISTP LITATOM FIXP SMALLP NLISTP ARRAYP FLOATP NUMBERP STACKP)
        (FUNCTIONS ATOM)
        (DECLARE%: DONTCOPY (EXPORT (MACROS CHECK \StatsZero \StatsAdd1 IPLUS16 SMALLPOSP SETXVAR SETQ.NOREF IEQ
                                           )
                                    (CONSTANTS WordsPerPage))
               (TEMPLATES SPREADAPPLY* SPREADAPPLY SETQ.NOREF))
        (DECLARE%: EVAL@COMPILE DONTCOPY (RECORDS FREELISTENTRY HASHENTRY))
        [COMS                                                    ; atoms
              (FNS GETTOPVAL SETTOPVAL FSETVAL \SETGLOBALVAL.UFN \SETFVAR.UFN GETPROPLIST \ATOMCELL SETPROPLIST)
              (COMS (MACROS \PROPCELL)
                    (OPTIMIZERS \ATOMCELL GETPROPLIST SETPROPLIST))
              (FNS \MKATOM \CREATE.SYMBOL \MKATOM.FULL \INITATOMPAGE)
              (FNS MAPATOMS ATOMHASH#PROBES)
              (COMS                                              ; For MAKEINIT & TeleRaid
                    ;; This code has one major shortcoming which will not normally turn up.  If the local and remote sysouts conflict in their
                    ;; package setups it is possible for this code to return symbols interned in what for the teleraid'ing machine would be the
                    ;; correct package, but for the remote machine is in fact incorrect.  This warrents a warning in the documentation.  The
                    ;; problem lies in the fact that you *cannot* uncopy a symbol correctly between two machines with incompatible package
                    ;; setups.  An example of such a situation would be where on one machine the package FOO inherits BAR, and on the other
                    ;; BAR is present directly in FOO.  BAR's package cell will be different in both cases.  Two solutions come to mind; both
                    ;; would break the VSAVEWORK feature.  The first would be to UNCOPY symbols into special "remote symbol" objects.  The
                    ;; second is to create uninterned symbols with the correct name and smash their package cell to be that of a correctly named
                    ;; package.  Both of these schemes would require special reading and printing code.

                    (MACROS READSYS.HAS.PACKAGES)
                    (VARS READSYS.PACKAGE.FROM.NAME READSYS.PACKAGE.FROM.INDEX)
                    (FNS INITATOMS COPYATOM UNCOPYATOM MAKE.LOCAL.ATOM SYMBOL.VALUE SYMBOL.PNAME SYMBOL.PACKAGE
                         OLD.FIND.SYMBOL LOOKUP-SYMBOL FIND.PACKAGE FIND.SYMBOL PACKAGE.NAME))
              (COMS                                              ; See \PNAMELIMIT comment below
                    (VARS (\PNAMELIMIT 255))
                    (INITVARS (\PNAMES.IN.BLOCKS?)))
              (COMS ;; Flag for the closure cache
                    (INITVARS (SI::*CLOSURE-CACHE-ENABLED*))
                    (GLOBALVARS SI::*CLOSURE-CACHE-ENABLED*))
              (FNS \DEFINEDP PUTD \PUTD GETD PUTDEFN GETDEFN)
              (FNS \STKMIN)
              (INITVARS (\OPSTACKEFFECT)
                     (\OPLENGTH))
              (GLOBALVARS \OPSTACKEFFECT \OPLENGTH)
              (VARS (COMPILEATPUTDFLG))
              (DECLARE%: DONTCOPY (EXPORT (RECORDS LITATOM CL:SYMBOL VALINDEX VCELL DEFINITIONCELL FNHEADER
                                                   PNAMECELL PACKAGEINDEX PNAMEBASE PNAMEINDEX)
                                          (RECORDS NEW-ATOM)
                                          (MACROS \DEFCELL \VALCELL \PNAMECELL)
                                          (MACROS \ATOMVALINDEX \ATOMDEFINDEX \ATOMPNAMEINDEX \ATOMPROPINDEX
                                                  \INDEXATOMPNAME \INDEXATOMVAL \INDEXATOMDEF \ATOMNUMBER)
                                          (GLOBALVARS \NxtPnByte \CurPnPage \NxtAtomPage \AtomFrLst
                                                 \OneCharAtomBase \PNAMES.IN.BLOCKS? \SCRATCHSTRING
                                                 COMPILEATPUTDFLG)
                                          (CONSTANTS (\PNAMELIMIT 255)
                                                 (\CharsPerPnPage 512))
                                          (CONSTANTS (\NEWATOM-PNAMEOFFSET 0)
                                                 (\NEWATOM-VALOFFSET 2)
                                                 (\NEWATOM-DEFOFFSET 4)
                                                 (\NEWATOM-PLISTOFFSET 6)
                                                 (\NEWATOM-TYPE# 21))
                                          ;; \PNAMELIMIT is exported but needs to also be a VARS on this file to get it copied.  Note that
                                          ;; both commands must be edited together.
                                          ;; \NEWATOM-xxxxOFFSET is word offset in NEWATOM .  -- '90/07/19 ON
                                          ))
              (DECLARE%: EVAL@COMPILE DONTCOPY (MACROS COMPUTE.ATOM.HASH ATOM.HASH.REPROBE)
                     (ADDVARS (DONTCOMPILEFNS INITATOMS COPYATOM UNCOPYATOM READATOM MAKE.LOCAL.ATOM
                                              SYMBOL.VALUE SYMBOL.PNAME SYMBOL.PACKAGE OLD.FIND.SYMBOL LOOKUP-SYMBOL
                                              FIND.PACKAGE FIND.SYMBOL PACKAGE.NAME GETDEFN PUTDEFN FSETVAL]
        (COMS                                                    ; for executing boot expressions when first run
```

```
            (FNS \RESETSYSTEMSTATE INITIALEVALQT SIMPLEPRINT)
            (GLOBALVARS RESETFORMS BOOTFILES))
      (COMS                                                    ; stats
            (FNS PAGEFAULTS \SETTOTALTIME \SERIALNUMBER))
      (COMS                                                    ; Fast functions for moving and clearing storage
            (FNS \BLT \MOVEBYTES \CLEARWORDS \CLEARBYTES \CLEARCELLS)
            (DECLARE%: EVAL@COMPILE DONTCOPY (MACROS .CLEARNWORDS.))
            (COMS                                              ; Obsolete
                  (DECLARE%: EVAL@COMPILE DONTCOPY (EXPORT (MACROS \MOVEWORDS)))
                  (FNS \MOVEWORDS \ZEROBYTES \ZEROWORDS)))
      (LOCALVARS . T)
      [DECLARE%: DONTCOPY                                      ; For MAKEINIT & TeleRaid
             (ADDVARS (INITVALUES (\AtomFrLst 0))
                    (INITPTRS (\OneCharAtomBase NIL)
                           (\SCRATCHSTRING))
                    (INEWCOMS (FNS FSETVAL SETPROPLIST PUTDEFN \BLT)
                           (FNS \MKATOM \CREATE.SYMBOL \INITATOMPAGE \MOVEBYTES \STKMIN)
                           (FNS COPYATOM INITATOMS))
                    (EXPANDMACROFNS SMALLPOSP COMPUTE.ATOM.HASH ATOM.HASH.REPROBE \DEFCELL \VALCELL \PNAMECELL
                           \PROPCELL \INDEXATOMPNAME)
                    (MKI.SUBFNS (\PARSE.NUMBER . NILL)
                           (\MKATOM.FULL . NILL)
                           (\ATOMDEFINDEX . I.ATOMNUMBER)
                           (\ATOMVALINDEX . I.ATOMNUMBER)
                           (\ATOMPROPINDEX . I.ATOMNUMBER)
                           (\ATOMPNAMEINDEX . I.ATOMNUMBER)
                           (\ATOMCELL . I.\ATOMCELL)
                           (\GETBASEFIXP . I.GETBASEFIXP)
                           (\PUTBASEFIXP . I.PUTBASEFIXP)
                           (SETQ.NOREF . SETQ)
                           (SETTOPVAL . I.FSETVAL))
                    (RD.SUBFNS (\PARSE.NUMBER . NILL)
                           (\ATOMDEFINDEX . VATOMNUMBER)
                           (\ATOMPROPINDEX . VATOMNUMBER)
                           (\ATOMVALINDEX . VATOMNUMBER)
                           (SETQ.NOREF . SETQ)
                           (\INDEXATOMPNAME . VATOM)
                           (\INDEXATOMVAL . VATOM)
                           (\INDEXATOMDEF . VATOM)
                           (\ATOMNUMBER . VATOMNUMBER)
                           (\CREATE.SYMBOL . VNOSUCHATOM))
                    (RDCOMS (FNS UNCOPYATOM MAKE.LOCAL.ATOM SYMBOL.VALUE SYMBOL.PNAME SYMBOL.PACKAGE
                                 OLD.FIND.SYMBOL LOOKUP-SYMBOL FIND.PACKAGE FIND.SYMBOL PACKAGE.NAME \MKATOM
                                 GETTOPVAL GETPROPLIST SETTOPVAL GETDEFN \ATOMCELL)
                           (FNS LISTP)
                           (VARS (COPYATOMSTR)))
                    (RD.SUBFNS (\RPLPTR . VPUTBASEPTR))
                    (RDVALS (\AtomFrLst]
      (PROP FILETYPE LLBASIC)))

(DEFINEQ

(LISTP
  [LAMBDA (X)                                                  (* bvm%: "30-Jan-85 10:56")
                                                               ; usually done in microcode

    (AND (EQ (NTYPX X)
             \LISTP)
         (COND
            ((EQ CDRCODING 0)
             T)
            (T                                                 ; Check that it is not a list page header.  This is mostly for benefit
                                                               ; of teleraid
               (NEQ (fetch (POINTER WORDINPAGE) of X)
                    0)))
         X])

(LITATOM
  [LAMBDA (X)                                                  ; Edited 12-Feb-91 16:14 by jds
                                                               ; compiles open to NTYPX check

    ((OPCODES COPY TYPEMASK.N 64 EQ)
     X])

(FIXP
  [LAMBDA (X)                                                  (* lmm "10-MAR-81 15:08")
                                                               ; compiles open to TYPEPs

    (\TYPEMASK.UFN X (LRSH \TT.FIXP 8])

(SMALLP
  [LAMBDA (X)                                                  (* lmm "10-MAR-81 15:10")
                                                               ; compiles open to TYPEP

    (SELECTC (NTYPX X)
        (\SMALLP X)
        NIL])
```

```
(NLISTP
  [LAMBDA (X)                                                    (* lmm "10-MAR-81 15:07")
                                                                 ; compiles open
    (NOT (LISTP X])

(ARRAYP
  [LAMBDA (X)                                                    (* lmm "10-MAR-81 15:11")
                                                                 ; compiles open to TYPEP
    (SELECTC (NTYPX X)
         (\ARRAYP X)
        NIL])

(FLOATP
  [LAMBDA (X)                                                    (* lmm "10-MAR-81 15:11")
                                                                 ; compiles open to TYPEP
    (SELECTC (NTYPX X)
         (\FLOATP X)
        NIL])

(NUMBERP
  [LAMBDA (X)                                                    (* lmm "10-MAR-81 15:12")
    (\TYPEMASK.UFN X (LRSH \TT.NUMBERP 8])

(STACKP
  [LAMBDA (X)                                                    (* lmm "10-MAR-81 15:13")
    (SELECTC (NTYPX X)
         (\STACKP X)
        NIL])

)


(DEFINLINE ATOM (X)
   (OR (NULL X)
       (AND (\TYPEMASK.UFN X 8)
            T)))

(DECLARE%: DONTCOPY

;; FOLLOWING DEFINITIONS EXPORTED

(DECLARE%: EVAL@COMPILE

(PUTPROPS CHECK MACRO [ARGS (COND
                                [(AND (BOUNDP 'CHECK)
                                      CHECK)
                                 (CONS 'PROGN (for I in ARGS
                                                 collect (LIST 'OR I (LIST 'RAID (KWOTE (LIST 'Check-failure%: I
                                                                                                                ]
                                (T (CONS COMMENTFLG ARGS])

(PUTPROPS \StatsZero BYTEMACRO (OPENLAMBDA (N)
                                 (\PUTBASE N 0 0)
                                 (\PUTBASE N 1 0)))

(PUTPROPS \StatsAdd1 DMACRO [OPENLAMBDA (A)
                              (PROG ((LO (IPLUS16 (\GETBASE A 1)
                                                  1)))
                                    (DECLARE (LOCALVARS LO))     ; Increment double word at A by 1
                                    (\PUTBASE A 1 LO)
                                    (COND
                                       ((EQ LO 0)
                                        (\PUTBASE A 0 (ADD1 (\GETBASE A 0])

(PUTPROPS IPLUS16 MACRO ((X Y)                                   ; Kludge to do 16-bit plus
                         (\LOLOC (\ADDBASE X Y))))

(PUTPROPS SMALLPOSP MACRO (OPENLAMBDA (X)
                            (AND (SMALLP X)
                                 (IGEQ X 0))))

[PROGN (PUTPROPS SETXVAR MACRO [X `(SETQ.NOREF %, (CADAR X)
                                            %,
                                            (CADR X])
       (PUTPROPS SETXVAR DMACRO (X (OR (AND (EQ (CAAR X)
                                                'QUOTE)
                                            (LITATOM (CADAR X)))
                                       (SHOULDNT))
                                   (GLOBALVARS \VALSPACE)
                                   (LIST 'SETQ.NOREF (CADAR X)
                                         (CADR X)))))]
```

```
(PUTPROPS SETQ.NOREF DMACRO ((VAR VAL)
                             (\PUTBASEPTR (LOCF (fetch (LITATOM VALUE) of 'VAR))
                                    0 VAL)))

(PROGN (PUTPROPS IEQ MACRO ((X Y)
                            (IEQP X Y)))
       (PUTPROPS IEQ DMACRO (= . EQ)))
)

(DECLARE%: EVAL@COMPILE

(RPAQQ WordsPerPage 256)

(CONSTANTS WordsPerPage)
)

;; END EXPORTED DEFINITIONS

(SETTEMPLATE 'SPREADAPPLY* '(FUNCTIONAL |..| EVAL))

(SETTEMPLATE 'SPREADAPPLY '(FUNCTIONAL EVAL . PPE))

(SETTEMPLATE 'SETQ.NOREF '(SET EVAL . PPE))
)

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(BLOCKRECORD FREELISTENTRY ((FREELINK FULLXPOINTER)))

[BLOCKRECORD HASHENTRY ((HASHPAGE# WORD)
                        (HASHFIRSTOFFSET WORD)
                        (HASHLASTFREE FULLXPOINTER))
       (ACCESSFNS HASHENTRY (HASHMASK (fetch HASHFIRSTOFFSET of DATUM)
                                      (PROGN (replace HASHPAGE# of DATUM with MAX.SMALLP)
                                             (replace HASHFIRSTOFFSET of DATUM with NEWVALUE]
)
)

;; atoms

(DEFINEQ

(GETTOPVAL
  [LAMBDA (X)                                                    (* edited%: " 3-Apr-85 16:38")
    (fetch (LITATOM VALUE) of X])

(SETTOPVAL
  [LAMBDA (ATM VAL)                                              (* edited%: " 3-Apr-85 19:37")
    (SELECTQ ATM
         (NIL (AND VAL (LISPERROR "ATTEMPT TO SET NIL OR T" VAL)))
         (T (OR (EQ VAL T)
                (LISPERROR "ATTEMPT TO SET NIL OR T" VAL)))
         (replace (LITATOM VALUE) of ATM with (UNLESSRDSYS VAL (\COPY VAL])

(FSETVAL
  [LAMBDA (ATM VAL)                                              (* edited%: " 3-Apr-85 19:36")
                                                                 ; SETTOPVAL without error checks for MAKEINIT only
    (replace (LITATOM VALUE) of ATM with VAL])

(\SETGLOBALVAL.UFN
  [LAMBDA (V A)                                                  (* bvm%: " 6-Jun-85 11:54")
    (replace (VALINDEX VALUE) of A with V])

(\SETFVAR.UFN
  [LAMBDA (V VCELL)                                              (* edited%: " 3-Apr-85 16:40")
    (replace (VCELL VALUE) of VCELL with V])

(GETPROPLIST
  [LAMBDA (ATM)                                                  (* edited%: " 3-Apr-85 16:40")
    (\GETBASEPTR (\PROPCELL ATM)
           0])

(\ATOMCELL
  [LAMBDA (X N)                                                  ; Edited  9-Nov-92 14:18 by sybalsky:mv:envos
    (LET ((ATOMNO (\ATOMDEFINDEX X)))
         (COND
            (NIL                                                 ; OLD VERSION
                 (EQ (\HILOC ATOMNO)
```

```
                    0)                                                  ; Xerox Lisp traditional symbol
                 (LET [(LOC (SELECTC N
                                 (\DEF.HI (\ATOMDEFINDEX ATOMNO))
                                 (\VAL.HI (\ATOMVALINDEX ATOMNO))
                                 (\PLIST.HI (\ATOMPROPINDEX ATOMNO))
                                 (\PNAME.HI (\ATOMPNAMEINDEX ATOMNO))
                                 (SHOULDNT]
                       (\ADDBASE (\VAG2 N LOC)
                              LOC)))
              [(FIXP ATOMNO)                                            ; Xerox Lisp traditional symbol
               (LET [(LOC (SELECTC N
                                (\DEF.HI \NEWATOM-DEFOFFSET)
                                (\VAL.HI \NEWATOM-VALOFFSET)
                                (\PLIST.HI \NEWATOM-PLISTOFFSET)
                                (\PNAME.HI \NEWATOM-PNAMEOFFSET)
                                (SHOULDNT]
                     (\ADDBASE \PNPSPACE (IPLUS LOC (ITIMES 10 ATOMNO]
              (T                                                        ; New symbol that appears after traditional symbol runs out.
                 (LET [(OFFSET (SELECTC N
                                    (\DEF.HI \NEWATOM-DEFOFFSET)
                                    (\VAL.HI \NEWATOM-VALOFFSET)
                                    (\PLIST.HI \NEWATOM-PLISTOFFSET)
                                    (\PNAME.HI \NEWATOM-PNAMEOFFSET)
                                    (SHOULDNT]
                       (\ADDBASE ATOMNO OFFSET])
```

(**SETPROPLIST**

```
  [LAMBDA (ATM LST)                                                     (* edited%: " 3-Apr-85 16:41")
    (replace (LITATOM PROPLIST) of ATM with LST])

)

(DECLARE%: EVAL@COMPILE

(PUTPROPS \PROPCELL MACRO ((ATOM)
                           (\ATOMCELL ATOM (CONSTANT \PLIST.HI))))
)


(DEFOPTIMIZER \ATOMCELL (&REST X)
                        [LET [(CE (CONSTANTEXPRESSIONP (CADR X]
                             (COND
                                [CE `((OPCODES ATOMCELL.N %, (CAR CE))
                                      %,
                                      (CAR X]
                                (T 'IGNOREMACRO])


(DEFOPTIMIZER GETPROPLIST (X)
                          `(\GETBASEPTR (\PROPCELL ,X)
                                  0))


(DEFOPTIMIZER SETPROPLIST (ATM LST)
                          `(\RPLPTR (\PROPCELL ,ATM)
                                  0
                                  ,LST))

(DEFINEQ
```

(**\MKATOM**

```
  [LAMBDA (BASE OFFST LEN FATP NONNUMERICP)                             (* bvm%: " 3-Aug-86 15:24")
    (PROG ([FATCHARSEENP (AND FATP (NOT (NULL (for I from OFFST to (SUB1 (IPLUS OFFST LEN))
                                              suchthat (IGREATERP (\GETBASEFAT BASE I)
                                                                  \MAXTHINCHAR]
           HASH HASHENT ATM# PNBASE FIRSTCHAR FIRSTBYTE REPROBE)
      ;; Because FATCHARSEENP is used in an EQ check later, it must be NIL or T only, hence the (NOT (NULL ...))
          (COND
             ((EQ LEN 0)                                                ; The Zero-length atom has hash code zero
              (SETQ HASH 0)
              (SETQ FIRSTBYTE 255)
              (GO LP)))
          (SETQ FIRSTCHAR (UNLESSRDSYS (\GETBASECHAR FATP BASE OFFST)
                                (NTHCHARCODE BASE OFFST)))              ; Grab the first character of the atom
          [UNLESSRDSYS (COND
                          [(AND (EQ LEN 1)
                                (ILEQ FIRSTCHAR \MAXTHINCHAR)
                                \OneCharAtomBase)                       ; The one-character atoms live in well known places, no need to
                                                                        ; hash
                           (RETURN (COND
                                      ((IGREATERP FIRSTCHAR (CHARCODE "9"))
                                       (\ADDBASE \OneCharAtomBase (IDIFFERENCE FIRSTCHAR 10)))
                                      ((IGEQ FIRSTCHAR (CHARCODE "0"))
                                                                        ; These one-character atoms are integers.  Sigh
                                       (IDIFFERENCE FIRSTCHAR (CHARCODE "0")))
```

```
                                    (T (\ADDBASE \OneCharAtomBase FIRSTCHAR]
                              ((AND (NOT NONNUMERICP)
                                    (ILEQ FIRSTCHAR (CHARCODE "9"))
                                    (SETQ HASHENT (\PARSE.NUMBER BASE OFFST LEN FATP 10 \ORIGREADTABLE)))
                                                                    ; \PARSE.NUMBER returns a number or NIL
                               (RETURN HASHENT]                     ; Calculate first probe
          (SETQ FIRSTBYTE (LOGAND FIRSTCHAR 255))
```

;; First byte is used to compute hash and reprobe. Use lower order byte of first character, since chances are that has the most information

```
          (COMPUTE.ATOM.HASH BASE OFFST LEN FIRSTBYTE FATP)         ; Build a hash value for this atom from the PNAME
     LP                                                             ; Top of the probe-and-compare-PNAMEs loop.
          [COND
             ((NEQ 0 (SETQ HASHENT (\GETBASE \AtomHashTable HASH)))
```

;; HASHENT is one greater than the atom number, so that atom zero can be stored. Go from atom number to pname, compare
;; strings

```
              (COND
                 ((UNLESSRDSYS [AND (EQ [ffetch (PNAMEBASE PNAMELENGTH) of (SETQ PNBASE (ffetch (PNAMEINDEX
                                                                                                PNAMEBASE)
                                                                                             of (SETQ ATM#
                                                                                                  (SUB1 HASHENT]
                                        LEN)
                                    [EQ FATCHARSEENP (AND (PROG1 (EQ 0 (ffetch (PNAMEBASE PNAMEFATPADDINGBYTE)
                                                                          of PNBASE))
```

;; Extra memory references to get the FATPNAMEP bit, so do a quick and dirty heuristic, based on
;; the fact that the second byte of a fatpname is always 0--wouldn't be worth it if the fatbit were more
;; easily accessible

```
                                                                  )
                                                           (ffetch (LITATOM FATPNAMEP) of (\ADDBASE \ATOMSPACE
                                                                                             ATM#]
                                    (COND
                                       [FATCHARSEENP                ; FATCHARSEENP=T now implies that both the probe and
                                                                    ; target are fat
                                              (for B1 from 1 to LEN as B2 from OFFST
                                                 always             ; Loop thru the characters in the putative atom and the existing
                                                                    ; PNAME, to see if they're the same
                                                       (EQ (\GETBASEFAT PNBASE B1)
                                                           (\GETBASEFAT BASE B2]
                                       [FATP                        ; The incoming string is fat, but there are no fat characters in the
                                                                    ; PNAME.
                                             (for B1 from 1 to LEN as B2 from OFFST
                                                always (EQ (\GETBASETHIN PNBASE B1)
                                                           (\GETBASEFAT BASE B2]
                                       (T                           ; Both the incoming string of chars and the PNAME are thin.
                                          (for B1 from 1 to LEN as B2 from OFFST
                                             always (EQ (\GETBASETHIN PNBASE B1)
                                                        (\GETBASETHIN BASE B2]
                              (EQ (\INDEXATOMPNAME (SETQ ATM# (SUB1 HASHENT)))
                                  BASE))
                       (RETURN (\ADDBASE \ATOMSPACE ATM#)))
                      (T                                            ; Doesn't match, so reprobe.  Want reprobe to be variable,
                                                                    ; preferably independent of primary probe.
                         [SETQ HASH (IPLUS16 HASH (OR REPROBE (SETQ REPROBE (ATOM.HASH.REPROBE HASH FIRSTBYTE]
                         (GO LP]                                    ; Not found, must make new atom
          (RETURN (UNINTERRUPTABLY
                      (LET ((NEWATOM (\CREATE.SYMBOL BASE OFFST LEN FATP FATCHARSEENP)))
                           [UNLESSRDSYS (\PUTBASE \AtomHashTable HASH (ADD1 (\ATOMPNAMEINDEX NEWATOM]
                           NEWATOM)])
```

(\CREATE.SYMBOL

```
  [LAMBDA (BASE OFFSET LEN FATP FATCHARSEENP)                       ; Edited  8-Feb-93 16:48 by jds
```

;;; Creates a new symbol whose pname is as indicated. FATP means the presented string is fat, while FATCHARSEENP means that there actually is a
;;; fat char in there (otherwise we will store a thin pname) --- Must be called UNINTERRUPTABLY and the caller is responsible for interning the symbol
;;; wherever it belongs

;; WARNING: Changes here (e.g., to where we seitch over to bigatoms) need to be reflected in MAPATOMS, too.

```
     (LET ([PNBASE (\ALLOCBLOCK (COND
                                   (FATCHARSEENP                    ; Allocate us a bunch of word-sized chars in pname space
                                          (FOLDHI (ADD1 LEN)
                                                  WORDSPERCELL))
                                   (T                               ; Allocation is in CELLS
                                      (FOLDHI (ADD1 LEN)
                                              BYTESPERCELL]
           PB CPP ATM)
          [COND
             ((IGEQ (SETQ ATM \AtomFrLst)
                    12287)
```

;; used to be:

```
              (IGEQ (SETQ ATM \AtomFrLst)
                    \MaxAtomFrLst)                                  ; This test WAS fast (it used to be EQ), with the old, painful
                                                                    ; result:
                                                                    ; (\MP.ERROR \MP.ATOMSFULL "No more atoms left")
```

```
            ;; Now, just create us a NEW-ATOM, and keep going:
            (SETQ ATM (CREATECELL \NEW-ATOM))
            (REPLACE (VALINDEX VALUE) OF ATM WITH 'NOBIND))
           ((EVENP ATM 256)                                    ; Can fit 256 new atoms into 10 pages.
            ;; Old Condition:
            (EVENP ATM \MDSIncrement)                          ; MDS pages are allocated in two-page chunks now
            ;; THE ITIEMS 10 IS NEW FOR BIGVM:
            (LET [(PN (ITIMES 10 (FOLDLO ATM WORDSPERPAGE]
                 (COND
                    ((NEW-SYMBOL-CODE NIL (IGEQ PN (IDIFFERENCE \LastAtomPage 1)))
                     ;; This used to cause the "You're running out of atoms" error.
                     (\MKATOM.FULL)))
                 (\MAKEMDSENTRY (FOLDLO ATM WORDSPERPAGE)
                        (LOGOR \TT.NOREF \TT.SYMBOLP \TT.ATOM \LITATOM))
                                                               ; Make entry in MDS type table
                 (\INITATOMPAGE PN)                            ; Make Def'n, TopVal, and Plist pages exist, and initialize

                 ]
           (replace (PNAMEINDEX PNAMEBASE) of ATM with PNBASE) ; PNAME starts on byte 1 always --- byte 0 is the length
           (COND
              (FATCHARSEENP (\BLT (\ADDBASE PNBASE 1)
                                  (\ADDBASE BASE OFFSET)
                                  LEN))
              [FATP (for I from OFFSET as J from 1 to LEN do (\PUTBASETHIN PNBASE J (\GETBASEFAT BASE I]
              (T (\MOVEBYTES BASE OFFSET PNBASE 1 LEN)))
           (replace (PNAMEBASE PNAMELENGTH) of PNBASE with LEN)
           (COND
              ((NOT \IN.MAKEINIT)                              ; Make the pname block permanent, since the replace above did
                                                               ; not addref it
               (\ADDREF PNBASE)))
           (SETQ \AtomFrLst (ADD1 \AtomFrLst))
           ;; If it's an old atom (so ATM is an atom#), change it to a LITATOM:
           (AND (FIXP ATM)
                (SETQ ATM (\ADDBASE \ATOMSPACE ATM)))
           (COND
              (FATCHARSEENP (freplace (LITATOM FATPNAMEP) of ATM with T)))
          ATM])
```

(\MKATOM.FULL

```
  [LAMBDA NIL                                                  (* bvm%: " 7-May-86 12:25")
```

;;; Cause a STORAGEFULL interrupt on the first atom of the penultimate page -- that should give 'early' warning.

```
    (DECLARE (GLOBALVARS \STORAGEFULL \INTERRUPTSTATE))
    (COND
       ((NOT \STORAGEFULL)
        (SETQ \STORAGEFULL T)
        (replace STORAGEFULL of \INTERRUPTSTATE with T)
        (SETQ \PENDINGINTERRUPT T)))
   NIL])
```

(\INITATOMPAGE

```
  [LAMBDA (PN)                                                 ; Edited 28-Oct-92 15:47 by sybalsky:mv:envos
    (COND
       [NIL (PROG ((OFFSET (UNFOLD PN WORDSPERPAGE))
                   VALBASE)
            ;; PN is the page number of the first atom.  OFFSET is the first atom.  Have to double that to get offsets in \DEFSPACE etc.  Atoms,
            ;; like everything, are allocated in double pages, so the 4 spaces have to be allocated in quad pages
            ;; assumes CCODEP bit in definition cell is default 'OFF' , so it's ok to have all def pages zero to start
                  (\NEW4PAGE (\ADDBASE2 \PNPSPACE OFFSET))
                  (\NEW4PAGE (\ADDBASE2 \DEFSPACE OFFSET))
                  (\NEW4PAGE (\ADDBASE2 \PLISTSPACE OFFSET))
                  (\NEW4PAGE (SETQ VALBASE (\ADDBASE2 \VALSPACE OFFSET)))
                  (FRPTQ (ITIMES CELLSPERPAGE 4)               ; Initialize value pages to value NOBIND
                         (\PUTBASEPTR VALBASE 0 (EVQ 'NOBIND))
                         (SETQ VALBASE (\ADDBASE VALBASE WORDSPERCELL]
       (T ;; New, big-VM code:  Allocate 10 pages in PNPspace at a crack, to hold 256 atoms.
          (LET ((OFFSET (UNFOLD PN WORDSPERPAGE))
                (ATM (UNFOLD (IQUOTIENT PN 10)
                             WORDSPERPAGE))
                VALBASE)
               ;; Create the new pages in what used to be PNAME space:
               (FOR I FROM 0 TO 9 AS OFF FROM OFFSET BY WORDSPERPAGE DO (\NEWPAGE (\ADDBASE \PNPSPACE OFF)))
               ;; Make all the atoms' values be NOBIND:
               (FOR I FROM 0 TO 255 AS OFF FROM OFFSET BY 10 DO (\PUTBASEPTR \PNPSPACE (IPLUS OFF
                                                                                             \NEWATOM-VALOFFSET
```

```
                                                                                                    )
                                                                                        'NOBIND])

)

(DEFINEQ

(MAPATOMS
  [LAMBDA (FN)
    (DECLARE (LOCALVARS . T))                                    ; Edited 8-Feb-93 16:47 by jds
    ;; 8-FEB-92 JDS:  We now switch over into big-atom mode at 12288  (changes in \CREATE.SYMBOL should be lected here)
    (PROG ((A 0))
          (for A from 0 to (IMIN \AtomFrLst 12286) do (APPLY* FN (\INDEXATOMPNAME A)))
          (COND
             ((IGREATERP \AtomFrLst 12286)
              (CL:FLET
               [(\SFLHASHLOOKUP (PAGE# HASHTABLE INSERT)
                        (bind (MASK _ (fetch HASHMASK of HASHTABLE))
                              PROBE HASHENT first (SETQ PROBE (LOGAND (LLSH PAGE# 2)
                                                                      MASK))
                           do [COND
                                 ((EQ (fetch HASHPAGE# of (SETQ HASHENT (\ADDBASE HASHTABLE PROBE)))
                                      PAGE#)
                                  (RETURN HASHENT))
                                 ((EQ 0 (fetch HASHPAGE# of HASHENT))
                                  (RETURN (COND
                                              (INSERT (replace HASHPAGE# of HASHENT with PAGE#)
                                                      HASHENT]
                              (SETQ PROBE (LOGAND (IPLUS PROBE 4)
                                                  MASK]
               (PROG ((HASHTABLE
                        (PROG ((DTD (\GETDTD \NEW-ATOM))
                               NPAGES HASHTABLE HASHENT HSIZE NEXTFREE NEXTPAGE LASTPAGE FIRSTFREE LASTFREE
                               OTHERLASTFREE PREVPAGELASTFREE PROBE MASK)
                              (SETQ NPAGES (ITIMES (for I from 0 to \MAXVMPAGE by 2
                                                      count (EQ (MDSTYPE# I)
                                                                \NEW-ATOM))
                                                   2))
                              (SETQ HSIZE (FIX (TIMES NPAGES 1.4)))
                                                                 ; Good size of hashtable for hashing pages of this type into
                              (SETQ HSIZE (find I from 8 by I suchthat (IGREATERP I HSIZE)))
                                                                 ; Get a power of 2
                              (SETQ HASHTABLE (\ALLOCBLOCK (ITIMES HSIZE 2)))
                              (replace HASHMASK of HASHTABLE with (SUB1 (ITIMES HSIZE 4)))
                              (SETQ NEXTFREE (fetch DTDFREE of DTD))
                              [do
                               (COND
                                  ((NEQ (SETQ NEXTPAGE (fetch (POINTER PAGE#) of NEXTFREE))
                                        LASTPAGE)                ; Cell on a new page
                                   [COND
                                      ((AND NEXTFREE (NEQ (NTYPX NEXTFREE)
                                                          \NEW-ATOM))
                                       (RETURN (RAID "Bad free list" NEXTFREE]
                                   (COND
                                      (LASTPAGE                  ; Hash LASTPAGE and see if we have already seen cells free on
                                                                 ; this page
                                       (SETQ HASHENT (\SFLHASHLOOKUP LASTPAGE HASHTABLE T))
                                       (COND
                                          [(SETQ OTHERLASTFREE (fetch HASHLASTFREE of HASHENT))
                                                                 ; Yes, we have seen others.  Link this section of the free list into
                                                                 ; it
                                           (COND
                                              ((EQ (fetch FREELINK of OTHERLASTFREE)
                                                   FIRSTFREE)
                                               ;; Aha, already in order.  This happens when we have a sequence LASTPAGE -> x ...  -> LASTPAGE
                                               ;; where everything in between the two LASTPAGE's got moved to earlier in the freelist
                                                (SETQ PREVPAGELASTFREE LASTFREE))
                                              (T (UNINTERRUPTABLY
                                                     [replace FREELINK of OTHERLASTFREE
                                                        with (PROG1 FIRSTFREE
                                                                    (replace FREELINK
                                                                       of (OR PREVPAGELASTFREE (RETURN (RAID "No
                                                                                                             PREVPAGE
LASTFREE")))

                                                                       with NEXTFREE)
                                                                    (replace FREELINK of LASTFREE
                                                                       with (fetch FREELINK of OTHERLASTFREE)))])]
                                                 (T (replace HASHFIRSTOFFSET of HASHENT with (\LOLOC FIRSTFREE))
                                                    (SETQ PREVPAGELASTFREE LASTFREE)))
                                           (replace HASHLASTFREE of HASHENT with LASTFREE)))
                                       (OR (SETQ FIRSTFREE NEXTFREE)
                                           (RETURN))
                                       (SETQ LASTPAGE NEXTPAGE)))
                                   (SETQ NEXTFREE (fetch FREELINK of (SETQ LASTFREE NEXTFREE]
                              (SETQ LASTPAGE (SETQ PREVPAGELASTFREE))
```

```
            (SETQ NEXTFREE (fetch DTDFREE of DTD))
      ;; Now take a quick second pass to link all odd pages immediately after the corresponding even pages.  Might possibly
      ;; have done this in the previous loop, but the logic gets pretty messy
            [do
             (COND
                ((NEQ (SETQ NEXTPAGE (fetch (POINTER PAGE#) of NEXTFREE))
                      LASTPAGE)                    ; Cell on a new page
                 [COND
                    (LASTPAGE
                     (COND
                        [(AND (ODDP LASTPAGE)
                              (SETQ HASHENT (\SFLHASHLOOKUP (LOGXOR LASTPAGE 1)
                                                            HASHTABLE))
                              (NEQ (fetch FREELINK of (SETQ OTHERLASTFREE (fetch HASHLASTFREE
                                                                                 of HASHENT)))
                                   FIRSTFREE))  ; There is an entry for our partner even page, and it is not
                                                ; immediately followed by its odd partner
                         (OR NIL
                             (UNINTERRUPTABLY
                                 [replace FREELINK of OTHERLASTFREE
                                    with (PROG1 FIRSTFREE
                                                (COND
                                                   (PREVPAGELASTFREE (replace FREELINK of
                                                                              PREVPAGELASTFREE
                                                                      with NEXTFREE))
                                                   (T (OR (EQ FIRSTFREE (fetch DTDFREE of DTD))
                                                          (RAID "No PREVPAGELASTFREE"))
                                                      (replace DTDFREE of DTD with NEXTFREE)))
                                                (replace FREELINK of LASTFREE
                                                   with (fetch FREELINK of OTHERLASTFREE)))])]
                         (T (SETQ PREVPAGELASTFREE LASTFREE]
                 (OR (SETQ FIRSTFREE NEXTFREE)
                     (RETURN))
                 (SETQ LASTPAGE NEXTPAGE)))
             (SETQ NEXTFREE (fetch FREELINK of (SETQ LASTFREE NEXTFREE]
             (RETURN HASHTABLE)))
       (SIZE (fetch DTDSIZE of (\GETDTD \NEW-ATOM)))
       (ATOM# 65536)
       RESULT FIRSTFREE LASTFREE HASHENT LASTPAGE LIMIT)
      (OR HASHTABLE (RETURN))
      (OR (EVENP SIZE)
          (SHOULDNT "Odd size?"))
      (COND
         ((.ALLOCATED.PER.PAGE. SIZE)
          (SETQ LASTPAGE (SUB1 \PagesPerMDSUnit))
          (SETQ LIMIT WORDSPERPAGE))
         (T (SETQ LASTPAGE 0)
            (SETQ LIMIT \MDSIncrement)))
      [for MDSPAGE# from 0 by \PagesPerMDSUnit while (<= MDSPAGE# \MAXVMPAGE)
         when (EQ (MDSTYPE# MDSPAGE#)
                  \NEW-ATOM)
         do [COND
               ([SETQ FIRSTFREE (COND
                                   ((SETQ HASHENT (OR (\SFLHASHLOOKUP MDSPAGE# HASHTABLE)
                                                      (\SFLHASHLOOKUP (LOGOR MDSPAGE# 1)
                                                                      HASHTABLE)))
                                    (\VAG2 (FOLDLO MDSPAGE# PAGESPERSEGMENT)
                                           (fetch HASHFIRSTOFFSET of HASHENT]
                (SETQ LASTFREE (fetch HASHLASTFREE
                                  of (OR (AND (EVENP (fetch HASHPAGE# of HASHENT))
                                              (\SFLHASHLOOKUP (LOGOR MDSPAGE# 1)
                                                              HASHTABLE))
                                         HASHENT]
      ;; Now collect all pointers not on free list.  This code parallels \INITMDSPAGE
            (for N from 0 to LASTPAGE
               do (for (DISP _ 0) while (<= (add DISP SIZE)
                                            LIMIT)
                     as (DATUMBASE _ (create POINTER
                                             PAGE# _ (IPLUS N MDSPAGE#)))
                     by (\ADDBASE DATUMBASE SIZE)
                     when [AND (OR (NOT FIRSTFREE)
                                   (for (X _ FIRSTFREE) by (fetch FREELINK of X)
                                      never (EQ X DATUMBASE) repeatuntil (EQ X LASTFREE]
                     do (APPLY* FN DATUMBASE)
                        (ADD ATOM# 1]
      (RETURN])
```

(ATOMHASH#PROBES

```
  [LAMBDA (STRING)                                    (* bvm%: " 8-Jul-86 21:50")
```

;;; Looks up STRING (a string or litatom) in atom hash table. If found, returns number of probes needed to find it, a minimum of one. If not found,
;;; returns NIL

```
    (PROG (DESIREDATOM# BASE OFFST LEN FIRSTBYTE FIRSTCHAR HASH HASHENT PNBASE REPROBE FATCHARSEENP FATP)
```

```
          [COND
             ((LITATOM STRING)
              (SETQ BASE (ffetch (LITATOM PNAMEBASE) of STRING))
              (SETQ OFFST 1)
              (SETQ LEN (ffetch (LITATOM PNAMELENGTH) of STRING))
              (SETQ FATP (SETQ FATCHARSEENP (ffetch (LITATOM FATPNAMEP) of STRING)))
              (SETQ DESIREDATOM# (\LOLOC STRING)))
             (T [SETQ BASE (ffetch (STRINGP BASE) of (SETQ STRING (MKSTRING STRING]
                (SETQ OFFST (ffetch (STRINGP OFFST) of STRING))
                (SETQ LEN (ffetch (STRINGP LENGTH) of STRING))
                [COND
                   ((SETQ FATP (ffetch (STRINGP FATSTRINGP) of STRING))
                    (SETQ FATCHARSEENP (for C infatstring STRING when (IGREATERP C \MAXTHINCHAR)
                                          do (RETURN T]
                (OR (ILEQ LEN \PNAMELIMIT)
                    (RETURN]
          (SETQ FIRSTCHAR (\GETBASECHAR FATP BASE OFFST))
          (SETQ FIRSTBYTE (LOGAND FIRSTCHAR 255))
          (COMPUTE.ATOM.HASH BASE OFFST LEN FIRSTBYTE FATP)
          (RETURN (for PROBES from 1 until (EQ 0 (SETQ HASHENT (\GETBASE \AtomHashTable HASH)))
                     do (COND
                           ([COND
                               (DESIREDATOM# (EQ DESIREDATOM# (SUB1 HASHENT)))
                               (T (AND (EQ [fetch (PNAMEBASE PNAMELENGTH) of (SETQ PNBASE (fetch (PNAMEINDEX
                                                                                                  PNAMEBASE)
                                                                                           of (SUB1 HASHENT]
                                           LEN)
                                       [EQ FATCHARSEENP (ffetch (LITATOM FATPNAMEP) of (\ADDBASE \ATOMSPACE
                                                                                               (SUB1 HASHENT]
                                       (COND
                                          [FATCHARSEENP          ; FATCHARSEENP=T now implies that both the probe and
                                                                 ; target are fat
                                                 (for B1 from 1 to LEN as B2 from OFFST
                                                    always       ; Loop thru the characters in the putative atom and the existing
                                                                 ; PNAME, to see if they're the same
                                                          (EQ (\GETBASEFAT PNBASE B1)
                                                              (\GETBASEFAT BASE B2]
                                          [FATP                  ; The incoming string is fat, but there are no fat characters in the
                                                                 ; PNAME.
                                                (for B1 from 1 to LEN as B2 from OFFST
                                                   always (EQ (\GETBASETHIN PNBASE B1)
                                                              (\GETBASEFAT BASE B2]
                                          (T                     ; Both the incoming string of chars and the PNAME are thin.
                                             (for B1 from 1 to LEN as B2 from OFFST
                                                always (EQ (\GETBASETHIN PNBASE B1)
                                                           (\GETBASETHIN BASE B2]
                                   (RETURN PROBES)))             ; Doesn't match, so reprobe.  Want reprobe to be variable,
                                                                 ; preferably independent of primary probe.
                              (SETQ HASH (IPLUS16 HASH (OR REPROBE (SETQ REPROBE (ATOM.HASH.REPROBE HASH FIRSTBYTE])

)

;; For MAKEINIT & TeleRaid

;; This code has one major shortcoming which will not normally turn up.  If the local and remote sysouts conflict in their package setups it is possible for
;; this code to return symbols interned in what for the teleraid'ing machine would be the correct package, but for the remote machine is in fact incorrect.
;; This warrents a warning in the documentation.  The problem lies in the fact that you *cannot* uncopy a symbol correctly between two machines with
;; incompatible package setups.  An example of such a situation would be where on one machine the package FOO inherits BAR, and on the other BAR
;; is present directly in FOO.  BAR's package cell will be different in both cases.  Two solutions come to mind; both would break the VSAVEWORK
;; feature.  The first would be to UNCOPY symbols into special "remote symbol" objects.  The second is to create uninterned symbols with the correct
;; name and smash their package cell to be that of a correctly named package.  Both of these schemes would require special reading and printing code.

(DECLARE%: EVAL@COMPILE

(PUTPROPS READSYS.HAS.PACKAGES MACRO (NIL (NEQ 1 READSYS.PACKAGE.FROM.NAME)))
)

(RPAQQ READSYS.PACKAGE.FROM.NAME 1)

(RPAQQ READSYS.PACKAGE.FROM.INDEX 1)

(DEFINEQ

## (INITATOMS

```
  [LAMBDA NIL   ; Edited 11-Dec-86 14:41 by Pavel
    ;; called only under MAKEINIT to initialize the making of atoms
    (CREATEPAGES \AtomHashTable \AtomHTpages)
    (SETQ \SCRATCHSTRING (ALLOCSTRING \PNAMELIMIT))   ; \SCRATCHSTRING created in remote space simply to make
                                                                 ; renaming simple. Could smash it to NIL inside init.sysout
                                                                 (* (CREATEPAGES \PNCHARSSPACE 1))
    (COPYATOM NIL)   ; NIL is atom 0
    (COPYATOM 'NOBIND)   ; atom 1
    ;; Now make the single character atoms -- all thin chars except the digits
    (for C from 0 to 255 when (OR (ILESSP C (CHARCODE 0))
                                  (IGREATERP C (CHARCODE 9)))
```
```

```
      do (COPYATOM (CHARACTER C)))
    (SETQ \OneCharAtomBase (\ADDBASE \ATOMSPACE 2))          ; = (CHARACTER 0) -- for FCHARACTER
    (COPYATOM (FUNCTION \EVALFORM))                          ; atom 256-10+2 = 248
    (COPYATOM (FUNCTION \GC.HANDLEOVERFLOW))                 ; atom 249
    (COPYATOM (FUNCTION \DTEST.UFN))                         ; atom 250
    (COPYATOM (FUNCTION \OVERFLOWMAKENUMBER))                ; atom 251
    (COPYATOM (FUNCTION \MAKENUMBER))                        ; atom 252
    (COPYATOM (FUNCTION \SETGLOBAL.UFN))                     ; atom 253
    (COPYATOM (FUNCTION \SETFVAR.UFN))                       ; atom 254
    (COPYATOM (FUNCTION \GCMAPTABLE))                        ; atom 255
    (COPYATOM (FUNCTION \INTERPRETER))                       ; atom 256
    (OR (EQ (\ATOMDEFINDEX (FUNCTION \INTERPRETER))
            256)
        (HELP (FUNCTION \INTERPRETER)
              " not atom 400Q"])
```

(COPYATOM

```
  [LAMBDA (X)                                                ; Edited 29-Apr-91 16:01 by jrb:
```

;; this function is only for the use of MAKEINIT, which passes it a local atom to be translated into an atom in the remote sysout.

```
    [ALLOCAL (LET ((PKG (CL:SYMBOL-PACKAGE X)))              ; SYMBOL-PACKAGE and *INTERLISP-PACKAGE* both NIL in
                                                             ; non-package world
                  (COND
                     ((EQ PKG *INTERLISP-PACKAGE*))
                     ((NULL PKG)                             ; This is an uninterned symbol, so add  prefix.
                      (SETQ X (CONCAT "" X)))
                     ((EQ PKG *KEYWORD-PACKAGE*)             ; keywords eval to self, so also set top val
                      (MKI.DSET X X)
                      (SETQ X (CONCAT ":" X)))
                     (T
```

;; Kludge time. We don't yet have the machinery to create packages in the init.sysout, so anything that isn't an Interlisp ;; symbol has to be turned into a flat-space symbol with appropriate prefix

```
                        (CL:MULTIPLE-VALUE-BIND (SYM WHERE)
                            (CL:FIND-SYMBOL (CL:SYMBOL-NAME X)
                                  PKG)
                          (SETQ WHERE (SELECTQ WHERE
                                          (:INTERNAL "::")
                                          (:EXTERNAL ":")
                                          (ERROR "Where is this symbol?" X)))
                          (COND
                             ((EQ PKG *LISP-PACKAGE*)
                              (SETQ SYM "LISP"))
                             ((EQ PKG *COMMON-LISP-PACKAGE*)
                              (SETQ SYM "CL"))
                             ((CL:STRING= (CL:PACKAGE-NAME PKG)
                                     "SYSTEM")
                              (SETQ SYM "SI"))
                             ((CL:STRING= (CL:PACKAGE-NAME PKG)
                                     "CONDITIONS")
                              (SETQ SYM "CONDITIONS"))
                             ((CL:STRING= (CL:PACKAGE-NAME PKG)
                                     "XEROX-COMMON-LISP")
                              (SETQ SYM "XCL"))
                             ((CL:STRING= (CL:PACKAGE-NAME PKG)
                                     "COMPILER")
                              (SETQ SYM "COMPILER"))
                             ((CL:STRING= (CL:PACKAGE-NAME PKG)
                                     "FASL")
                              (SETQ SYM "FASL"))
                             (T (HELP "Can only translate symbols in IL, CL, XCL, CONDITIONS, SI, COMPILER, FASL
                                      and keywords" X)))
                          (SETQ X (CONCAT SYM WHERE (CL:SYMBOL-NAME X))))))]
    (LET ((N (LOCAL (NCHARS X)))
          (BASE (ffetch (STRINGP BASE) of \SCRATCHSTRING))
          (OFFST (ffetch (STRINGP OFFST) of \SCRATCHSTRING)))      ; \SCRATCHSTRING is initialized in INITATOMS
         [for I from 1 to N do (\PUTBASEBYTE BASE (LOCAL (IPLUS OFFST I -1))
                                    (LOCAL (NTHCHARCODE X I]
         (\ATOMDEFINDEX (\MKATOM BASE OFFST N])
```

(UNCOPYATOM

```
  [LAMBDA (N)                                                ; Edited  6-Mar-87 11:55 by raf
```

;;; This is used only by VATOM (in READSYS) to turn atom numbers into similar local atoms. Note that it would be very difficult to create correctly ;;; exported symbols due to conflicts between the local and remote package setups.

```
    (PROG (ATOM.NAME PACKAGE.NAME)
```

;; Uncopy the atom name

```
          (SETQ ATOM.NAME (SYMBOL.PNAME N))
```

;; Find and uncopy the package name

```
          (SETQ PACKAGE.NAME (IF (READSYS.HAS.PACKAGES)
                                 THEN (PACKAGE.NAME (SYMBOL.PACKAGE N))
                               ELSE "INTERLISP"))
          (RETURN (MAKE.LOCAL.ATOM PACKAGE.NAME ATOM.NAME])
```

(MAKE.LOCAL.ATOM

```
[LAMBDA (PKG.NAME ATM.NAME)                                    ; Edited 17-Feb-87 16:20 by raf
```

;;; There are potential cases in which package setup differences between the local and remote machines will intern names in different packages. For ;;; example, if in the local package the name is an inherited symbol, but remotely the name is directly present in the paackage (shadowed symbol have ;;; the same problem). This is mildly troublesome, however any solution would break VSAVEWORK. In future it would be best to create a ;;; remote-symbol structure and pass that around.

```
    (ALLOCAL (CL:INTERN ATM.NAME (OR (CL:FIND-PACKAGE PKG.NAME)
                                     (CL:MAKE-PACKAGE PKG.NAME :USES NIL])
```

(SYMBOL.VALUE

```
[LAMBDA (SYMBOL)                                               ; Edited 22-Dec-92 17:05 by jds
```

;; Get a symbol's value. This is for RDSYS only.

```
(LET [(LOC (OLD.FIND.SYMBOL SYMBOL 1 (LOCAL (NCHARS SYMBOL]
     (COND
        (NIL ;; OLD VERSION
             (\GETBASEPTR (VADDBASE (VVAG2 12 LOC)
                                    LOC)
                    0))
        (T ;; NEW VERSION
           (\GETBASEPTR (VADDBASE (VVAG2 8 0)
                                  (IPLUS (ITIMES (LOGAND LOC 65535)
                                                 10)
                                         \NEWATOM-VALOFFSET))
                  0])
```

(SYMBOL.PNAME

```
[LAMBDA (N BUFFER)                                             ; Edited 22-Dec-92 16:32 by jds
```

;;; Uncopy the pname of symbol number N into a string and return it.

```
[ALLOCAL (SETQ BUFFER (OR BUFFER (ALLOCSTRING \PNAMELIMIT]
(PROG (ADDR LEN)
```

;; Uncopy the atom name

```
      [COND
         (NIL (SETQ ADDR (\GETBASEPTR (\ADDBASE2 \PNPSPACE N)
                                      0)))
         (T (SETQ ADDR (\GETBASEPTR (\ADDBASE (\VAG2 8 0)
                                              (IPLUS (ITIMES (LOGAND N 65535)
                                                             10)
                                                     \NEWATOM-PNAMEOFFSET))
                                    0]
      (SETQ LEN (\GETBASEBYTE ADDR 0))
      [for I from 1 to LEN do (LOCAL (RPLSTRING BUFFER I (FCHARACTER (\GETBASEBYTE ADDR I]
      (RETURN (LOCAL (SUBSTRING BUFFER 1 LEN])
```

(SYMBOL.PACKAGE

```
[LAMBDA (N)                                                    ; Edited  8-Nov-92 02:16 by sybalsky:mv:envos
```

;;; Given a symbol number, return a pointer to its remote package.

```
(PROG [(INDEX (COND
                 (NIL                                          ; OLD WAY
                      (LRSH (\GETBASE (\ADDBASE2 \PNPSPACE N)
                                      0)
                            8))
                 (NIL (T (LRSH (\GETBASE (\ADDBASE \PNPSPACE (IPLUS (ITIMES 10 N)
                                                                    \NEWATOM-PNAMEOFFSET 8))
                                         0)
                               8)))
                 (T (fetch (LITATOM PACKAGEINDEX) OF N]
      (RETURN (COND
                 ((EQ INDEX *UNINTERNED-PACKAGE-INDEX*)
                  NIL)
                 (T (\GETBASEPTR (FFETCH (ONED-ARRAY BASE) OF READSYS.PACKAGE.FROM.INDEX)
                           (LLSH INDEX 1])
```

(OLD.FIND.SYMBOL

```
[LAMBDA (BASE OFFST LEN FATP NONNUMERICP)                      ; Edited 17-Feb-87 16:43 by raf
  (PROG ([FATCHARSEENP (AND FATP (NOT (NULL (for I from OFFST to (SUB1 (IPLUS OFFST LEN))
                                            suchthat (IGREATERP (\GETBASEFAT BASE I)
                                                                \MAXTHINCHAR]
        HASH HASHENT ATM# PNBASE FIRSTCHAR FIRSTBYTE REPROBE)
```

;; Because FATCHARSEENP is used in an EQ check later, it must be NIL or T only, hence the (NOT (NULL ...))

```
      (COND
```

```
            ((EQ LEN 0)                                          ; The Zero-length atom has hash code zero
             (SETQ HASH 0)
             (SETQ FIRSTBYTE 255)
             (GO LP)))
          (SETQ FIRSTCHAR (UNLESSRDSYS (\GETBASECHAR FATP BASE OFFST)
                                (NTHCHARCODE BASE OFFST)))       ; Grab the first character of the atom
          [UNLESSRDSYS (COND
                         [(AND (EQ LEN 1)
                               (ILEQ FIRSTCHAR \MAXTHINCHAR)
                               \OneCharAtomBase)                 ; The one-character atoms live in well known places, no need to
                                                                 ; hash
                          (RETURN (COND
                                     ((IGREATERP FIRSTCHAR (CHARCODE "9"))
                                      (\ADDBASE \OneCharAtomBase (IDIFFERENCE FIRSTCHAR 10)))
                                     ((IGEQ FIRSTCHAR (CHARCODE "0"))
                                                                 ; These one-character atoms are integers.  Sigh
                                      (IDIFFERENCE FIRSTCHAR (CHARCODE "0")))
                                     (T (\ADDBASE \OneCharAtomBase FIRSTCHAR]
                         ((AND (NOT NONNUMERICP)
                               (ILEQ FIRSTCHAR (CHARCODE "9"))
                               (SETQ HASHENT (\PARSE.NUMBER BASE OFFST LEN FATP 10 \ORIGREADTABLE)))
                                                                 ; \PARSE.NUMBER returns a number or NIL
                          (RETURN HASHENT]                       ; Calculate first probe
          (SETQ FIRSTBYTE (LOGAND FIRSTCHAR 255))
    ;; First byte is used to compute hash and reprobe.  Use lower order byte of first character, since chances are that has the most information
          (COMPUTE.ATOM.HASH BASE OFFST LEN FIRSTBYTE FATP)      ; Build a hash value for this atom from the PNAME
     LP                                                          ; Top of the probe-and-compare-PNAMEs loop.
          [COND
             ((NEQ 0 (SETQ HASHENT (\GETBASE \AtomHashTable HASH)))
              ;; HASHENT is one greater than the atom number, so that atom zero can be stored.  Go from atom number to pname, compare
              ;; strings
              (COND
                 ([UNLESSRDSYS [AND (EQ [ffetch (PNAMEBASE PNAMELENGTH) of (SETQ PNBASE (ffetch (PNAMEINDEX
                                                                                                 PNAMEBASE)
                                                                                               of (SETQ ATM#
                                                                                                    (SUB1 HASHENT]
                                        LEN)
                                    [EQ FATCHARSEENP (AND (PROG1 (EQ 0 (ffetch (PNAMEBASE PNAMEFATPADDINGBYTE)
                                                                          of PNBASE))
                                                          ;; Extra memory references to get the FATPNAMEP bit, so do a quick and dirty heuristic, based on
                                                          ;; the fact that the second byte of a fatpname is always 0--wouldn't be worth it if the fatbit were more
                                                          ;; easily accessible

                                                                                                 )
                                                                                        (ffetch (LITATOM FATPNAMEP) of (\ADDBASE \ATOMSPACE
                                                                                                                          ATM#]
                                    (COND
                                       [FATCHARSEENP             ; FATCHARSEENP=T now implies that both the probe and
                                                                 ; target are fat
                                              (for B1 from 1 to LEN as B2 from OFFST
                                                 always          ; Loop thru the characters in the putative atom and the existing
                                                                 ; PNAME, to see if they're the same
                                                       (EQ (\GETBASEFAT PNBASE B1)
                                                           (\GETBASEFAT BASE B2]
                                       [FATP                     ; The incoming string is fat, but there are no fat characters in the
                                                                 ; PNAME.
                                             (for B1 from 1 to LEN as B2 from OFFST
                                                always (EQ (\GETBASETHIN PNBASE B1)
                                                           (\GETBASEFAT BASE B2]
                                       (T                        ; Both the incoming string of chars and the PNAME are thin.
                                          (for B1 from 1 to LEN as B2 from OFFST
                                             always (EQ (\GETBASETHIN PNBASE B1)
                                                        (\GETBASETHIN BASE B2]
                                (LOCAL (STREQUAL (LOCAL (CL:SYMBOL-NAME BASE))
                                                 (SYMBOL.PNAME (SETQ ATM# (SUB1 HASHENT]
                           (UNLESSRDSYS (RETURN (\ADDBASE \ATOMSPACE (SUB1 ATM#)))
                                  (RETURN ATM#)))
                        (T                                       ; Doesn't match, so reprobe.  Want reprobe to be variable,
                                                                 ; preferably independent of primary probe.
                           [SETQ HASH (IPLUS16 HASH (OR REPROBE (SETQ REPROBE (ATOM.HASH.REPROBE HASH FIRSTBYTE]
                           (GO LP]                               ; Not found, must make new atom
          (RETURN (UNINTERRUPTABLY
                      (LET ((NEWATOM (\CREATE.SYMBOL BASE OFFST LEN FATP FATCHARSEENP)))
                           [UNLESSRDSYS (\PUTBASE \AtomHashTable HASH (ADD1 (\ATOMPNAMEINDEX NEWATOM]
                           NEWATOM))])
```

(LOOKUP-SYMBOL

```
  [LAMBDA (TABLE STRING SXHASH ENTRY-HASH)                       ; Edited 17-Feb-87 10:43 by raf
```

;;; Find where the symbol named String is stored in Table. Index is returned, or NIL if it is not present. Length and Hash are the length and sxhash of
;;; String. Entry-Hash is the entry-hash of the string and length."

```
    (LET* ((VEC (\GETBASEPTR TABLE 0))                           ; CL::PACKAGE-HASHTABLE-TABLE
           (HASH (\GETBASEPTR TABLE 2))                          ; CL::PACKAGE-HASHTABLE-HASH
```

```
          (LEN (FFETCH (ONED-ARRAY TOTAL-SIZE) OF VEC))          ; CL:ARRAY-TOTAL-SIZE
          [H2 (ADD1 (IREMAINDER SXHASH (IDIFFERENCE LEN 2]       ; REHASH-FACTOR

          )
         (DECLARE (TYPE (CL:SIMPLE-ARRAY (CL:UNSIGNED-BYTE 8))
                        HASH)
                  (TYPE (CL:SIMPLE-ARRAY (CL:UNSIGNED-BYTE 16))
                        VEC))
         (PROG ((INDEX-VAR (IREMAINDER SXHASH LEN))
                SYMBOL-NUMBER EHASH)
               (IF NIL
                   THEN (CL:FORMAT T "Probe @ ~s~%" INDEX-VAR))
          LOOP
               (SETQ EHASH (\GETBASEBYTE (FFETCH (ONED-ARRAY BASE) OF HASH)
                                 INDEX-VAR))                     ; CL:AREF
               [COND
                  [(EQL EHASH ENTRY-HASH)
                   (IF NIL
                       THEN (CL:FORMAT T "Entry hash MATCHES~%"))
                   (LET [(SYMBOL-NAME (SYMBOL.PNAME (SETQ SYMBOL-NUMBER (\GETBASE (FFETCH (ONED-ARRAY BASE)
                                                                                       OF VEC)
                                                                                 INDEX-VAR]
                                                                                 ; CL:AREF
                         (IF NIL
                             THEN (CL:FORMAT T "Got symbol index~%"))
                         ;; pname length is first byte of pname
                         (COND
                            ((LOCAL (STREQUAL SYMBOL-NAME STRING))
                             (IF NIL
                                 THEN (CL:FORMAT T " found~%"))
                             (GO DOIT))
                            (T (IF NIL
                                   THEN (CL:FORMAT T "Didn't match~%"]
                   ((EQL 0 EHASH)
                    (IF NIL
                        THEN (CL:FORMAT T "Hit deleted entry (no match)~%"))
                    (SETQ INDEX-VAR NIL)
                    (GO DOIT))
                   (T (IF NIL
                          THEN (CL:FORMAT T "Entry hash does not match~%"]
               (SETQ INDEX-VAR (IREMAINDER (IPLUS INDEX-VAR H2)
                                           LEN))                 ; SYMBOL-HASH-REPROBE
               (IF NIL
                   THEN (CL:FORMAT T "Reprobe @ ~s~%" INDEX-VAR))
               (GO LOOP)
          DOIT
               (RETURN SYMBOL-NUMBER])
```

(FIND.PACKAGE

```
  [LAMBDA (NAME)                                                 ; Edited  6-Mar-87 11:50 by raf
```

;;; Given a name, find the package with that name or nickname. This is a specialized, macroexpanded and de-optimized version of IL:GETHASH

```
    (PROG ((ITEM (LOCAL (MKSTRING NAME)))
           (HA READSYS.PACKAGE.FROM.NAME)
           BITS INDEX SLOT SKEY FIRSTINDEX REPROBE LIMIT ABASE VALUE)
          (SETQ BITS (STRINGHASHBITS ITEM))
          (SETQ INDEX (LOGAND BITS (ffetch (HARRAYP LASTINDEX) of HA)))
                                                                 ; \FIRSTINDEX
          (SETQ ABASE (ffetch HARRAYPBASE of HA))
          (SETQ FIRSTINDEX INDEX)
          (SETQ REPROBE (LOGOR (LOGAND (LOGXOR BITS (LRSH BITS 8))
                                       (IMIN 63 (FFETCH (HARRAYP LASTINDEX) OF HA)))
                               1))                               ; \REPROBE
          (SETQ LIMIT (ffetch (HARRAYP LASTINDEX) of HA))
      LP  (SETQ SLOT (\ADDBASE4 ABASE INDEX))                    ; \HASHSLOT
          (COND
             [(SETQ VALUE (ffetch (HASHSLOT VALUE) of SLOT))     ; Slot is occupied
              (SETQ SKEY (V\UNCOPY (ffetch (HASHSLOT KEY) of SLOT)))
              (COND
                 ((STREQUAL ITEM SKEY)                           ; Found it
                  (GO FOUND]
             ((NULL (ffetch (HASHSLOT KEY) of SLOT))             ; Empty slot
              (RETURN NIL)))
          (SETQ INDEX (LOGAND (IPLUS16 INDEX REPROBE)
                              LIMIT))                            ; Since table size is a power of two, any wraparound in the
                                                                 ; IPLUS16 will be consistent with the LOGAND
          (COND
             ((EQ INDEX FIRSTINDEX)                              ; Should never happen, since we don't allow full occupancy
              (SHOULDNT "Hashing in full hash table")))
          (GO LP)
      FOUND
          (RETURN (AND (NEQ VALUE \HASH.NULL.VALUE)
                       VALUE])
```

(FIND.SYMBOL

```
  [LAMBDA (STRING PACKAGE)                                    ; Edited 16-Feb-87 15:59 by raf
```

;;; Given a string, find a symbol by that name. This is macroexpanded and altered code from LLPACKAGE

```
    (LET* ((LENGTH (LOCAL (FFETCH (STRINGP LENGTH) OF STRING)))
           [HASH (COND
                     ((EQL 0 LENGTH)
                      0)
                     (T (PROG* ((TERMINUS LENGTH)
                                (HASH (LLSH (LOCAL (NTHCHARCODE STRING 1))
                                            8))
                                (CHAR# 2))
                           A0355
                               [COND
                                  ((IGREATERP CHAR# TERMINUS)
                                   (RETURN (PROGN HASH]
                               (PROGN)
                               [SETQ HASH (IPLUS16 (IPLUS16 (SETQ HASH (IPLUS16 HASH (LLSH (LOGAND HASH 4095)
                                                                                           2)))
                                                            (LLSH (LOGAND HASH 255)
                                                                  8))
                                                   (LOCAL (NTHCHARCODE STRING CHAR#]
                               (SETQ CHAR# (ADD1 CHAR#))
                               (GO A0355]                     ; SYMBOL-HASH
            (EHASH (IPLUS (IREMAINDER (LOGXOR LENGTH HASH (RSH HASH 8)
                                              (RSH HASH 16)
                                              (RSH HASH 19))
                                      254)
                          2))                                 ; ENTRY-HASH
            (SYM)
            (WHERE)
            (DONE))
           [COND
              ((NOT (\GETBASEPTR PACKAGE 14))                 ; CL::%PACKAGE-EXTERNAL-ONLY
               (IF NIL
                   THEN (PRINT "Checking INTERNAL symbols"))
               (LET ((INDEX (LOOKUP-SYMBOL (\GETBASEPTR PACKAGE 16)
                                STRING HASH EHASH)))          ; CL::%PACKAGE-INTERNAL-SYMBOLS
                    (COND
                       (INDEX (SETQ SYM INDEX)
                              (SETQ WHERE :INTERNAL)
                              (SETQ DONE T]
           [COND
              ((NOT DONE)
               (IF NIL
                   THEN (PRINT "Checking EXTERNAL symbols"))
               (LET ((INDEX (LOOKUP-SYMBOL (\GETBASEPTR PACKAGE 18)
                                STRING HASH EHASH)))          ; CL::%PACKAGE-INTERNAL-SYMBOLS
                    (COND
                       (INDEX (SETQ SYM INDEX)
                              (SETQ WHERE :EXTERNAL)
                              (SETQ DONE T]
           [COND
              ((NOT DONE)
               (IF NIL
                   THEN (CL:FORMAT T "Checking USE'd packages~%%"))
               (LET ((HEAD (\GETBASEPTR PACKAGE 2))           ; CL::%PACKAGE-TABLES

                     )
                    (PROG ((PREV HEAD)
                           (TABLE (CDR HEAD)))
                      USED-PACKAGE-LOOP
                          [COND
                             ((OR DONE (NULL TABLE))
                              (RETURN (PROGN (CL:VALUES NIL NIL]
                          [PROGN (LET ((INDEX (LOOKUP-SYMBOL (CAR TABLE)
                                                             STRING HASH EHASH)))
                                                             ; CL::%PACKAGE-INTERNAL-SYMBOLS
                                       (COND
                                          (INDEX (COND
                                                    ((NEQ PREV HEAD)
                                                     (LET* ((A0347 PREV)
                                                            (A0346 (CDR A0347))
                                                            (A0349 TABLE)
                                                            (A0348 (CDR A0349))
                                                            (A0351 HEAD)
                                                            (A0350 (CDR A0351)))
                                                           (CDR (RPLACD A0347 A0348))
                                                           (CDR (RPLACD A0349 A0350))
                                                           (CDR (RPLACD A0351 TABLE))
                                                           A0346)))
                                                 (SETQ SYM INDEX)
                                                 (SETQ WHERE :INHERITED)
                                                 (SETQ DONE T))
                                          (T]
```

```
                        (PROGN (SETQ PREV (PROG1 TABLE
                                                 (PROGN (SETQ TABLE (CDR TABLE))
                                                        NIL)))
                               NIL)
                        (GO USED-PACKAGE-LOOP]
          (LOCAL (CL:VALUES SYM WHERE])
```

(**PACKAGE.NAME**

```
  [LAMBDA (RMPKG)                                          ; Edited 12-Feb-87 17:29 by raf
    (AND RMPKG (\UNCOPY (\GETBASEPTR RMPKG 4])

)

;; See \PNAMELIMIT comment below

(RPAQQ \PNAMELIMIT 255)

(RPAQ? \PNAMES.IN.BLOCKS? )

;; Flag for the closure cache

(RPAQ? SI::*CLOSURE-CACHE-ENABLED* )

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS SI::*CLOSURE-CACHE-ENABLED*)
)

(DEFINEQ
```

(**\DEFINEDP**

```
  [LAMBDA (A)                                              (* edited%: " 3-Apr-85 19:45")
    (AND (LITATOM A)
         (fetch (LITATOM DEFPOINTER) of A)
         T])
```

(**PUTD**

```
  [LAMBDA (FN DEF FLG)                                     ; Edited  5-Aug-93 10:19 by sybalskY:MV:ENVOS
    (PROG1 DEF
        [COND
           ((NOT (LITATOM FN))
            (\ILLEGAL.ARG FN))
           ([NOT (OR (LISTP DEF)
                     (NULL DEF)
                     (TYPEP DEF 'COMPILED-CLOSURE]
            (\ILLEGAL.ARG DEF))
           ((AND (NULL FLG)
                 (TYPEP DEF 'COMPILED-CLOSURE)
                 (NEQ (fetch (COMPILED-CLOSURE FRAMENAME) of DEF)
                      FN))                                 ; Definition being stored has a different frame name, so fix it
            (SETQ DEF (\RENAMEDFN DEF FN]
        (\PUTD FN DEF))])
```

(**\PUTD**

```
  [LAMBDA (FN DEF)                                         (* lmm " 7-Nov-86 03:54")
    (LET ((DCELL (fetch (LITATOM DEFINITIONCELL) of FN)))
         (UNINTERRUPTABLY
             (PROG ((DVAL DEF)
                    CODEBASE)
                   (COND
                      [(TYPEP DVAL 'COMPILED-CLOSURE)
                       (SETQ CODEBASE (fetch (COMPILED-CLOSURE FNHEADER) of DVAL))
                       (replace (DEFINITIONCELL PSEUDOCODEP) of DCELL with NIL)
                       (COND
                          ((fetch (COMPILED-CLOSURE ENVIRONMENT) of DVAL)
                                                           ; Full closure, have to store it as non-ccodep
                           (replace CCODEP of DCELL with NIL)
                           (GO CLOSURE))
                          (T                               ; Strip out code base
                             (SETQ DVAL CODEBASE]
                      ((AND (ARRAYP DVAL)
                            (EQ (fetch (ARRAYP TYP) of DVAL)
                                \ST.CODE))                 ; Code array -- only from the code reader or compiler
                       (SETQ CODEBASE (SETQ DVAL (fetch (ARRAYP BASE) of DVAL)))
                       (replace (DEFINITIONCELL PSEUDOCODEP) of DCELL with NIL))
                      (T (GO EXPR)))
               CODE
                    (replace (DEFINITIONCELL CCODEP) of DCELL with T)
               CLOSURE
                    (replace (DEFINITIONCELL ARGTYPE) of DCELL with (fetch (FNHEADER ARGTYPE) of CODEBASE))
                    (replace (DEFINITIONCELL FASTP) of DCELL with (EQ 0 (fetch (FNHEADER NTSIZE) of CODEBASE)))
                    (replace (DEFINITIONCELL DEFPOINTER) of DCELL with DVAL)
                    (RETURN DEF)
```

```
              EXPR
                  (replace (DEFINITIONCELL DEFCELLFLAGS) of DCELL with 0)
                  (replace (DEFINITIONCELL DEFPOINTER) of DCELL with DVAL)
                  (RETURN DEF)))])
```

(GETD

```
  [LAMBDA (A)                                                    ; Edited  7-Jan-88 15:47 by jop
    (IF (LITATOM A)
        THEN (LET* ((A (fetch (LITATOM DEFINITIONCELL) of A))
                    (DEF (fetch (DEFINITIONCELL DEFPOINTER) of A)))
                   (COND
                      ((NOT (fetch (DEFINITIONCELL CCODEP) of A))
                       DEF)
                      (SI::*CLOSURE-CACHE-ENABLED* (SI::GET-CACHE-CLOSURE DEF))
                      (T (create COMPILED-CLOSURE
                                 FNHEADER _ DEF])
```

(PUTDEFN

```
  [LAMBDA (FN CA SIZE)                                           (* edited%: " 3-Apr-85 19:55")
                                                                 ; special version of PUTD that runs only at MAKEINIT time
    (PROG ((DCELL (fetch (LITATOM DEFINITIONCELL) of FN))
           [BLOCKINFO (PROGN ;; Reserve enough space.  FILECODEBLOCK leaves file pointing at first data word, so BASE is set to that below.
                             ;; BLOCKINFO is used for setting block trailer.
                             (FILECODEBLOCK (FOLDHI SIZE BYTESPERCELL)
                                    (fetch (CODEARRAY ALIGNED) of CA]
           (BASE (FILEARRAYBASE)))
          (replace (DEFINITIONCELL DEFPOINTER) of DCELL with BASE)
          (replace (DEFINITIONCELL ARGTYPE) of DCELL with (fetch (CODEARRAY ARGTYPE) of CA))
          (replace (DEFINITIONCELL FASTP) of DCELL with (EQ (fetch (CODEARRAY NTSIZE) of CA)
                                                            0))
          (replace (DEFINITIONCELL CCODEP) of DCELL with T)
          (replace (DEFINITIONCELL PSEUDOCODEP) of DCELL with NIL)
          [COND
             ((FMEMB FN LOCKEDFNS)
              (\LOCKCELL DCELL 1)
              (\LOCKCELL BASE (FOLDHI (IPLUS (fetch (POINTER WORDINPAGE) of BASE)
                                             (FOLDHI SIZE BYTESPERWORD))
                                      WORDSPERPAGE]
          [COND
             ((EQ FN (LOCAL (FUNCTION \RESETSTACK)))             ; special kludge to remember where \RESETSTACK is in the
                                                                 ; MAKEINIT
              (SETQ RESETPTR (FILEARRAYBASE))
              (SETQ RESETPC (fetch (CODEARRAY STARTPC) of CA]
          (AOUT CA 0 SIZE OUTX 'CODE)
          (BOUTZEROS (MODUP SIZE BYTESPERCELL))
          (FILEBLOCKTRAILER BLOCKINFO])
```

(GETDEFN

```
  [LAMBDA (A)                                                    (* lmm "20-AUG-81 12:17")
    (fetch (LITATOM DEFPOINTER) of A])

)

(DEFINEQ
```

(\STKMIN

```
  [LAMBDA (CODE CODEISBLOCK PRINT)
    (DECLARE (LOCALVARS . T))                                    ; Edited 10-Nov-88 17:01 by jds
    ;; compute minimum stack space to run in this function, for either D-machine (which checks at every opcode) or Maiko (which only checks at a
    ;; selected number of opcodes.

    ;; this function is tightly coded because it is executed every function loaded

    (ALLOCAL
     (PROGN
      ;; can be run renamed but will work on local space.

      [if (NOT \OPSTACKEFFECT)
          then (SETQ \OPSTACKEFFECT (\ALLOCBLOCK (FOLDHI 256 BYTESPERCELL)))
               (SETQ \OPLENGTH (\ALLOCBLOCK (FOLDHI 256 BYTESPERCELL)))
               [for I from 0 to 255
                  do (\PUTBASEBYTE \OPSTACKEFFECT I
                                   (- 2 (LET ((OP (\FINDOP I))
                                              LEVADJ)
                                             (SELECTQ (fetch (OPCODE OPCODENAME)
                                                             OP)
                                                  ((FN0 FN1 FN2 FN3 FN4 FNX SWAP NOP APPLYFN RETURN)
                                                      2)
                                                  ((UNBIND DUNBIND UNWIND POP.N)
                                                      -1)
                                                  ((BIND SUBRCALL MISCN)
                                                      1)
                                                  (OR (NUMBERP (if (LISTP (SETQ LEVADJ (fetch (OPCODE LEVADJ)
                                                                                              OP)))
```

```
                                                                then (SETQ LEVADJ (CAR LEVADJ))
                                                               else LEVADJ))
                                                        (SELECTQ LEVADJ
                                                             ((CJUMP NCJUMP)
                                                                              ; these only check if they jump
                                                                 -1)
                                                             ((JUMP)
                                                                 2)
                                                             (PROGN 2]
                    (for I from 0 to 255 do (\PUTBASEBYTE \OPLENGTH I (ADD1 (OR (CADDR (\FINDOP I))
                                                                                -1]
        [IF (NOT CODEISBLOCK)
            THEN (SETQ CODE (OR (\GET-COMPILED-CODE-BASE CODE)
                                (fetch (ARRAYP BASE)
                                       CODE]
        (LLSH (PROG (MAX OP STKE (PC (fetch (FNHEADER STARTPC)
                                             CODE))
                         (DEPTH (IPLUS (IMAX (fetch (FNHEADER NA) of CODE)
                                             0)
                                       8
                                       (UNFOLD (ADD1 (fetch (FNHEADER PV) of CODE))
                                               CELLSPERQUAD)
                                       4)))
                    (SETQ MAX (PLUS DEPTH 8)))
                    ;; this PROG computes the depth in cells. The llsh around converts it to D-machine words.
                    ;; the initial maximum is the actual size of the frame, plus 4 extra cells for space to store info in case of an overflow. The default
                    ;; maximum is 8 more than that. By walking the code, it finds if there are any other runs that would increase it beyond that. At jumps or
                    ;; "Maiko check" opcodes, the depth is reset to 0.
                     LP  (if (EQ 0 (SETQ OP (\GETBASEBYTE CODE PC)))
                             then ;; end of the function
                                  (RETURN MAX))
                    ;; the following is for debugging
                          (AND PRINT (CL:FORMAT T "~%%~3o: ~3o d<~3d> mx<~3d>" PC OP DEPTH MAX))
                          (SELECTQ (SETQ STKE (- 2 (\GETBASEBYTE \OPSTACKEFFECT OP)))
                               (2 ;; special code indicating that this opcode checks the stack level
                                  (AND PRINT (PRIN1 "*"))
                                  (SETQ DEPTH 0))
                               (add DEPTH STKE))
                          (if (GREATERP DEPTH MAX)
                              then (SETQ MAX DEPTH))
                          (CL:INCF PC (\GETBASEBYTE \OPLENGTH OP))
                          (GO LP))
                    1])

)

(RPAQ? \OPSTACKEFFECT )

(RPAQ? \OPLENGTH )

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \OPSTACKEFFECT \OPLENGTH)
)

(RPAQQ COMPILEATPUTDFLG NIL)

(DECLARE%: DONTCOPY

;; FOLLOWING DEFINITIONS EXPORTED

(DECLARE%: EVAL@COMPILE

[ACCESSFNS LITATOM ((DEFINITIONCELL (\DEFCELL DATUM))
                    (PROPCELL (\PROPCELL DATUM))
                    (VCELL (\VALCELL DATUM))
                    (PNAMECELL (\PNAMECELL DATUM)))
        ;; VCELL can also be accessed directly from a value index via the record VALINDEX (as in \SETGLOBALVAL.UFN) --- Similarly,
        ;; PNAMEINDEX accesses PNAMECELL for use by \MKATOM and UNCOPYATOM
        (TYPE? (LITATOM DATUM))
        (BLOCKRECORD PROPCELL ((NIL BITS 4)                    ; former flags locations
                               (PROPLIST POINTER)
                               (NIL BITS 8)                    ; Package byte
                               (NIL BITS 8)                    ; Flags from defcell
                               ;; PROPCell flags:
                               (NIL BITS 1)
                               (GENSYMP FLAG)
                               (FATPNAMEP FLAG)
                               (NIL BITS 5)
                               ;; Filler for final cell:
```

```
                              (NIL BITS 8]

(SYNONYM CL:SYMBOL (LITATOM))

[ACCESSFNS VALINDEX ((VCELL (COND
                              [(AND (FIXP DATUM)
                                    (ILESSP DATUM 65535))       ; Xerox Lisp traditional symbol
                               (\ADDBASE2 \PNPSPACE (IPLUS \NEWATOM-VALOFFSET (ITIMES 10 DATUM]
                              (T                                ; New symbol
                                                                ; '90/07/19 ON
                                 (\ADDBASE DATUM \NEWATOM-VALOFFSET]

(BLOCKRECORD VCELL ((VALUE FULLPOINTER)))

[BLOCKRECORD DEFINITIONCELL ((CCODEP FLAG)
                             (FASTP FLAG)
                             (ARGTYPE BITS 2)                   ; Former flag location
                             (DEFPOINTER POINTER)
                             (NIL POINTER)                      ; Proplist cell
                             (NIL BITS 8)                       ; package
                             ;; DEFCELL flags overflow from top 4 bits of the real cell:
                             (NIL BITS 4)
                             (PSEUDOCODEP FLAG)
                             (NIL BITS 3)
                             ;; proplist falgs and filler:
                             (NIL BITS 16))
       (BLOCKRECORD DEFINITIONCELL ((DEFCELLFLAGS BITS 4)
                                    (NIL POINTER)               ; defn ptr
                                    (NIL BITS 4)
                                    (NIL POINTER)               ; filler for proplist ptr
                                    (NIL BITS 8)
                                    (AUXDEFCELLFLAGS BYTE)
                                    (NIL BITS 16]

[BLOCKRECORD FNHEADER ((STKMIN WORD)
                       (NA SIGNEDWORD)
                       (PV SIGNEDWORD)
                       (STARTPC WORD)
                       (CLOSUREP FLAG)                          ; T if this is a "compiled closure"
                       (BYTESWAPPED FLAG)                       ; T if, on 386, we reswapped the code section of this function for
                                                                ; faster access.
                       (ARGTYPE BITS 2)                         ; 0 = LAMBDA
                                                                ; 2 = LAMBDA nospread
                                                                ; 1 = NLAMBDA
                                                                ; 3 = NLAMBDA nospread
                       ;; 4 NIL BITS USED TO BE HERE.
                       (%#FRAMENAME XPOINTER)
                       (NTSIZE WORD)                            ; Size of the Name Table, IN WORDS.  This value is always
                                                                ; rounded up to the next Quad-word in size, and there'
                                                                ; guaranteed to be one entry of zeros in the length.
                       (NLOCALS BYTE)
                       (FVAROFFSET BYTE))
       (ACCESSFNS FNHEADER ((LSTARP (ILESSP (fetch (FNHEADER NA) of DATUM)
                                            0))
                            (OVERHEADWORDS (PROGN 8))
                            (NATIVE (PROGN NIL))                ; T if this is a NATIVE-code function (never true!)
                            (ALIGNED (IPLUS (fetch (FNHEADER NTSIZE) of DATUM)
                                            (fetch (FNHEADER OVERHEADWORDS) of T)))
                            (FIXED NIL (replace (FNHEADER STKMIN) of DATUM with (\STKMIN DATUM T)))
                            (NPVARWORDS (UNFOLD (ADD1 (fetch (FNHEADER PV) of DATUM))
                                                WORDSPERQUAD))
                            (FRAMENAME (fetch (FNHEADER %#FRAMENAME) of DATUM)
                                   (UNINTERRUPTABLY
                                       (CHECK (NEQ (\HILOC DATUM)
                                                   \STACKHI))
                                       (\DELREF (fetch (FNHEADER %#FRAMENAME) of DATUM))
                                       (\ADDREF NEWVALUE)
                                       (replace (FNHEADER %#FRAMENAME) of DATUM with NEWVALUE))]

[BLOCKRECORD PNAMECELL ((NIL BITS 4)
                        (PNAMEBASE XPOINTER)
                        (NIL POINTER)                           ; val, def, prop cells
                        (NIL POINTER)
                        (NIL POINTER)
                        (PACKAGEINDEX BYTE)
                        (NIL BITS 24)                           ; filler for other flags

                       )
       (BLOCKRECORD PNAMECELL ((FULLPNAMEBASE FULLXPOINTER)     ; Replacing this smashes PACKAGEINDEX to 0

                               ))
       (ACCESSFNS PNAMECELL ((PACKAGE [LET ((I (FETCH (PNAMECELL PACKAGEINDEX) OF DATUM)))
                                                                ; This ugly construct allows cl:symbol-package to run in the init,
                                                                ; where *PACKAGE-FROM-INDEX* is not yet bound.
```

```
                                        (COND
                                           ((EQ 0 I)
                                            NIL)
                                           (T (CL:AREF *PACKAGE-FROM-INDEX* I]
                                    (REPLACE (PNAMECELL PACKAGEINDEX) OF DATUM WITH (IF (NULL NEWVALUE)
                                                                                        THEN
                                                                                    *UNINTERNED-PACKAGE-INDEX*
                                                                                     ELSE (
                                                                                           CL::%%PACKAGE-INDEX
                                                                                            NEWVALUE]

[ACCESSFNS PACKAGEINDEX ((PACKAGE (IF (EQ 0 DATUM)                ; This ugly construct allows cl:symbol-package to run in the init,
                                                                  ; where *PACKAGE-FROM-INDEX* is not yet bound.
                                      THEN NIL
                                    ELSE (CL:AREF *PACKAGE-FROM-INDEX* DATUM]

(BLOCKRECORD PNAMEBASE ((PNAMELENGTH BYTE)                        ; Length is always here, be the pname thin or fat
                        (PNAMEFATPADDINGBYTE BYTE)                ; This byte is zero for fat pnames so that the pname chars are
                                                                  ; word-aligned

                        ))

[ACCESSFNS PNAMEINDEX ((PNAMECELL (COND
                                    [(AND (FIXP DATUM)
                                          (ILESSP DATUM 65535)) ; Xerox Lisp traditional symbol
                                     (\ADDBASE \PNPSPACE (IPLUS \NEWATOM-PNAMEOFFSET (ITIMES 10 DATUM]
                                    (T                            ; New symbol
                                                                  ; '90/07/19 ON
                                       (\ADDBASE DATUM \NEWATOM-PNAMEOFFSET]
)

(DECLARE%: EVAL@COMPILE

(BLOCKRECORD NEW-ATOM (;; An extended symbol, for expanding atom space.  Kept in its own datatype.
                       (PNAME XPOINTER)                           ; PNAME, same as litatom.
                       (VALUE POINTER)
                       (DEF POINTER)
                       (PROPLIST POINTER)
                       ;; Flags that used to be above the pointers, e.g. package, ccodep, gensymp:
                       (NIL BITS 32)))
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS \DEFCELL MACRO ((ATOM)
                          (\ATOMCELL ATOM \DEF.HI)))

(PUTPROPS \VALCELL MACRO ((ATOM)
                          (\ATOMCELL ATOM \VAL.HI)))

(PUTPROPS \PNAMECELL MACRO ((ATOM)
                            (\ATOMCELL ATOM \PNAME.HI)))
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS \ATOMVALINDEX DMACRO [OPENLAMBDA (X)
                                  (COND
                                     ((EQ (NTYPX X)
                                          \LITATOM)               ; Original litatoms
                                      (\LOLOC X))
                                     ((EQ (NTYPX X)
                                          \NEW-ATOM)              ; new 3-byte symbols
                                      X)
                                     (T (SHOULDNT])

(PUTPROPS \ATOMDEFINDEX DMACRO [OPENLAMBDA (X)
                                  (COND
                                     ((EQ (NTYPX X)
                                          \LITATOM)               ; Original litatoms
                                      (\LOLOC X))
                                     ((EQ (NTYPX X)
                                          \NEW-ATOM)              ; new 3-byte symbols
                                      X)
                                     (T (SHOULDNT])

(PUTPROPS \ATOMPNAMEINDEX DMACRO [OPENLAMBDA (X)
                                    (COND
                                       ((EQ (NTYPX X)
                                            \LITATOM)             ; Original litatoms
                                        (\LOLOC X))
                                       ((EQ (NTYPX X)
                                            \NEW-ATOM)            ; new 3-byte symbols
                                        X)
                                       (T (SHOULDNT])
```

```
(PUTPROPS \ATOMPROPINDEX DMACRO [(X)
                                 (COND
                                    ((EQ (NTYPX X)
                                         \LITATOM)            ; Original litatoms
                                     (\LOLOC X))
                                    ((EQ (NTYPX X)
                                         \NEW-ATOM)           ; new 3-byte symbols
                                     X)
                                    (T (SHOULDNT])

(PUTPROPS \INDEXATOMPNAME DMACRO (OPENLAMBDA (X)
                                   (COND
                                      [(FIXP X)               ; Xerox Lisp traditional symbol
                                       (COND
                                          ((SMALLP X)
                                           (\VAG2 \AtomHI X))
                                          (T (\VAG2 (LRSH X 16)
                                                    (LOGAND X 65535]
                                      (T                      ; New symbol
                                         X))))

(PUTPROPS \INDEXATOMVAL DMACRO (OPENLAMBDA (X)
                                 (COND
                                    [(FIXP X)                 ; Xerox Lisp traditional symbol
                                     (COND
                                        ((SMALLP X)
                                         (\VAG2 \AtomHI X))
                                        (T (\VAG2 (LRSH X 16)
                                                  (LOGAND X 65535]
                                    (T                        ; New symbol
                                       X))))

(PUTPROPS \INDEXATOMDEF DMACRO (OPENLAMBDA (X)
                                 (COND
                                    [(FIXP X)                 ; Xerox Lisp traditional symbol
                                     (COND
                                        ((SMALLP X)
                                         (\VAG2 \AtomHI X))
                                        (T (\VAG2 (LRSH X 16)
                                                  (LOGAND X 65535]
                                    (T                        ; New symbol
                                       X))))

(PUTPROPS \ATOMNUMBER DMACRO (= . \LOLOC))
)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \NxtPnByte \CurPnPage \NxtAtomPage \AtomFrLst \OneCharAtomBase \PNAMES.IN.BLOCKS? \SCRATCHSTRING
       COMPILEATPUTDFLG)
)

(DECLARE%: EVAL@COMPILE

(RPAQQ \PNAMELIMIT 255)

(RPAQQ \CharsPerPnPage 512)

(CONSTANTS (\PNAMELIMIT 255)
       (\CharsPerPnPage 512))
)

(DECLARE%: EVAL@COMPILE

(RPAQQ \NEWATOM-PNAMEOFFSET 0)

(RPAQQ \NEWATOM-VALOFFSET 2)

(RPAQQ \NEWATOM-DEFOFFSET 4)

(RPAQQ \NEWATOM-PLISTOFFSET 6)

(RPAQQ \NEWATOM-TYPE# 21)

(CONSTANTS (\NEWATOM-PNAMEOFFSET 0)
       (\NEWATOM-VALOFFSET 2)
       (\NEWATOM-DEFOFFSET 4)
       (\NEWATOM-PLISTOFFSET 6)
       (\NEWATOM-TYPE# 21))
)
)

;; END EXPORTED DEFINITIONS

(DECLARE%: EVAL@COMPILE DONTCOPY
```

```
(DECLARE%: EVAL@COMPILE

(PUTPROPS COMPUTE.ATOM.HASH MACRO [(BASE OFFST LEN FIRSTBYTE FATP)
                                                        ; Sets variable HASH to atom hash of indicated string
                                   (SETQ HASH (LLSH FIRSTBYTE 8))
                                   (for CHAR# from (ADD1 OFFST) to (SUB1 (IPLUS OFFST LEN))
                                      do (SETQ HASH (IPLUS16 (IPLUS16 (SETQ HASH
                                                                            (IPLUS16 HASH (LLSH (LOGAND HASH 4095)
                                                                                                2)))
                                                                      (LLSH (LOGAND HASH 255)
                                                                            8))
                                                               (UNLESSRDSYS (COND
                                                                               (FATP (LOGAND (\GETBASEFAT BASE
                                                                                                          CHAR#)
                                                                                             255))
                                                                               (T (\GETBASETHIN BASE CHAR#)))
                                                                            (NTHCHARCODE BASE CHAR#])

(PUTPROPS ATOM.HASH.REPROBE MACRO [(HASH FIRSTBYTE)
                                   (LOGAND 63 (LOGOR 1 (LOGXOR FIRSTBYTE HASH])
)

(ADDTOVAR DONTCOMPILEFNS INITATOMS COPYATOM UNCOPYATOM READATOM MAKE.LOCAL.ATOM SYMBOL.VALUE SYMBOL.PNAME
                         SYMBOL.PACKAGE OLD.FIND.SYMBOL LOOKUP-SYMBOL FIND.PACKAGE FIND.SYMBOL
                         PACKAGE.NAME GETDEFN PUTDEFN FSETVAL)
)

;; for executing boot expressions when first run

(DEFINEQ

(\RESETSYSTEMSTATE
  [LAMBDA NIL                                                   (* rmk%: " 5-JUN-81 17:32")
    (\KEYBOARDON T)
    (\RESETTERMINAL])


(INITIALEVALQT
  [LAMBDA NIL                                                   (* bvm%: "21-APR-83 12:02")
    (DECLARE (GLOBALVARS BOOTFILES))
    (\SETIOPOINTERS)
    (PROG ((RL BOOTFILES)
           FL L)
          (OR RL (RETURN))
          (SIMPLEPRINT "evaluating initial expressions:
                ")                                              ; BOOTFILES is the list of boot files in reverse order
      R   (SETQ FL (CONS (CAR RL)
                         FL))
          (COND
             ((SETQ RL (CDR RL))
              (GO R)))
      L1  [COND
             ([LISTP (SETQ L (GETTOPVAL (CAR FL]
              (SIMPLEPRINT (CAR FL))                            (* Print the name of the bootfile)
              (DSPBOUT (CHARCODE CR))
              (PROG NIL
                L2  [EVAL (PROG1 (CAR L)
                                 (SETTOPVAL (CAR FL)
                                            (SETQ L (CDR L))))]
                    (AND (LISTP L)
                         (GO L2)))
              (SETTOPVAL (CAR FL)
                         'NOBIND]
          (COND
             ((SETQ FL (CDR FL))
              (GO L1)))
          (SETQ BOOTFILES NIL)
          (INTERPRET.REM.CM)                                    (* See if command line has anything to say)
      )                                                         ; Value is T so that correct value is returned when this is called
                                                                ; from within COPYSYS0
    T])


(SIMPLEPRINT
  [LAMBDA (X N)                                                 (* bvm%: "13-Feb-85 22:25")
    (COND
       [(OR (LITATOM X)
            (STRINGP X))
        (for I from 1 to (NCHARS X) do (DSPBOUT (NTHCHARCODE X I]
       ((LISTP X)
        (COND
           ((EQ N 0)
            (SIMPLEPRINT "&"))
           (T (DSPBOUT (CHARCODE %())
              (PROG NIL
                LP  [SIMPLEPRINT (CAR X)
                                 (SETQ N (COND
```

```
                                    ((SMALLPOSP N)
                                     (SUB1 N))
                                    (T 3]
                     (COND
                        ((EQ N 0)
                         (SIMPLEPRINT " --)"))
                        ((NULL (SETQ X (CDR X)))
                         (SIMPLEPRINT ")"))
                        ((NLISTP X)
                         (SIMPLEPRINT " . ")
                         (SIMPLEPRINT X)
                         (SIMPLEPRINT ")"))
                        (T (SIMPLEPRINT " ")
                           (GO LP])

)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS RESETFORMS BOOTFILES)
)
```

;; stats

```
(DEFINEQ
```

(PAGEFAULTS

```
  [LAMBDA NIL                                                    (* rrb "13-NOV-80 15:36")
    (DECLARE (GLOBALVARS \MISCSTATS))
    (fetch PAGEFAULTS of \MISCSTATS])
```

(\SETTOTALTIME

```
  [LAMBDA NIL                                                    (* JonL "17-Dec-83 00:23")
                                                                 ; updates the total time field of the misc stats page.
    (\BOXIPLUS (LOCF (fetch TOTALTIME of \MISCSTATS))
           (CLOCKDIFFERENCE (fetch STARTTIME of \MISCSTATS])
```

(\SERIALNUMBER

```
  [LAMBDA NIL                                                    (* rmk%: " 9-JUN-81 14:49")
    (fetch (IFPAGE SerialNumber) of \InterfacePage])

)
```

;; Fast functions for moving and clearing storage

```
(DEFINEQ
```

(\BLT

```
  [LAMBDA (DBASE SBASE NWORDS)                                   (* lmm "30-Mar-85 05:43")
                                                                 ; Generally in ucode -- must guarantee transferral by moving
                                                                 ; high-order address first

    (PROG [(NN (CONSTANT (EXPT 2 14]
          (RETURN (COND
                     ((GREATERP NWORDS NN)                       ; dorado has microcode only for up to 2^15
                      (\BLT (\ADDBASE DBASE NN)
                            (\ADDBASE SBASE NN)
                            (DIFFERENCE NWORDS NN))
                      (\BLT DBASE SBASE NN))
                     (T (for I from (SUB1 NWORDS) by -1 to 0 do (\PUTBASE DBASE I (\GETBASE SBASE I)))
                        DBASE])
```

(\MOVEBYTES

```
  [LAMBDA (SBASE SBYTE DBASE DBYTE NBYTES)                       (* rmk%: "23-OCT-82 14:24")
                                                                 ; Simple version for bootstrapping

    (COND
       ((IGREATERP NBYTES 0)
        (PROG ((SB (\ADDBASE SBASE (FOLDLO SBYTE BYTESPERWORD)))
               (DB (\ADDBASE DBASE (FOLDLO DBYTE BYTESPERWORD)))
               SBN DBN NWORDS)
              (COND
                 [(EQ (SETQ SBN (IMOD SBYTE BYTESPERWORD))
                      (SETQ DBN (IMOD DBYTE BYTESPERWORD)))      ; Can move words
                  (COND
                     ((EQ SBN 1)
                      (\PUTBASEBYTE DB 1 (\GETBASEBYTE SB 1))
                      (SETQ DB (\ADDBASE DB 1))
                      (SETQ SB (\ADDBASE SB 1))
                      (add NBYTES -1)))
                  (\BLT DB SB (SETQ NWORDS (FOLDLO NBYTES BYTESPERWORD)))
                  (COND
                     ((EQ (IMOD NBYTES BYTESPERWORD)
                          1)
                      (\PUTBASEBYTE (\ADDBASE DB NWORDS)
```

```
                              0
                              (\GETBASEBYTE (\ADDBASE SB NWORDS)
                                     0]
                    (T (FRPTQ NBYTES (\PUTBASEBYTE DB (PROG1 DBN (add DBN 1))
                                                (\GETBASEBYTE SB (PROG1 SBN (add SBN 1])
```

(\CLEARWORDS

```
  [LAMBDA (BASE NWORDS)                                          (* bvm%: "20-Feb-85 12:30")
    (PROG1 BASE
         (while (IGREATERP NWORDS 32767) do ;; BLT wants NWORDS to be small.  We play it safe by keeping the count smaller than 2^15,
                                            ;; avoiding a Dorado uCode bug
                                            (.CLEARNWORDS. BASE 32768)
                                            (SETQ BASE (\ADDBASE BASE 32768))
                                            (SETQ NWORDS (IDIFFERENCE NWORDS 32768)))
         (COND
            ((IGREATERP NWORDS 0)
             (.CLEARNWORDS. BASE NWORDS))))])
```

(\CLEARBYTES

```
  [LAMBDA (BASE OFFST NBYTES)                                    (* bvm%: "29-Jan-85 18:56")
    (COND
       ((IGREATERP NBYTES 0)
        (COND
           ((ODDP OFFST)
            (\PUTBASEBYTE BASE OFFST 0)
            (add OFFST 1)
            (add NBYTES -1)))                                    ; OFFST is now even
        (SETQ BASE (\ADDBASE BASE (FOLDLO OFFST BYTESPERWORD)))
        (COND
           ((ODDP NBYTES)                                        ; Final byte to be zeroed
            (\PUTBASEBYTE BASE (SUB1 NBYTES)
                   0)))                                          ; Now all we have to do is zero the word-aligned part in the
                                                                 ; middle
        (\CLEARWORDS BASE (FOLDLO NBYTES BYTESPERWORD])
```

(\CLEARCELLS

```
  [LAMBDA (BASE NCELLS)                                          (* bvm%: "20-Feb-85 12:51")
    [while (IGEQ NCELLS (FOLDLO 32767 WORDSPERCELL)) do          ; Keep the BLTs small.  See \CLEARWORDS
                                                        (.CLEARNWORDS. BASE 32768)
                                                        (SETQ BASE (\ADDBASE BASE 32768))
                                                        (SETQ NCELLS (IDIFFERENCE NCELLS (FOLDLO 32768
                                                                                                 WORDSPERCELL]
    (COND
       ((IGREATERP NCELLS 0)
        (SETQ NCELLS (UNFOLD NCELLS WORDSPERCELL))
        (.CLEARNWORDS. BASE NCELLS])

)

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(PUTPROPS .CLEARNWORDS. MACRO (OPENLAMBDA (BASE NWORDS)
                                  ;; Clear NWORDS words starting at base.  Assumes NWORDS is smallp and greater than zero.  Compiler
                                  ;; refuses to optimize out an IGREATERP test here, so push back to caller
                                  (\PUTBASE BASE (SUB1 NWORDS)
                                        0)
                                  [COND
                                     ((NEQ NWORDS 1)
                                      (\BLT BASE (\ADDBASE BASE 1)
                                            (SUB1 NWORDS]
                                  NIL))
)
)

;; Obsolete

(DECLARE%: EVAL@COMPILE DONTCOPY

;; FOLLOWING DEFINITIONS EXPORTED

(DECLARE%: EVAL@COMPILE

(PUTPROPS \MOVEWORDS MACRO (OPENLAMBDA (SBASE SOFFSET DBASE DOFFSET NWORDS)
                               (\BLT (\ADDBASE DBASE DOFFSET)
                                     (\ADDBASE SBASE SOFFSET)
                                     NWORDS)))
)
)
```

;; END EXPORTED DEFINITIONS

```
(DEFINEQ

(\MOVEWORDS
  [LAMBDA (SBASE SOFFSET DBASE DOFFSET NWORDS)                    (* bvm%: "15-JUN-82 13:56")
    (\BLT (\ADDBASE DBASE DOFFSET)
          (\ADDBASE SBASE SOFFSET)
          NWORDS])

(\ZEROBYTES
  [LAMBDA (BASE FIRST LAST)                                       (* bvm%: "29-Jan-85 19:12")
    (\CLEARBYTES BASE FIRST (ADD1 (IDIFFERENCE LAST FIRST])

(\ZEROWORDS
  [LAMBDA (BASE ENDBASE)                                          (* bvm%: "29-Jan-85 12:54")
    (while (IGREATERP (\HILOC ENDBASE)
                      (\HILOC BASE))
       do (\CLEARWORDS BASE (IDIFFERENCE (SUB1 WORDSPERSEGMENT)
                                         (\LOLOC BASE)))
          (\PUTBASE (\VAG2 (\HILOC BASE)
                           (SUB1 WORDSPERSEGMENT))
                 0 0)                                             ; Done this way to avoid non-SMALLP arithmetic when (\LOLOC
                                                                  ; BASE) = 0
          (SETQ BASE (\VAG2 (ADD1 (\HILOC BASE))
                            0)))
    (PROG [(DIF (IDIFFERENCE (\LOLOC ENDBASE)
                             (\LOLOC BASE]
          (COND
             ((IGEQ DIF 0)
              (\PUTBASE BASE 0 0)
              (\CLEARWORDS (\ADDBASE BASE 1)
                     DIF])

)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(LOCALVARS . T)
)

(DECLARE%: DONTCOPY

(ADDTOVAR INITVALUES (\AtomFrLst 0))

(ADDTOVAR INITPTRS (\OneCharAtomBase NIL)
                   (\SCRATCHSTRING))

(ADDTOVAR INEWCOMS (FNS FSETVAL SETPROPLIST PUTDEFN \BLT)
                   (FNS \MKATOM \CREATE.SYMBOL \INITATOMPAGE \MOVEBYTES \STKMIN)
                   (FNS COPYATOM INITATOMS))

(ADDTOVAR EXPANDMACROFNS SMALLPOSP COMPUTE.ATOM.HASH ATOM.HASH.REPROBE \DEFCELL \VALCELL \PNAMECELL \PROPCELL
                         \INDEXATOMPNAME)

(ADDTOVAR MKI.SUBFNS (\PARSE.NUMBER . NILL)
                     (\MKATOM.FULL . NILL)
                     (\ATOMDEFINDEX . I.ATOMNUMBER)
                     (\ATOMVALINDEX . I.ATOMNUMBER)
                     (\ATOMPROPINDEX . I.ATOMNUMBER)
                     (\ATOMPNAMEINDEX . I.ATOMNUMBER)
                     (\ATOMCELL . I.\ATOMCELL)
                     (\GETBASEFIXP . I.GETBASEFIXP)
                     (\PUTBASEFIXP . I.PUTBASEFIXP)
                     (SETQ.NOREF . SETQ)
                     (SETTOPVAL . I.FSETVAL))

(ADDTOVAR RD.SUBFNS (\PARSE.NUMBER . NILL)
                    (\ATOMDEFINDEX . VATOMNUMBER)
                    (\ATOMPROPINDEX . VATOMNUMBER)
                    (\ATOMVALINDEX . VATOMNUMBER)
                    (SETQ.NOREF . SETQ)
                    (\INDEXATOMPNAME . VATOM)
                    (\INDEXATOMVAL . VATOM)
                    (\INDEXATOMDEF . VATOM)
                    (\ATOMNUMBER . VATOMNUMBER)
                    (\CREATE.SYMBOL . VNOSUCHATOM))

(ADDTOVAR RDCOMS (FNS UNCOPYATOM MAKE.LOCAL.ATOM SYMBOL.VALUE SYMBOL.PNAME SYMBOL.PACKAGE OLD.FIND.SYMBOL
                      LOOKUP-SYMBOL FIND.PACKAGE FIND.SYMBOL PACKAGE.NAME \MKATOM GETTOPVAL GETPROPLIST
                      SETTOPVAL GETDEFN \ATOMCELL)
                 (FNS LISTP)
                 (VARS (COPYATOMSTR)))

(ADDTOVAR RD.SUBFNS (\RPLPTR . VPUTBASEPTR))
```

```
(ADDTOVAR RDVALS (\AtomFrLst))
)

(PUTPROPS LLBASIC FILETYPE CL:COMPILE-FILE)

(PUTPROPS LLBASIC COPYRIGHT ("Venue & Xerox Corporation" 1981 1982 1983 1984 1985 1986 1987 1988 1990 1991 1992
                             1993))
```

FUNCTION INDEX

MACRO INDEX

VARIABLE INDEX

RECORD INDEX

CONSTANT INDEX

TEMPLATE INDEX

OPTIMIZER INDEX

PROPERTY INDEX