File created: 22-Sep-92 19:22:00 **{DSK}<usr>local>lde>lispcore>sources>MACHINEINDEPENDENT.;2**

changes to: (FNS NAMEFIELD COMSNAME NAMEFIELD-STRING)
(VARS MACHINEINDEPENDENTCOMS)

previous date: 27-Feb-91 18:30:38 {DSK}<usr>local>lde>lispcore>sources>MACHINEINDEPENDENT.;1

Read Table: XCL

Package: INTERLISP

Format: XCCS


```
(RPAQQ MACHINEINDEPENDENTCOMS
       ((COMS                                                    ; "File loader"
              (FNS LOAD? FILESLOAD DOFILESLOAD FINDFILE-WITH-EXTENSIONS)
              (INITVARS (*COMPILED-EXTENSIONS* (LIST FASL.EXT COMPILE.EXT))))
        (COMS                                                    ; random machine-independent utilities
              (FNS DMPHASH HASHOVERFLOW)
              (DECLARE\: EVAL@COMPILE DONTCOPY (MACROS HASHOVERFLOW.ARRAYTEST HASHOVERFLOW.UPDATEARRAY))
              (FNS BKBUFS CHANGENAME CHNGNM CLBUFS COMSNAME DEFINE FNS.PUTDEF EQMEMB EQUALN FNCHECK FNTYP1
                   LCSKIP MAPRINT MKLIST NAMEFIELD NAMEFIELD-STRING NLIST PRINTBELLS PROMPTCHAR RAISEP READFILE
                   READLINE REMPROPLIST RESETBUFS TAB UNSAVED1 WRITEFILE CLOSE-AND-MAYBE-DELETE UNSAFE.TO.MODIFY
                   )
              (VARS UNSAFE.TO.MODIFY.FNS)
              (COMS                                              ; FILEDATE, for finding out the creation date of source files, from
                                                                 ; the compiled files.
                    ;; FASL isn't loaded when MACHINEINDEPENDENT is, so we have to fake the FASL checker for now.  It's defined in
                    ;; FASLOAD.
                    (FNS FILEDATE)
                    (P (MOVD? 'NILL 'FASL-FILEDATE)))
              (P (MOVD? 'CL:FMAKUNBOUND 'UNDOABLY-FMAKUNBOUND))
                                                                 ; used in FNS.PUTDEF before CMLUNDO loaded
              )
        (COMS                                                    ; Functions for retrieving and remembering FILEMAPs and file
                                                                 ; reader environments
              (FNS FILEMAP \\PARSE-FILE-HEADER GET-ENVIRONMENT-AND-FILEMAP LOOKUP-ENVIRONMENT-AND-FILEMAP
                   GET-FILEMAP-FROM-FILECREATED \\FILEMAP-HASHOVERFLOW FLUSHFILEMAPS LISPSOURCEFILEP GETFILEMAP
                   PUTFILEMAP UPDATEFILEMAP PRINT-READER-ENVIRONMENT)
              (INITVARS (*FILEMAP-LIMIT* 20)
                     (*FILEMAP-VERSIONS* 2)
                     (*FILEMAP-HASH* (HASHARRAY *FILEMAP-LIMIT* (FUNCTION \\FILEMAP-HASHOVERFLOW)
                                                (FUNCTION STRING-EQUAL-HASHBITS)
                                                (FUNCTION STRING.EQUAL))))
              (DECLARE\: EVAL@COMPILE DONTCOPY (RECORDS FILEMAPHASH)
                     (GLOBALVARS *FILEMAP-LIMIT* *FILEMAP-VERSIONS* *FILEMAP-HASH*)))
        (COMS (* * LVLPRINT)
              (FNS LVLPRINT LVLPRIN1 LVLPRIN2 LVLPRIN LVLPRIN0))
        (COMS                                                    ; used by PRINTOUT
              (FNS FLUSHRIGHT PRINTPARA PRINTPARA1))
        (COMS                                                    ; SUBLIS and friends
              (FNS SUBLIS SUBPAIR DSUBLIS))
        (COMS (* * CONSTANTS)
              (FNS CONSTANTOK)
              (P (MOVD? 'EVQ 'CONSTANT)
                 (MOVD? 'EVQ 'DEFERREDCONSTANT)
                 (MOVD? 'EVQ 'LOADTIMECONSTANT)))
        (COMS (* * SCRATCHLIST)
              (PROP MACRO SCRATCHLIST ADDTOSCRATCHLIST)
              (PROP INFO SCRATCHLIST))
        (GLOBALVARS SYSFILES LOADOPTIONS LISPXCOMS CLISPTRANFLG COMMENTFLG HISTSTR4 LISPXREADFN REREADFLG
               HISTSTR0 CTRLUFLG NOLINKMESS PROMPTCHARFORMS PROMPT#FLG FILERDTBL SPELLINGS2 USERWORDS BELLS
               CLISPARRAY)
        (FNS NLAMBDA.ARGS)
        (DECLARE\: DONTEVAL@LOAD DOCOPY                          ; initialization of variables used in many places
               (ADDVARS (CLISPARRAY)
                      (CLISPFLG)
                      (CTRLUFLG)
                      (EDITCALLS)
                      (EDITHISTORY)
                      (EDITUNDOSAVES)
                      (EDITUNDOSTATS)
                      (GLOBALVARS)
                      (LCASEFLG)
                      (LISPXBUFS)
                      (LISPXCOMS)
                      (LISPXFNS)
                      (LISPXHIST)
                      (LISPXHISTORY)
                      (LISPXPRINTFLG)
                      (NOCLEARSTKLST)
```

```
                    (NOFIXFNSLST)
                    (NOFIXVARSLST)
                    (P.A.STATS)
                    (PROMPTCHARFORMS)
                    (READBUF)
                    (READBUFSOURCE)
                    (REREADFLG)
                    (RESETSTATE)
                    (SPELLSTATS1))
              (VARS (CHCONLST '(NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL
                                    ))
                    (CHCONLST1 '(NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL
                                    NIL))
                    (CHCONLST2 '(NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL
                                    NIL))
                    (CLEARSTKLST T)
                    (CLISPTRANFLG 'CLISP\ )
                    (HISTSTR0 "<c.r.>")
                    (HISTSTR2 "repeat")
                    (HISTSTR3 "from event:")
                    (HISTSTR4 "ignore")
                    (LISPXREADFN 'READ)
                    (USEMAPFLG T))
              (P (MAPC '((APPLY BLKAPPLY)
                         (SETTOPVAL SETATOMVAL)
                         (GETTOPVAL GETATOMVAL)
                         (APPLY* BLKAPPLY*)
                         (RPLACA FRPLACA)
                         (RPLACD FRPLACD)
                         (STKNTH FSTKNTH)
                         (STKNAME FSTKNAME)
                         (CHARACTER FCHARACTER)
                         (STKARG FSTKARG)
                         (CHCON DCHCON)
                         (UNPACK DUNPACK)
                         (ADDPROP /ADDPROP)
                         (ATTACH /ATTACH)
                         (DREMOVE /DREMOVE)
                         (DSUBST /DSUBST)
                         (NCONC /NCONC)
                         (NCONC1 /NCONC1)
                         (PUT /PUT)
                         (PUTPROP /PUTPROP)
                         (PUTD /PUTD)
                         (REMPROP /REMPROP)
                         (RPLACA /RPLACA)
                         (RPLACD /RPLACD)
                         (SET /SET)
                         (SETATOMVAL /SETATOMVAL)
                         (SETTOPVAL /SETTOPVAL)
                         (SETPROPLIST /SETPROPLIST)
                         (SET SAVESET)
                         (PRINT LISPXPRINT)
                         (PRIN1 LISPXPRIN1)
                         (PRIN2 LISPXPRIN2)
                         (SPACES LISPXSPACES)
                         (TAB LISPXTAB)
                         (TERPRI LISPXTERPRI)
                         (PRINT SHOWPRINT)
                         (PRIN2 SHOWPRIN2)
                         (PUTHASH /PUTHASH)
                         '*
                         (FNCLOSER /FNCLOSER)
                         (FNCLOSERA /FNCLOSERA)
                         (FNCLOSERD /FNCLOSERD)
                         (EVQ DELFILE)
                         (NILL SMASHFILECOMS)
                         (PUTASSOC /PUTASSOC)
                         (LISTPUT1 PUTL)
                         (NILL I.S.OPR)
                         (NILL RESETUNDO)
                         (NILL LISPXWATCH)
                         'ADDSTATS
                         (NILL FREEVARS)
                         'USEDFREE
                         (COPYBYTES COPYCHARS))
                        (FUNCTION (LAMBDA (X)
                                          (MOVD? (CAR X)
                                                 (CADR X)))))
                 (MAPC '((TIME PRIN1 LISPXPRIN1)
                         (TIME SPACES LISPXSPACES)
                         (TIME PRINT LISPXPRINT)
                         (DEFC PRINT LISPXPRINT)
                         (DEFC PUTD /PUTD)
                         (DEFC PUTPROP /PUTPROP)
                         (DOLINK FNCLOSERD /FNCLOSERD)
                         (DOLINK FNCLOSERA /FNCLOSERA)
```

```
                        (DEFLIST PUTPROP /PUTPROP)
                        (SAVEDEF1 PUTPROP /PUTPROP)
                        (MKSWAPBLOCK PUTD /PUTD))
                      (FUNCTION (LAMBDA (X)
                                        (AND (CCODEP (CAR X))
                                             (APPLY 'CHANGENAME X)))))
                (MAPC '((EVALQT (LAMBDA NIL (PROG (TEM)
                                                  (RESETRESTORE NIL 'RESET)
                                                  LP
                                                  (PROMPTCHAR '_ T)
                                                  (LISPX (LISPXREAD T T))
                                                  (GO LP))))
                        (LISPX (LAMBDA (LISPXX)
                                       (PRINT (AND LISPXX (PROG (LISPXLINE LISPXHIST TEM)
                                                                (RETURN (COND ((AND (NLISTP LISPXX)
                                                                                    (SETQ LISPXLINE
                                                                                          (READLINE T NIL T)))
                                                                               (APPLY LISPXX (CAR LISPXLINE)))
                                                                              (T (EVAL LISPXX))))))
                                              T T)))
                        (LISPXREAD (LAMBDA (FILE RDTBL)
                                           (COND (READBUF (PROG1 (CAR READBUF)
                                                                 (SETQ READBUF (CDR READBUF))))
                                                 (T (READ FILE RDTBL)))))
                        (LISPXREADP (LAMBDA (FLG)
                                            (COND ((AND READBUF (SETQ READBUF (LISPXREADBUF READBUF)))
                                                   T)
                                                  (T (READP T FLG)))))
                        (LISPXUNREAD (LAMBDA (LST)
                                             (SETQ READBUF (APPEND LST (CONS HISTSTR0 READBUF)))))
                        (LISPXREADBUF (LAMBDA (RDBUF)
                                              (PROG NIL LP (COND ((NLISTP RDBUF)
                                                                  (RETURN NIL))
                                                                 ((EQ (CAR RDBUF)
                                                                      HISTSTR0)
                                                                  (SETQ RDBUF (CDR RDBUF))
                                                                  (GO LP))
                                                                 (T (RETURN RDBUF))))))
                        (LISPX/ (LAMBDA (X)
                                        X))
                        (LOWERCASE (LAMBDA (FLG)
                                           (PROG1 LCASEFLG
                                                  (RAISE (NULL FLG))
                                                  (RPAQ LCASEFLG FLG))))
                        (FILEPOS (LAMBDA (STR FILE)
                                         (PROG NIL LP (COND ((EQ (PEEKC FILE)
                                                                 (NTHCHAR STR 1))
                                                             (RETURN T)))
                                               (READC FILE)
                                               (GO LP))))
                        (FILEPKGCOM (NLAMBDA NIL NIL)))
                      (FUNCTION (LAMBDA (L)
                                        (OR (GETD (CAR L))
                                            (PUTD (CAR L)
                                                  (CADR L))))))))
        (DECLARE\: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA RESETBUFS DMPHASH
                                                                                FILESLOAD)
                                                                               (NLAML FILEMAP)
                                                                               (LAMA READFILE NLIST)))
        (LOCALVARS . T)))

;; "File loader"

(DEFINEQ

(LOAD?
  (LAMBDA (FILE LDFLG PRINTFLG)                                    (* |lmm| " 2-Sep-85 13:15")
    (|bind| FULL |until| (SETQ FULL (FINDFILE FILE)) |do| (SETQ FILE (LISPERROR "FILE NOT FOUND" FILE T))
        |finally|
        (RETURN (|if| (FMEMB FULL LOADEDFILELST)
                   |then| FULL
                |else| (LET* ((ROOT (ROOTFILENAME FULL))
                              (DATES (GETPROP ROOT 'FILEDATES))
                              (FILEPROP (GETPROP ROOT 'FILE)))
                             (|if| (AND DATES (|if| (EQ (FILENAMEFIELD FULL 'EXTENSION)
                                                        COMPILE.EXT)
                                                    |then| (AND (OR (NULL FILEPROP)
                                                                    (FMEMB (CDAR FILEPROP)
                                                                           '(|Compiled| COMPILED)))
                                                                (EQUAL (CAAR DATES)
                                                                       (FILEDATE FULL T)))
                                                    |else| (AND FILEPROP (EQ (CDAR FILEPROP)
                                                                             T)
                                                                (OR (EQ (CDAR DATES)
                                                                        FULL)
                                                                    (EQUAL (CAAR DATES)
                                                                           (FILEDATE FULL))))))
```

```
                                |then| FULL
                               |else| (LOAD FULL LDFLG PRINTFLG)))))))))
```

(FILESLOAD

```
(NLAMBDA FILES                                                    (* |lmm| "10-Dec-84 17:23")
```

;; Calls to this are written on files by the FILES command. This function does the load-time evaluation of the command.

```
(DOFILESLOAD (NLAMBDA.ARGS FILES))))
```

(DOFILESLOAD

```
(LAMBDA (FILES)
  (DECLARE (USEDFREE LDFLG))                                      ; Edited 4-May-88 14:23 by bvm
                                                                  ; does the work of FILESLOAD
  (|for| FILE |inside| FILES |bind| DIRS LOADOPTIONSFLG FORCEDEXT? NOERRORFLG WORD FULL (FN _ 'LOAD?)
                          (EXT _ :COMPILED)
    |first| (COND
             ((BOUNDP 'LDFLG)
              ;; Under a load;  give priority to directory of currently loading file.
              (LET ((INPUTNAME (FULLNAME *STANDARD-INPUT*)))
                   (|if| (AND (NEQ INPUTNAME *STANDARD-INPUT*)
                              (NEQ INPUTNAME T))
                        |then|                                    ; If reading from terminal or nameless stream, don't do this.
                        (SETQ DIRS (CONS (PACKFILENAME.STRING 'VERSION NIL 'NAME NIL 'EXTENSION NIL
                                                              'BODY INPUTNAME)
                                         (CONS T DIRECTORIES)))
                        (SETQ LOADOPTIONSFLG LDFLG)))))
    |join| (COND
            ((OR (LITATOM FILE)
                 (STRINGP FILE))                                  ; A file to do something with
             (PROG NIL
                   (COND
                      ((AND (EQ FN 'LOAD?)
                            (GETPROP (ROOTFILENAME FILE)
                                     'FILEDATES))                 ; Already loaded
                       (RETURN)))
               LP  (COND
                      ((SETQ FULL (SELECTQ EXT
                                           (NIL                   ; No extension to guide us
                                                (FINDFILE FILE T DIRS))
                                           (:COMPILED             ; Look for some sort of compiled file, or failing that a source
                                                      (OR (FINDFILE-WITH-EXTENSIONS FILE DIRS
                                                                                    *COMPILED-EXTENSIONS*)
                                                          (AND (NOT FORCEDEXT?)
                                                               (FINDFILE FILE T DIRS))))
                                           (PROGN                 ; Look for explicitly supplied extension
                                                  (FINDFILE (PACKFILENAME.STRING 'BODY FILE 'EXTENSION EXT)
                                                            T DIRS)))))
                             (NOERRORFLG (RETURN))
                             ((AND (SETQ FILE (CL:CERROR "Forget about loading ~A" "File ~A not found~@[ on~{
                                                                                    ~A~}~]" FILE DIRS))
                                   (OR (LITATOM FILE)
                                       (STRINGP FILE)))          ; User RETURNed a new file name
                              (GO LP))
                             (T                                   ; if proceed from ERROR, blow off loading this file
                                (RETURN)))
                      (RETURN (LIST (SELECTQ FN
                                             (CHECKIMPORTS        ; LOADOPTIONSFLG has a different meaning for imports
                                                           (CHECKIMPORTS FULL T)
                                                           FULL)
                                             (LOAD?               ; already weeded out the ones with filedates
                                                    (LOAD FULL LOADOPTIONSFLG))
                                             (CL:FUNCALL FN FULL LOADOPTIONSFLG))))))
            (T (|while| (LISTP FILE)
                  |do| (SELECTQ (CAR FILE)
                                (LOADCOMP (SETQQ FN LOADCOMP?)
                                          (SETQ LOADOPTIONSFLG NIL)
                                          (SETQ EXT NIL))
                                (LOADFROM (SETQQ FN LOADFROM)
                                          (SETQ EXT NIL))
                                (FROM (|pop| FILE)
                                      (SETQ DIRS (MKLIST (COND
                                                           ((OR (EQ (SETQ WORD (CAR FILE))
                                                                    'VALUEOF)
                                                                (COND
                                                                   ((AND (EQ WORD 'VALUE)
                                                                         (EQ (CADR FILE)
                                                                             'OF))
                                                                    (|pop| FILE)
                                                                    T)))
                                                            (|pop| FILE)
                                                            (EVAL (CAR FILE)))
                                                           ((AND (SELCHARQ (CHCON1 WORD)
                                                                           (({ <)
                                                                            NIL)
```

```
                                                                    T)
                                                                (BOUNDP (SETQ WORD (PACK* WORD 'DIRECTORIES)))
                                                                (SETQ WORD (EVALV WORD)))
                                                                 ; KLUDGE: Turns, e.g., (FROM LISPUSERS) into (FROM
                                                                 ; VALUEOF LISPUSERSDIRECTORIES)
                                                            WORD)
                                                           (T (CAR FILE))))))
                                    (COMPILED (SETQ FORCEDEXT? T)
                                              (SETQ EXT :COMPILED))
                                    (LOAD (SETQQ FN LOAD?))
                                    ((EXTENSION EXT)
                                         (SETQ FILE (LISTP (CDR FILE)))
                                         (SETQ EXT (CAR FILE)))
                                    ((SOURCE SYMBOLIC)
                                         (SETQ EXT NIL))
                                    (IMPORT (SETQQ FN CHECKIMPORTS)
                                            (SETQ EXT NIL))
                                    (NOERROR (SETQ NOERRORFLG T))
                                    (COND
                                       ((FMEMB (CAR FILE)
                                               LOADOPTIONS)
                                        (SETQ LOADOPTIONSFLG (CAR FILE)))
                                       (T                                ; invalid option in FILESLOAD
                                          NIL)))
                              (|pop| FILE))
                        NIL)))))
```

(FINDFILE-WITH-EXTENSIONS

```
  (LAMBDA (FILE DIRLST EXTENSIONS)                                  ; Edited  8-Dec-86 17:57 by bvm
```

;;; Search for FILE on the directories contained in DIRLST, where NIL and T refer to the login and connected dirs, respectively. On each directory, prefer ;;; files having extension found in EXTENSIONS in the indicated order. If FILE already has an extension, EXTENSIONS is ignored; if FILE already has a ;;; host/dir, DIRLST is ignored.

```
    (|if| FILE
        |then|
        (LET
         ((FIELDS (UNPACKFILENAME.STRING FILE))
          DIR&FIELDS HASDIRECTORY HASEXTENSION VAL)
         (|for| TAIL |on| FIELDS |by| (CDDR TAIL) |do| (SELECTQ (CAR TAIL)
                                                              (EXTENSION (SETQ HASEXTENSION T))
                                                              ((HOST DEVICE DIRECTORY)
                                                                  (SETQ HASDIRECTORY T))
                                                              NIL))
         (|if| HASDIRECTORY
             |then|                                                 ; Don't search dirs, just look where it says
                   (|if| HASEXTENSION
                       |then| (INFILEP FILE)
                     |else| (|for| EXT |in| EXTENSIONS |when| (SETQ VAL (INFILEP (PACKFILENAME.STRING
                                                                                    `(EXTENSION ,EXT ,@FIELDS))))
                                |do| (RETURN VAL)))
           |else| (|for| DIR |inside| (|if| (NULL DIRLST)
                                          |then|                    ; If DIRLST is defaulted, always look first on connected dir.
                                                (|if| DIRECTORIES
                                                    |then| (CONS T (REMOVE T DIRECTORIES))
                                                  |else| T)
                                        |else|                      ; use explicit DIRLST, ignoring connected dir unless it's on
                                                                    ; DIRECTORIES
                                               DIRLST)
                     |when| (PROGN (SETQ DIR&FIELDS (SELECTQ DIR
                                                            (NIL       ; Login dir
                                                                 `(DIRECTORY ,(DIRECTORYNAME NIL)
                                                                             ,@FIELDS))
                                                            (T         ; Connected dir
                                                               FIELDS)
                                                            `(DIRECTORY ,DIR ,@FIELDS)))
                                   (SETQ VAL (|if| HASEXTENSION
                                                   |then| (INFILEP (PACKFILENAME.STRING DIR&FIELDS))
                                                 |else| (|for| EXT |in| EXTENSIONS
                                                             |when| (SETQ VAL (INFILEP (PACKFILENAME.STRING
                                                                                        `(EXTENSION ,EXT ,@DIR&FIELDS))))
                                                             |do| (RETURN VAL)))))
                     |do| (RETURN VAL)))))))

)

(RPAQ? *COMPILED-EXTENSIONS* (LIST FASL.EXT COMPILE.EXT))

;; random machine-independent utilities

(DEFINEQ

(DMPHASH
  (NLAMBDA L                                                        (* |rmk:| " 6-Apr-84 14:30")
    (MAPC L (FUNCTION (LAMBDA (ARRAYNAME)
                      (DECLARE (SPECVARS ARRAYNAME))
```

```
                        (ERSETQ (PROG ((A (EVALV ARRAYNAME 'DMPHASH))
                                       AP)
                                      (PRINT (LIST 'RPAQ ARRAYNAME (COND
                                                                       ((LISTP A)
                                                                        (SETQ AP (CAR A))
                                                                        (LIST 'CONS
                                                                              (LIST 'HARRAY (HARRAYSIZE AP)
                                                                                    (KWOTE (HARRAYPROP
                                                                                               AP
                                                                                               'OVERFLOW)))
                                                                              (KWOTE (CDR A))))
                                                                       (T (LIST 'HASHARRAY (HARRAYSIZE A)
                                                                                (KWOTE (HARRAYPROP AP
                                                                                               'OVERFLOW)))))))
                                      (MAPHASH (OR AP A)
                                             (FUNCTION (LAMBDA (VAL ITEM)
                                                          (PRINT (LIST 'PUTHASH (KWOTE ITEM)
                                                                       (KWOTE VAL)
                                                                       ARRAYNAME)))))))))
```

(HASHOVERFLOW

```
(LAMBDA (HARRAY)                                          ; Edited 26-Feb-91 13:16 by jds
```

;; Should be called from PUTHASH on hash overflow, but for implementations where PUTHASH calls ERRORX directly, may be called from ;; ERRORX2 when the offender is a listp. HARRAY is guaranteed to be either HARRAYP or (LIST HARRAYP)

```
    (PROG ((OLDARRAY (HASHOVERFLOW.ARRAYTEST HARRAY))
           NEWARRAY NEWSIZE OLDNUMKEYS OVACTION NEWOVFLW)
          (COND
             ((LISTP HARRAY)
              (SETQ OVACTION (CDR HARRAY))
```

;; Get OVERFLOW method from original HARRAY since it would erroneously be ERROR if we got the method from the coerced ;; OLDARRAY

```
              (SETQ NEWOVFLW 'ERROR))
             (T (SETQ OVACTION (SETQ NEWOVFLW (HARRAYPROP OLDARRAY 'OVERFLOW)))))
          (SETQ OLDNUMKEYS (HARRAYPROP OLDARRAY 'NUMKEYS))
```

;; Compute the new array size:

```
          (SETQ NEWSIZE (SELECTQ OVACTION
                                 (NIL ;; SIZE*1.5 --- favor to bbn, since pdp-11 doesnt have floatng point, and LRSH on other systems might be
                                      ;; faster than IQUOTIENT
                                      ;; [32749 IS THE BIGGEST PRIME < 32765, THE LIMIT ON ARRAY SIZES]
                                      (IMAX (+ OLDNUMKEYS 3)
                                            (IMIN 32749 (+ OLDNUMKEYS (LRSH (CL:1+ OLDNUMKEYS)
                                                                            1)))))
                                 (ERROR (|do| (ERRORX (LIST 26 HARRAY))))
                                 (|if| (FLOATP OVACTION)
                                     |then| (IMAX (+ OLDNUMKEYS 3)
                                                  (IMIN 32760 (FIXR (FTIMES OLDNUMKEYS OVACTION))))
                                   |elseif| (FIXP OVACTION)
                                     |then| (IMAX (+ OLDNUMKEYS 3)
                                                  (IMIN 32749 (+ OLDNUMKEYS OVACTION)))
                                   |elseif| (AND (FNTYP OVACTION)
                                                 (NUMBERP (SETQ OVACTION (APPLY* OVACTION HARRAY))))
                                     |then| (|if| (FLOATP OVACTION)
                                                  |then|                    ; recompute NUMKEYS since OVACTION might have removed
                                                                            ; keys
                                                       (IMAX (+ (SETQ OLDNUMKEYS (HARRAYPROP OLDARRAY 'NUMKEYS))
                                                                3)
                                                             (IMIN 32749 (FIXR (FTIMES OLDNUMKEYS OVACTION))))
                                                  |else| OVACTION)
                                   |else|                                   ; Default: multiply by 1.5
                                         (SETQ OLDNUMKEYS (HARRAYPROP OLDARRAY 'NUMKEYS))
                                         (IMAX (+ OLDNUMKEYS 3)
                                               (IMIN 32749 (+ OLDNUMKEYS (LRSH (CL:1+ OLDNUMKEYS)
                                                                               1)))))))
          (SETQ NEWARRAY (REHASH OLDARRAY (HASHARRAY NEWSIZE NEWOVFLW (HARRAYPROP OLDARRAY 'HASHBITSFN)
                                                     (HARRAYPROP OLDARRAY 'EQUIVFN))))
          (HASHOVERFLOW.UPDATEARRAY HARRAY NEWARRAY OLDARRAY)
          (RETURN HARRAY))))

)

(DECLARE\: EVAL@COMPILE DONTCOPY

(DECLARE\: EVAL@COMPILE

(PROGN (PUTPROPS HASHOVERFLOW.ARRAYTEST MACRO ((HARRAY)
                                               (CAR (OR (LISTP HARRAY)
                                                        (ERRORX (LIST 27 HARRAY))))))
       (PUTPROPS HASHOVERFLOW.ARRAYTEST DMACRO ((HARRAY)
                                                (\\DTEST HARRAY 'HARRAYP))))

(PROGN (PUTPROPS HASHOVERFLOW.UPDATEARRAY MACRO ((HARRAY NEWARRAY OLDARRAY)
                                                 (FRPLACA HARRAY NEWARRAY)))
```

```
       (PUTPROPS HASHOVERFLOW.UPDATEARRAY DMACRO ((HARRAY NEWARRAY OLDARRAY)
                                                  (\\COPYHARRAYP NEWARRAY OLDARRAY))))
)
)

(DEFINEQ

(BKBUFS
  (LAMBDA (BUFS ID)                                          (* DD\: " 6-Oct-81 15:34")
    (PROG (L S)
          (COND
             ((NLISTP BUFS)
              (RETURN))
             (T (SETQ L (CAR BUFS))
                (SETQ S (CDR BUFS))))
          (COND
             ((READP T)
              ;; User types ahead before command causing buffer to be restored was executed.  In this case, his type-ahead would come BEFORE
              ;; the restored buffer, when it should be after it, because the command causing the buffer to be restored had to have been given
              ;; before the type-ahead.
              (PRINTBELLS)
              (DOBE)
              (CLEARBUF T T)
              (BKSYSBUF S)
              (BKSYSBUF (SYSBUF T))
              (SYSBUF))
             (S (BKSYSBUF S)))
          (COND
             (L (AND ID (PRIN1 ID T))
                ;; ID will be suppressed by LISPX to prevent it being typed in middle of input.  Note that anything put back in SYSBUF will be
                ;; printed (echoed) as it is read.
                (PRIN1 L T)
                (BKLINBUF L)))
          (RETURN))))

(CHANGENAME
  (LAMBDA (FN FROM TO)                                       (* |wt:| "18-SEP-78 21:29")
    (COND
       ((CHANGENAME1 (GETD FN)
              FROM TO FN)
        (AND FILEPKGFLG (EXPRP FN)
             (MARKASCHANGED FN 'FNS))
        FN))))

(CHNGNM
  (LAMBDA (FN OLD FLG)
    (PROG (NEW DEF X Y Z)
          (SETQ FN (FNCHECK FN NIL T))                       ; No error, becuase maybe OLD isnt defined yet, e.g. BREAK
                                                             ; ((FOO IN FUM)) where FOO not defined.
          (SETQ OLD (OR (FNCHECK OLD T T)
                        OLD))
          (SETQ DEF (GETD (OR (GETP FN 'ADVISED)
                              (GETP FN 'BROKEN)
                              FN)))
          (SETQ NEW (PACK (LIST OLD '-IN- FN)))
          (COND
             (FLG (AND (NULL (STKPOS NEW))
                       (/PUTD NEW))
                  (COND
                     ((SETQ Z (/DREMOVE OLD (GETP FN 'NAMESCHANGED)))
                      (/PUT FN 'NAMESCHANGED Z))
                     (T (/REMPROP FN 'NAMESCHANGED)))
                  (/REMPROP NEW 'ALIAS)
                  (SETQ Y OLD)
                  (SETQ X NEW))
             (T (SETQ Y NEW)
                (SETQ X OLD)
                (COND
                   ((AND (MEMB OLD (GETP FN 'NAMESCHANGED))
                         (GETD NEW)
                         (GETP NEW 'ALIAS))
                    (RETURN NEW)))))
          (COND
             ((NULL DEF)
              (RETURN (CONS DEF '(|not| |defined|))))
             ((NULL (RESETVARS ((NOLINKMESS T))
                               (RETURN (CHANGENAME1 DEF X Y FN))))
              (RETURN (CONS X (APPEND '(|not| |found| |in|)
                                      (LIST FN))))))
          (COND
             ((NULL FLG)
              (COND
                 ((NULL (SETQ DEF (GETD OLD)))
```

```
                (SETQ DEF (LIST 'NLAMBDA (GENSYM)))
                (PRINT (CONS OLD '(|was| |undefined|))
                       T T)))
            (/PUTD NEW (SAVED OLD NIL DEF OLD))
            (/ADDPROP FN 'NAMESCHANGED OLD)
            (/PUT NEW 'ALIAS (CONS FN OLD))))
        (RETURN Y))))
```

(CLBUFS

```
(LAMBDA (NOCLEARFLG NOTYPEFLG BUF)                          ; wt: 10-MAR-77 21 5
```

;; NOCLEARFLG=T means CLEARBUF has already been done, and anything in the buffer now is type-ahead, e.g. calls from EVALQT, and call ;; from BREAK on control-h INTERRUPT.

;; NOTYPEFLG=T means user should not be typing ahead. If READP is T, warn him to stop and wait. Occurs when CLBUFS is being done ;; BEFORE some action, e.g. DWIM interaction, loading SYSBUF for EXEC commands, etc. as opposed to AFTER some action, e.g. an error ;; occurred.

```
(PROG (LBUF SBUF)
      (COND
         (NOCLEARFLG (GO SKIP))
         ((AND NOTYPEFLG (READP T))
          (PRINTBELLS)
          (DOBE)))
      (CLEARBUF T T)
      (SETQ READBUF BUF)
  SKIP
      (SETQ CTRLUFLG NIL)                                   ; In case user control-e's or control-d's after typing control-u and
                                                            ; changing his mind.
      (SETQ LBUF (LINBUF T))
      (SETQ SBUF (SYSBUF T))
      (LINBUF)
      (SYSBUF)
      (COND
         ((STREQUAL LBUF '"
                         ")
          (SETQ LBUF NIL)))
      (RETURN (COND
                 ((OR SBUF LBUF)
                  (CONS LBUF SBUF)))))))
```

(COMSNAME

```
(LAMBDA (FILE SUFFIXFLG DIRFLG)
```

;; IF SUFFIXFLG is T, returns name and suffix field, otherwise just NAMEFIELD

```
(LET ((STR (NAMEFIELD-STRING FILE SUFFIXFLG DIRFLG)))
```

;; COMS names must be smashed to upper-case; if you want the unsmashed filename, use NAMEFIELD

```
     (|if| (NOT (U-CASEP STR))
         |then| (SETQ STR (U-CASE STR)))
     (MKATOM STR))))
```

(DEFINE

```
(LAMBDA (X TYPE-IN)                                         (* |mpl| "15-Jul-85 11:22")
  (MAPCAR X (FUNCTION (LAMBDA (X)
                        (COND
                           ((NLISTP X)
                            (ERROR '"incorrect defining form" X)))
                        (FNS.PUTDEF (CAR X)
                               'FNS
                               (COND
                                  ((NULL (CDDR X))
                                   (CADR X))
                                  (T (CONS 'LAMBDA (CDR X))))
                               (|if| TYPE-IN
                                   |then| 'DEFINED
                                 |else| 'LOAD)))))))
```

(FNS.PUTDEF

```
(LAMBDA (NAME TYPE DEFINITION REASON)                       ; Edited 20-Nov-87 14:24 by woz
  (PROG NIL
        (|if| (OR (AND DEFINITION (NLISTP DEFINITION))
                  (NOT (FMEMB (CAR DEFINITION)
                              LAMBDASPLST)))
            |then| (ERROR DEFINITION "Illegal function definition"))
        (SELECTQ DFNFLG
             ((NIL T)
                  (|if| (UNSAFE.TO.MODIFY NAME "redefine")
                      |then| (ERROR NAME " not redefined" T)))
             NIL)
        (|if| (EQ REASON 'DEFINED)
            |then|
```

;; woz: i think this test is wrong; what about CHANGED? SEdit special cases FNS in sedit::completion, and calls ;; FIXEDITDATE directly, but shouldn't have to.

```
                  (FIXEDITDATE DEFINITION))
```

```
          (IF (AND (HASDEF NAME 'FUNCTIONS)
                   (NEQ (CAR DEFINITION)
                        'NLAMBDA))
              THEN                                          ; For a while, we can't prevent the use of both a DEFMACRO
                                                            ; and NLAMBDA for the same name.
                  (DELDEF NAME 'FUNCTIONS))
          (COND
             ((OR (NULL DFNFLG)
                  (EQ DFNFLG T))
              (COND
                 ((GETD NAME)
                  (VIRGINFN NAME T)
                  ;; ((EQUAL DEFINITION (GETD NAME)) (RETURN NAME)) Used to be part of the following COND.  ripped out because editing
                  ;; out of the function cell wasn't completing fully.
                  (COND
                     ((NULL DFNFLG)
                      (PROGN                                ; if EXEC-FORMAT existed earlier, I'd use it
                             (LISPXPRIN1 "New fns definition for " T)
                             (LISPXPRIN2 NAME T)
                             (LISPXPRIN1 ".
                                    " T))
                      (SAVEDEF NAME)))))
                  (COND
                     (ADDSPELLFLG (ADDSPELL NAME)))
                  (UNDOABLY-SETF (CL:SYMBOL-FUNCTION NAME)
                        DEFINITION)
                  ;; Removed:  (REMPROP NAME 'EXPR)  because it wasn't saving the definition where UNSAVEDEF could find it.
                  )
                 (T                                         ; DFNFLG is PROP or ALLPROP.  However, treat anything else
                                                            ; the same as PROP.
                    (AND ADDSPELLFLG (ADDSPELL NAME 0))
                    (CL:UNLESS (EQ DEFINITION (GETD NAME))
                        ;; woz:  don't want to have an EXPR property if have the definition in the function cell, so be careful here.
                        (CL:WHEN (AND (OR (NULL REASON)
                                          (EQ REASON 'CHANGED))
                                      (EQ DEFINITION (GETPROP NAME 'EXPR)))
                            ;; editing a definition out of the saved EXPR property, and since DFNFLG is PROP, let the user know not installed
                            (LISPXPRIN1 "New fns definition for " T)
                            (LISPXPRIN2 NAME T)
                            (LISPXPRIN1 " (but not installed).
                                   " T))
                        (/PUTPROP NAME 'EXPR DEFINITION))))
          (COND
             (FILEPKGFLG (MARKASCHANGED NAME 'FNS REASON)))
          (RETURN NAME))))
```

(**EQMEMB**

```
(LAMBDA (X Y)                                               (* |lmm:| 17 APR 75 305)
  (OR (EQ X Y)
      (AND (LISTP Y)
           (FMEMB X Y)
          T))))
```

(**EQUALN**

```
(LAMBDA (X Y DEPTH)                                         (* |wt:| "12-JUN-80 10:57")
  ;; like EQUAL but stops, returning T, if depth of car recursion plus depth of cdr recursion ever exceeds DEPTH.
  (COND
     ((EQ X Y))
     ((NLISTP X)
      (COND
         ((NUMBERP X)
          (AND (NUMBERP Y)
               (EQP X Y)))
         ((STRINGP X)
          (STREQUAL X Y))
         ((STACKP X)
          (EQP X Y))))
     ((NLISTP Y)
      NIL)
     ((AND DEPTH (ILESSP DEPTH 1))
      '?)
     (T (SELECTQ (EQUALN (CAR X)
                         (CAR Y)
                         (AND DEPTH (SETQ DEPTH (SUB1 DEPTH))))
             (? '?)
             (T (EQUALN (CDR X)
                        (CDR Y)
                        DEPTH))
             NIL))))
```

(FNCHECK

```
(LAMBDA (FN NOERRORFLG SPELLFLG PROPFLG TAIL)                    (* |bvm:| "30-OCT-83 21:59")
  (PROG (X BLOCK BLOCK/FN)
    TOP (COND
           ((NOT (LITATOM FN))
            (GO ERROR))
           ((GETD FN))
           ((GETP FN 'EXPR)
            (AND (NULL PROPFLG)
                 (GO ERROR)))
           ((NULL DWIMFLG)
            (GO ERROR))
           ((AND (CAR (NLSETQ (SETQ X (OR (MISSPELLED? FN 70 USERWORDS SPELLFLG TAIL (FUNCTION GETD))
                                          (MISSPELLED? FN 70 SPELLINGS2 SPELLFLG TAIL)))))
                 (NEQ X FN))
            (SETQ FN X)
            (GO TOP))
           ((AND (EQ (SYSTEMTYPE)
                     'D)
                 (|for| FL |in| (WHEREIS FN) |thereis| (|for| FILE |inside| (OR (GETP FL 'FILEGROUP)
                                                                                 FL)
                                                                |thereis| (SETQ BLOCK (|find| B
                                                                                        |in| (FILECOMSLST FILE 'BLOCKS)
                                                                                        |suchthat| (AND (CAR X)
                                                                                                         (MEMB FN BLOCK))))))
                 (GETD (SETQ BLOCK/FN (PACK* '\\ (CAR BLOCK)
                                             '/ FN))))
            ;; In Interlisp-D, get actual name of internal block fn.  This is a little odd, since in a truly block-compiled system you couldn't get at the
            ;; subfns
            (SETQ FN BLOCK/FN))
           (T (GO ERROR)))
        (AND ADDSPELLFLG (ADDSPELL FN 0))
        (RETURN FN)
    ERROR
        (COND
           (NOERRORFLG (RETURN NIL)))
        (SETQ FN (ERROR FN '"not a function" (NULL (RELSTK (OR (STKPOS 'LOAD)
                                                               (STKPOS 'LOADFROM))))))
        (GO TOP))))
```

(FNTYP1

```
(LAMBDA (X)
  (AND CLISPARRAY (SETQ X (GETHASH X CLISPARRAY))
       (FNTYP X))))
```

(LCSKIP

```
(LAMBDA (FN FLG)                                                 (* |bvm:| "24-Oct-86 17:09")
  ;; Skip or copy FN, FLG T to copy
  (PROG (LEN LA)
        (|if| (EQ (PEEKCCODE)
                  (CHARCODE SPACE))
            |then| (COND
                      ((EQ (SETQ LA (READ))
                           'BINARY)
                       (RETURN (BINSKIP FN FLG NIL NIL LA)))
                      ((SETQ LEN (GETPROP LA 'CODEREADER))       ; Peter's hook for interfacing byte compiler.
                       (RETURN (APPLY* (CDR LEN)
                                       FN FLG NIL NIL LA)))))
        (ERROR "Bad or incompatible compiled function" FN))))
```

(MAPRINT

```
(LAMBDA (LST FILE LEFT RIGHT SEP PFN LSPXPRNTFLG)                (* |wt:| 15-SEP-77 15 43)
  (RESETVARS ((LISPXPRINTFLG LSPXPRNTFLG))
             (COND
                ((NULL PFN)
                 (SETQ PFN (FUNCTION LISPXPRIN1))))
             (COND
                ((NULL SEP)
                 (SETQ SEP '\ )))
             (COND
                (LEFT (LISPXPRIN1 LEFT FILE)))
             (COND
                ((NLISTP LST)
                 (GO EXIT)))
         LP  (APPLY* PFN (CAR LST)
                     FILE)
             (COND
                ((NULL (SETQ LST (CDR LST)))
                 (GO EXIT))
                ((NLISTP LST)
                 (LISPXPRIN1 '" . " FILE)
```

```
                (APPLY* PFN LST FILE)
                (GO EXIT)))
            (LISPXPRIN1 SEP FILE)
            (GO LP)
         EXIT
            (COND
               (RIGHT (LISPXPRIN1 RIGHT FILE))))))
```

(MKLIST

```
(LAMBDA (X)                                                    (* |lmm:| 21 AUG 75 428)
  (AND X (OR (LISTP X)
             (LIST X)))))
```

(NAMEFIELD

```
(LAMBDA (FILE SUFFIXFLG DIRFLG)                                ; Edited  5-Dec-90 22:32 by nm
```

;; IF SUFFIXFLG is T, returns name and suffix field, otherwise just NAMEFIELD

```
(MKATOM (NAMEFIELD-STRING FILE SUFFIXFLG DIRFLG))))
```

(NAMEFIELD-STRING

```
(LAMBDA (FILE SUFFIXFLG DIRFLG)
```

;; IF SUFFIXFLG is T, returns name and suffix field, otherwise just NAMEFIELD

```
  (LET ((STR (COND
                ((EQ DIRFLG 'ONLY)
                 (UNPACKFILENAME.STRING FILE 'DIRECTORY))
                ((EQ SUFFIXFLG 'ONLY)
                 (UNPACKFILENAME.STRING FILE 'EXTENSION))
                ((AND (NULL SUFFIXFLG)
                      (NULL DIRFLG))
                 (UNPACKFILENAME.STRING FILE 'NAME))
                (T
```

;; The general case. EXTENSION is fairly icky because UNPACKFILENAME.STRING behaves differently than
;; UNPACKFILENAME, in that it returns a null string instead of NIL for extensionless files

```
                   (PACKFILENAME.STRING 'DIRECTORY (AND DIRFLG (UNPACKFILENAME.STRING FILE 'DIRECTORY))
                          'NAME
                          (UNPACKFILENAME.STRING FILE 'NAME)
                          'EXTENSION
                          (AND SUFFIXFLG (SETQ SUFFIXFLG (UNPACKFILENAME.STRING FILE 'EXTENSION))
                               (> (NCHARS SUFFIXFLG)
                                  0)
                               SUFFIXFLG))))))
        STR)))
```

(NLIST

```
(LAMBDA N                                                      (* |bvm:| "14-Feb-85 23:48")
  (PROG (V (I N))
    LP  (COND
           ((EQ I 0)
            (RETURN V))
           ((OR V (ARG N I))
            (SETQ V (CONS (ARG N I)
                          V))))
        (SETQ I (SUB1 I))
        (GO LP))))
```

(PRINTBELLS

```
(LAMBDA NIL                                                    (* |wt:| 10-MAR-77 21 15)
  (PRIN3 BELLS T)))
```

(PROMPTCHAR

```
(LAMBDA (ID FLG HISTORY)
  (DECLARE (SPECVARS ID HISTORY PROMPTSTR))                    (* |lmm| " 9-Jun-85 20:53")
```

;; First checks READBUF, and strips off any leading pseudo-carriage retturns, and computes the new readbuf for repeated operations. If following
;; this, READBUF is not NIL, never prints ID. Otherwise prints ID if FLG is T, or if READP is NIL. FLG is T for calls from EVALQT and BREAK, NIL
;; from editor.

```
  (PROG (N MOD PROMPTSTR)
        (COND
           (FLG (AND READBUF (SETQ READBUF (LISPXREADBUF READBUF))
                     (RETURN NIL))                             ; redoing an event

                )
           ((LISPXREADP)                                       ; LISPXREADP returns T if there is anything on this line, but
                                                               ; returns NIL if just a c.r.
            (RETURN NIL)))
        (COND
           ((AND HISTORY PROMPT#FLG)
            (SETQ PROMPTSTR (COND
                               ((IGREATERP (SETQ N (ADD1 (CADR HISTORY)))
                                       (SETQ MOD (OR (CADDDR HISTORY)
```

```
                                                    100)))     ; This event is the roll-over event.
                                (IDIFFERENCE N MOD))
                               (T N)))))
          (COND
             (PROMPTCHARFORMS
                    ;; gives user a hook for operations to be performed each event, e.g.  monitoring functions, checking if typescript window is up
                    ;; etc.  also these forms can change what is printed by resetting promptstr and / or id
                    (MAPC PROMPTCHARFORMS (FUNCTION (LAMBDA (X)
                                                            (ERSETQ (EVAL X)))))))
          (AND PROMPTSTR (PRIN2 PROMPTSTR T))
          (AND ID (PRIN1 ID T)))))
```

(RAISEP

```
(LAMBDA (TTBL)                                              (* |wt:| 1-AUG-77 14 15)
```

;; True if lisp is in mode where it raises lower case inputs to uppercase.

```
(COND
   ((RAISE NIL TTBL)
    (RAISE T TTBL)
    T))))
```

(READFILE

```
(CL:LAMBDA (FILE &OPTIONAL RDTBL (ENDTOKEN 'STOP)
                 PACKAGE)
        (DECLARE (GLOBALVARS LOADPARAMETERS))                   ; Edited 20-Jan-87 16:22 by bvm:
        (WITH-READER-ENVIRONMENT *OLD-INTERLISP-READ-ENVIRONMENT*
            (RESETLST
                (RESETSAVE NIL (LIST 'CLOSEF? (SETQ FILE (OPENSTREAM FILE 'INPUT NIL NIL LOADPARAMETERS))))
                (IF (EQ (SKIPSEPRCODES FILE)
                        (CHARCODE ";"))
                    THEN (SETQ *READTABLE* CMLRDTBL)
                         (SETQ *PACKAGE* (CL:FIND-PACKAGE "USER")))
                (|if| RDTBL
                    |then| (SETQ *READTABLE* (\\DTEST RDTBL 'READTABLEP)))
                (|if| PACKAGE
                    |then| (SETQ *PACKAGE* (\\DTEST PACKAGE 'PACKAGE)))
                (LET ((EOFTOKEN "eof")
                      ENV TEM HELPCLOCK)
                     (DECLARE (SPECVARS HELPCLOCK))
                     (CL:VALUES (|until| (OR (EQ (SETQ TEM (CL:READ FILE NIL EOFTOKEN))
                                                 EOFTOKEN)
                                             (EQ TEM ENDTOKEN))
                                        |collect| (|if| (EQ (CAR TEM)
                                                            'DEFINE-FILE-INFO)
                                                        |then|          ; have to eval this to get the reader environment right for the rest
                                                                        ; of the file
                                                              (SET-READER-ENVIRONMENT (SETQ ENV (\\DO-DEFINE-FILE-INFO
                                                                                                 FILE
                                                                                                 (CDR TEM)))))
                                                        TEM)
                                ENV))))))
```

(READLINE

```
(LAMBDA (RDTBL LINE LISPXFLG)                               (* AJB " 1-Aug-85 14:50")
  (DECLARE (SPECVARS LINE LISPXFLG SPACEFLG))
  (PROG ((FL T)
         TEM SPACEFLG CHRCODE START)
    TOP (COND
           ((LISTP READBUF)
            (GO LP2))
           ((NULL (READP T))
            (CLEARBUF T)
```

;; This is in case there is a c.r. in the single character buffer. Note that if there were other atoms on the line terminated by a c.r., after ;; readline finished, the c.r. would be gone. Thus this check for consistency.

```
            (RETURN LINE)))
    LP  (SETQ SPACEFLG NIL)
    LP1 (COND
           ((SYNTAXP (SETQ CHRCODE (CHCON1 (SETQ TEM (PEEKC FL (OR RDTBL T)))))
                  'EOL)                                     ; C.R.
            (READC FL)
            (COND
               ((AND LINE SPACEFLG)
                (AND (EQ FL T)
                     (PRIN1 '|...| T))
                (GO LP))
               (T (GO OUT))))
           ((OR (SYNTAXP CHRCODE 'RIGHTPAREN RDTBL)
                (SYNTAXP CHRCODE 'RIGHTBRACKET RDTBL))
            (READ FL RDTBL)
            (AND LISPXFLG (NULL (CDR LINE))
                 (SETQ LINE (NCONC1 LINE NIL)))
```

;; The ']' is treated as NIL if it is the only thing on the line when READLINE is called with LISPXFLG=T. The reason for CDR is that

```
            ;; LISPX calls readline giving it the initial atom on the line.
             (GO OUT))
            ((AND (EQ CHRCODE (CHARCODE SPACE))
                  (SYNTAXP CHRCODE 'SEPR RDTBL))                ; SPACE the syntaxp check is to allow for space being a read
                                                                ; macro
             (SETQ SPACEFLG T)
             (READC FL)
             (GO LP1)))
          (SETQ TEM (COND
                       ((OR (EQ LISPXREADFN 'READ)
                            (IMAGESTREAMTYPEP T 'TEXT))         ; So the call will be linked, so the user can break on read.
                                                                ; TEXTSTREAMS must use READ
                        (READ FL RDTBL))
                       (T (APPLY* LISPXREADFN FL RDTBL))))
      ;; The reason for not embedding the setq in the ncon1 is that the act of reading may change L, e.g.  via a ^W read macro.
          (COND
             ((EQ TEM HISTSTR4)
              ;; fo implemeing read macros that are for effect only.  ignore the value returned by read.  if we had soft interrupts from iowaits, we
              ;; wouldnt needs this.
              (GO LP1)))
          (SETQ LINE (NCONC1 LINE TEM))
          (COND
             ((SYNTAXP (SETQ TEM (CHCON1 (LASTC FL)))
                     'RIGHTBRACKET RDTBL)
              ;; The reason why readline is driven by the last character insead of doing a peekc before reding is that due to eadmacros, it is
              ;; possible for several things to be read, e.g.  A B C '(FOO) terminated by square bracket should terminate the line.  However, it is not
              ;; sufficient just to check whether the value read is a list or not since '()' and NIL must also be treated differently.
              (GO OUT))
             ((NULL (SYNTAXP TEM 'RIGHTPAREN RDTBL))
              (GO LP))
             ((AND LISPXFLG (NULL SPACEFLG)
                   (NULL (CDDR LINE)))
              ;; A list terminates the line if if called from LISPX and is both the firt thing on a line and not preceded by a space.
              (GO OUT))
             (T (AND (EQ FL T)
                     (PRIN1 '|...| T))
                (GO LP)))
          (GO LP)
       OUT (COND
             ((AND (LISTP LINE)
                   CTRLUFLG)                                    ; User typed control-u during reading.
              (SETQ CTRLUFLG NIL)
              (COND
                 ((NULL (NLSETQ (EDITE LINE)))                  ; Exited with a STOP.
                  (SETQ REREADFLG 'ABORT)))))
          (COND
             (START (COND
                       ((NEQ START (CADADR READBUF))
                        (SHOULDNT))
                       (T                                       ; the rplaca is to handle small numbers
                          (RPLACA (CDADR READBUF)
                                  (SETN START (GETFILEPTR FL)))))
                    (SETFILEPTR FL -1)))
          (RETURN LINE)
       LP2 (COND
             ((EQ (CAR READBUF)
                  HISTSTR0)
              (SETQ READBUF (CDR READBUF))
              (RETURN LINE))
             ((NULL (SETQ READBUF (LISPXREADBUF READBUF)))
              ;; checks for things like HISTSTR2 etc.  this can occur if you redo an event contaiing a readline.  can also occur under a break if you
              ;; call a function which calls readline, because break unreads stuff, leaving the 'from event' tag on.
              (GO TOP)))
          (SETQ TEM READBUF)
          (SETQ READBUF (CDR READBUF))
          (SETQ LINE (NCONC1 LINE (CAR TEM)))
          (COND
             ((NULL READBUF)
              ;; really shouldnt happen, as there should be a '<c.r.>' marker.  however, in the case of a fix command, user might delete it.
              (RETURN LINE)))
          (GO LP2))))
```

(**REMPROPLIST**

```
(LAMBDA (ATM PROPS)                                             ; wt: 30-JUL-77 13 32
  (PROG (LST LST1 TEM)
        (COND
           ((NULL (SETQ LST1 (SETQ LST (GETPROPLIST ATM))))
            (RETURN NIL)))
    LP  (COND
           ((NLISTP LST1)
```

```
             (GO OUT))
            ((NOT (FMEMB (CAR LST1)
                         PROPS)))
            ((EQ LST1 LST)
             (SETQ LST (CDDR LST)))
            ((SETQ TEM (CDDR LST1))
             (RPLNODE2 LST1 TEM)
             (GO LP))
            (T                                          ; the last property, also not the first one.
               (RPLACD (NLEFT LST 1 LST1))
               (GO OUT)))
         (SETQ LST1 (CDDR LST1))
         (GO LP)
     OUT (SETPROPLIST ATM LST)
         (RETURN))))
```

(**RESETBUFS**

```
(NLAMBDA FORMS                                          (* |lmm| " 9-APR-78 00:27")
  (DECLARE (LOCALVARS . T))
  (PROG (($$BUFS (PROGN (LINBUF)
                        (SYSBUF)
                        (CLBUFS NIL T READBUF))))
        (RETURN (PROG1 (APPLY (FUNCTION PROGN)
                              FORMS
                              'INTERNAL)
                       (AND $$BUFS (BKBUFS $$BUFS)))))))
```

(**TAB**

```
(LAMBDA (POS MINSPACES FILE)
  (PROG (X)
        (COND
           ((NOT (IGREATERP (IPLUS (SETQ X (POSITION FILE))
                                   (OR (NUMBERP MINSPACES)
                                       1))
                            POS))
            (SPACES (IDIFFERENCE POS X)
                    FILE))
           ((EQ MINSPACES T)                            ; MINSPACES=T means space over to POS unless you are
                                                        ; already beyond it.

            )
           (T (TERPRI FILE)
              (SPACES POS FILE))))))
```

(**UNSAVED1**

```
(LAMBDA (FN TYP)                                        (* |bvm:| "29-Sep-86 23:24")
  (PROG (DEF PROP)
    TOP (COND
           ((NOT (LITATOM FN)))
           ((SETQ DEF (COND
                         ((SETQ PROP TYP)
                          (GET FN TYP))
                         ((GET FN (SETQ PROP 'EXPR)))
                         ((GET FN (SETQ PROP 'CODE)))
                         ((GET FN (SETQ PROP 'SUBR)))))
            (VIRGINFN FN T)
            (/REMPROP FN PROP)
            (COND
               ((NEQ DFNFLG T)
                (SAVEDEF FN)))
            (/PUTD FN DEF T)
            (AND ADDSPELLFLG (ADDSPELL FN))
            (RETURN PROP))
           ((OR (GETD FN)
                (GETPROPLIST FN))                       ; Not a misspelling
            (RETURN (COND
                       (TYP (CONCAT "(" TYP " not found)"))
                       (T "(nothing found)"))))
           ((SETQ PROP (FNCHECK FN T))
            (SETQ FN PROP)
            (GO TOP)))
        (ERROR FN '"not a function"))))
```

(**WRITEFILE**

```
(LAMBDA (X FILE)                                        (* |bvm:| "30-Aug-86 16:45")
```

;; X is a list of expression (or an atom that evaluates to a list) X is written on FILE. If X begins with a PRINTDATE expression, a new one is written.
;; Following the PRETTYDEF conventions, if FILE is listed, it is left open. Otherwise a stop is printed and it is closed.

```
  (WITH-READER-ENVIRONMENT *OLD-INTERLISP-READ-ENVIRONMENT*
      (RESETLST
          (PROG (STREAM OPENED)
                (COND
                   ((LISTP FILE)
```

```
                    (SETQ FILE (CAR FILE))
                    (SETQ OPENED T)))
                (RESETSAVE NIL (LIST (FUNCTION CLOSE-AND-MAYBE-DELETE)
                                     (SETQ STREAM (OPENSTREAM FILE 'OUTPUT))))
                (RESETSAVE (OUTPUT STREAM))
                (COND
                   ((ATOM X)
                    (SETQ X (EVAL X))))
                (PRIN1 "
                       (PRIN1 (QUOTE \"
                       WRITEFILE OF ")
                (PRIN2 (SETQ FILE (FULLNAME STREAM)))
                (PRIN1 " MADE BY ")
                (PRIN1 (USERNAME))
                (PRIN1 " ON ")
                (PRIN1 (DATE))
                (PRIN1 "
                       \")T)

                       ")
                (|for| X1 |in| X |do| (PRINTDEF X1 NIL (EQ (CAR (LISTP X1))
                                                           'DEFINEQ))
                                     (TERPRI))
                (|if| (NULL OPENED)
                    |then| (ENDFILE))
                (RETURN FILE))))))
```

(CLOSE-AND-MAYBE-DELETE

```
  (LAMBDA (STREAM)                                          ; Edited 19-Mar-87 16:43 by jrb:
```

;;; For use in RESETSAVE. Closes STREAM, and if happened under error, deletes the file

```
    (|if| (OPENP STREAM)
        |then| (SETQ STREAM (CLOSEF STREAM)))
    (AND RESETSTATE (DELFILE STREAM))))
```

(UNSAFE.TO.MODIFY

```
  (LAMBDA (FN OPTION)                                       (* |lmm| "31-Jul-85 02:06")
    (|if| (FMEMB FN UNSAFE.TO.MODIFY.FNS)
        |then| (PRINTOUT T "Warning: " FN " may be unsafe to " (OR OPTION "modify")
                       " -- continue? ")
               (|if| (EQ (|if| (GETD 'ASKUSER)
                           |then| (ASKUSER DWIMWAIT 'N)
                         |else| (READ T))
                         'Y)
                   |then| NIL
                 |else| T))))

)

(RPAQQ UNSAFE.TO.MODIFY.FNS
       (/PUT /PUTD /REMPROP ADDCHAR ADDCHAR ADDSPELL ADVISEWDS ALLOCSTRING APPLY APPLY ASSOC AWAIT.EVENT
             BITBLT.ERASE BITMAPCOPY BITMAPCREATE BKBITBLT BLOCK BLOCK BLTCHAR BLTCHAR BLTSHADE BREAK BREAK0
             BREAK1 CHARSET CHCON1 CLEAR.LINE? CLOCK CLOCKDIFFERENCE CLOSEW CONCAT CREATEW CURSOR CURSORHOTSPOT
             DELETETO DO.CRLF DRAWLINE DSPBACKUP DSPCLIPPINGREGION DSPCLIPPINGREGION DSPCREATE DSPDESTINATION
             DSPFILL DSPFONT DSPLEFTMARGIN DSPRIGHTMARGIN DSPSCROLL DSPSOURCETYPE DSPXOFFSET DSPXPOSITION
             DSPYPOSITION EQLENGTH EQP EQUAL ERASE.TO.END.OF.LINE ERASE.TO.END.OF.PAGE ERRORMESS1 ERRORSET EVAL
             EVALQT EXPRP FASSOC FILENAMEFIELD FIXR FLIPCURSOR FLAST FMEMB GENSYM GETHASH GETMOUSESTATE GETPROP
             GETSTREAM GETWINDOWUSERPROP HELP HISTORYSAVE IDATE IMAGESTREAMTYPEP IMOD INIT.CURSOR INTEGERLENGTH
             INTERRUPTABLE INTERSECTREGIONS IREMAINDER LAST LASTC LISPX LISPX/ LISPXFIND LISPXFIND1 LISPXPRINT
             LISPXPUT LISPXPUT LISPXREAD LISPXREADBUF LISPXUNREAD LISTGET LISTPUT MEMB MKATOM MKSTRING
             MONITOR.AWAIT.EVENT MOVETOUPPERLEFT NOTIFY.EVENT NTH NTHCHARCODE OBTAIN.MONITORLOCK OPENW OPENWP
             OVERFLOW? PACK* PAGEHEIGHT PRIN1 PRIN1 PRIN2 PRIN2 PRIN3 PRIN3 PRINT PRINT PRINTCCODE PRINTLEVEL
             PROGN PROMPTCHAR PUTWINDOWPROP QUOTE READ CL:READ READLINE READLINE READP REALSTKNTH REGIONP
             RELEASE.PUP RELSTK RESETRESTORE RESHOWTITLE RETFROM RPLCHARCODE RPLSTRING SETCURSOR SETTERMTABLE
             SHOWPRIN2 SHOWPRINT SHOWWFRAME SHOWWTITLE SKIPSEPRS SPACES STKPOS STREAMP SUBATOM SUBSTRING SYNTAXP
             TERPRI TIMEREXPIRED? TIMEREXPIRED? TOTOPW TTBIN TTBITWIDTH TTCRLF TTDELETELINE TTSKREAD
             TTWAITFORINPUT TTWAITFORINPUT TTYDISPLAYSTREAM TTYIN TTYIN.CLEANUP TTYIN.FINISH TTYIN.READ
             TTYIN.SETUP TTYIN1 TTYIN1RESTART TTYINREAD TYPENAME UNBREAK0 UNDOSAVE UNPACKFILENAME.STRING WFROMDS
             WINDOW.MOUSE.HANDLER))
```

;; FILEDATE, for finding out the creation date of source files, from the compiled files.

;; FASL isn't loaded when MACHINEINDEPENDENT is, so we have to fake the FASL checker for now. It's defined in FASLOAD.

```
(DEFINEQ
```

(FILEDATE

```
  (LAMBDA (FILE CFLG)                                       ; Edited 17-Feb-89 11:26 by jds
                                                            ; CFLG IS T FOR COMPILED FILES
    (COND
       (FILE (CAR (NLSETQ (RESETLST
                          (PROG (STREAM OLDPTR VALUE)
                                (COND
                                   ((SETQ STREAM (OPENP FILE 'INPUT))
```

```
                                        (SETQ OLDPTR (GETFILEPTR STREAM)))
                                       (T                          ; OPENSTREAM used instead of INFILEP to allow for error
                                                                   ; correction.
                                          (RESETSAVE NIL (LIST 'CLOSEF (SETQ STREAM (OPENSTREAM FILE
                                                                                              'INPUT))))))
                                ;; This code used to have some gross kludgery for checking file dates of grouped files during the loadup
                                ;; procedure, now gone -bvm
                                (COND
                                   ((RANDACCESSP STREAM)
                                    (SETFILEPTR STREAM 0)
                                    (COND
                                       ((SETQ VALUE (FASL-FILEDATE STREAM CFLG))
                                        ;; Aha, a Dfasl file
                                        ;; Having decided it's a DFASL, FASL-FILEDATE returned the date, and it's in VALUE
                                        ;; already.

                                        )
                                       (T                          ; Any other filetype
                                          (SETFILEPTR STREAM 0)
                                          (CL:MULTIPLE-VALUE-BIND (ENV FORM)
                                              (\\PARSE-FILE-HEADER STREAM 'RETURN)
                                            (COND
                                               ((AND CFLG (LISTP FORM))
                                                                   ; First expression is for compiled file, next one is its source
                                                (SETQ FORM (WITH-READER-ENVIRONMENT ENV (READ STREAM)))))
                                            (COND
                                               ((EQ (CAR (LISTP FORM))
                                                    'FILECREATED)
                                                (SETQ VALUE (CAR (LISTP (CDR FORM))))))))))
                                (COND
                                   (OLDPTR (SETFILEPTR STREAM OLDPTR)))
                                (RETURN VALUE)))))))))

)

(MOVD? 'NILL 'FASL-FILEDATE)

(MOVD? 'CL:FMAKUNBOUND 'UNDOABLY-FMAKUNBOUND)
```

;; used in FNS.PUTDEF before CMLUNDO loaded

;; Functions for retrieving and remembering FILEMAPs and file reader environments

```
(DEFINEQ
```

(**FILEMAP**

```
  (NLAMBDA (FILEMAP)                                               (* |bvm:| "27-Aug-86 23:41")
```

;;; Called by the FILEMAP expression at the end of every standard Interlisp file

```
    (DECLARE (USEDFREE FILECREATEDLST))                            ; FILECREATEDLST bound in LOAD or LOADFNS and set by
                                                                   ; FILECREATED
    (PUTFILEMAP (FULLNAME (GETSTREAM NIL 'INPUT))
           FILEMAP FILECREATEDLST NIL T)))
```

(**PARSE-FILE-HEADER**

```
  (LAMBDA (STREAM FILECREATEDFN RETURNFORM INITIALENV)              (* |bvm:| " 8-Sep-86 12:37")
```

;;; Parses the stuff at front of STREAM, which is assumed positioned at zero, and returns as its first value a reader environment for the file, or NIL if this ;;; is not a Lisp source file. If a FILECREATED expression is found, then calls FILECREATEDFN with the file pointer positioned immediately after the ;;; symbol FILECREATED, and returns the fn's value as its second value. FILECREATEDFN = RETURN returns the entire FILECREATED expression. ;;; Finally, in the case where no FILECREATED expression was found, returns as second value the actual first expression if RETURNFORM is true (this ;;; is needed for callers that don't want to lose when the stream is non-randaccess).

;;; The first expression on the file is read in the current reader environment. Usually this wants to be IL.

```
    (WITH-READER-ENVIRONMENT (OR INITIALENV *OLD-INTERLISP-READ-ENVIRONMENT*)
        (SELCHARQ (SKIPSEPRCODES STREAM)
             (";"                                                  ; Assume is common lisp file
                  *COMMON-LISP-READ-ENVIRONMENT*)
             ("("                                                  ; Start of Lisp expression, could be either DEFINE-FILE-INFO or
                                                                   ; FILECREATED
                  (PROG (ENV FIRSTSYM RESULT HERE)
                    TOP (SETQ HERE (GETFILEPTR STREAM))
                        (READCCODE STREAM)
                        (SETQ FIRSTSYM (AND (SYNTAXP (SKIPSEPRCODES STREAM)
                                                     'OTHER)
                                            (RATOM STREAM)))
                        (COND
                           ((AND (EQ FIRSTSYM 'DEFINE-FILE-INFO)
                                 (NULL ENV))
                            (SETQ ENV (\\DO-DEFINE-FILE-INFO STREAM (CL:READ-DELIMITED-LIST (CHARCODE ")")
                                                                                            STREAM)))
                            (COND
```

```
                              ((AND FILECREATEDFN (EQ (SKIPSEPRCODES STREAM)
                                                      (CHARCODE "(")))
                               (SET-READER-ENVIRONMENT ENV)
                               (GO TOP))
                              (T ;; Odd case--a DEFINE-FILE-INFO expression but no FILECREATED afterwards or caller doesn't want to
                                 ;; see it
                                 (RETURN (CL:VALUES ENV NIL HERE))))))
                         (|if| (EQ FIRSTSYM 'FILECREATED)
                             |then| (OR ENV (SETQ ENV *OLD-INTERLISP-READ-ENVIRONMENT*))
                                    (SETQ RESULT (SELECTQ FILECREATEDFN
                                                          (RETURN (CONS 'FILECREATED (CL:READ-DELIMITED-LIST
                                                                                       (CHARCODE ")")
                                                                                       STREAM)))
                                                          (NIL NIL)
                                                          (CL:FUNCALL FILECREATEDFN STREAM)))
                           |elseif| RETURNFORM
                             |then| (SETQ RESULT (CL:READ-DELIMITED-LIST (CHARCODE ")")
                                                                        STREAM)))
                         (RETURN (CL:VALUES ENV RESULT HERE))))
          NIL))))
```

(**GET-ENVIRONMENT-AND-FILEMAP**

```
(LAMBDA (STREAM DONTCACHE)                                          (* |bvm:| "26-Sep-86 11:39")
```

;; Returns three values: the stream's reader environment, its filemap, either obtained from the file itself, or from its property list, and the byte ;; location where the FILECREATED expression starts.

```
(LET ((FULL (COND
               ((STREAMP STREAM)
                (FULLNAME STREAM))
               (T STREAM)))
      MAPENTRY MAP ENV OLDPOS)
     (SETQ MAPENTRY (GETHASH FULL *FILEMAP-HASH*))
     (COND
        ((AND MAPENTRY (OR (SETQ MAP (|fetch| FMFILEMAP |of| MAPENTRY))
                           (NULL USEMAPFLG)))
```

;; Have all we need. Return the map only if USEMAPFLG is true or the map was obtained by scanning the file

```
         (|replace| FMRECENT? |of| MAPENTRY |with| T)
         (CL:VALUES (|fetch| FMENVIRONMENT |of| MAPENTRY)
                 (AND MAP (OR USEMAPFLG (NOT (|fetch| FMFROMFILE? |of| MAPENTRY)))
                      MAP)
                 (|fetch| FMFILECREATEDLOC |of| MAPENTRY)
                 (|fetch| FMFILECREATEDLST |of| MAPENTRY)))
        ((OR (NOT (SETQ STREAM (OPENP STREAM 'INPUT)))
             (NOT (RANDACCESSP STREAM)))                            ; Out of luck
         NIL)
        (T                                                          ; Have to read file
           (SETQ OLDPOS (GETFILEPTR STREAM))
           (SETFILEPTR STREAM 0)
           (CL:MULTIPLE-VALUE-BIND (ENV NEWMAP FCLOCATION)
               (\\PARSE-FILE-HEADER STREAM (COND
                                             ((AND (NULL MAP)
                                                   USEMAPFLG)
                                              (FUNCTION GET-FILEMAP-FROM-FILECREATED))))
             (SETFILEPTR STREAM OLDPOS)
             (COND
                ((AND NEWMAP (NOT DONTCACHE))
                 (PUTFILEMAP FULL NEWMAP NIL ENV T FCLOCATION)))
             (CL:VALUES ENV (OR NEWMAP MAP)
                    FCLOCATION))))))
```

(**LOOKUP-ENVIRONMENT-AND-FILEMAP**

```
(LAMBDA (FULL ROOTNAMEP)                                            ; Edited 4-May-88 15:30 by bvm
```

;; Returns four values: the file's reader environment, its filemap, either obtained from the file itself, or from its property list, the byte location where ;; the FILECREATED expression starts, and the FILECREATEDLST of the file (used by ADDFILE). Unlike GET-ENVIRONMENT-AND-FILEMAP, ;; this function merely looks up cached info. If ROOTNAMEP is true, then FULLNAME is actually a root name, and we want to look up the most ;; recent.

```
(LET ((HIGHEST-VERSION -1)
      MAPENTRY)
     (|if| ROOTNAMEP
         |then| (MAPHASH *FILEMAP-HASH*
                         (FUNCTION (LAMBDA (ENTRY KEY)
                                     (LET (V)
                                          (|if| (AND (STRPOS FULL KEY NIL NIL NIL NIL UPPERCASEARRAY)
                                                     (STRING-EQUAL FULL (ROOTFILENAME KEY))
                                                     (IGREATERP (SETQ V (OR (FILENAMEFIELD KEY 'VERSION)
                                                                            0))
                                                                HIGHEST-VERSION))
                                              |then| (SETQ MAPENTRY ENTRY)
                                                     (SETQ HIGHEST-VERSION V))))))
       |else| (SETQ MAPENTRY (GETHASH FULL *FILEMAP-HASH*)))
     (|if| MAPENTRY
         |then| (|replace| FMRECENT? |of| MAPENTRY |with| T)
```

```
                    (CL:VALUES (|fetch| FMENVIRONMENT |of| MAPENTRY)
                           (|fetch| FMFILEMAP |of| MAPENTRY)
                           (|fetch| FMFILECREATEDLOC |of| MAPENTRY)
                           (|fetch| FMFILECREATEDLST |of| MAPENTRY))))))
```

(GET-FILEMAP-FROM-FILECREATED

```
(LAMBDA (STREAM)                                                    (* |bvm:| "29-Aug-86 15:06")
```

;; get map from address shown in FILECREATED expression, which is of form (FILECREATED file date mapaddr)

```
    (SKREAD STREAM)
    (SKREAD STREAM)
    (CAR (NLSETQ (LET ((MAPADDR (READ STREAM)))
                      (COND
                         ((AND (FIXP MAPADDR)
                               (LESSP MAPADDR (GETEOFPTR STREAM))
                               (PROGN (SETFILEPTR STREAM MAPADDR)
                                      (EQ (SKIPSEPRCODES STREAM)
                                          (CHARCODE "(")))
                               (EQ (CAR (SETQ MAPADDR (READ STREAM)))
                                   'FILEMAP))
                          (CADR MAPADDR))))))))
```

(\\FILEMAP-HASHOVERFLOW

```
(LAMBDA (HARRAY)                                                    (* |bvm:| "26-Sep-86 12:11")
```

;;; Called when *FILEMAP-HASH* overflows. Trim back old entries

```
    (LET ((NUMENTRIES (HARRAYPROP HARRAY 'NUMKEYS))
          ENTRIES)
         (|if| (> NUMENTRIES *FILEMAP-LIMIT*)
             |then| (MAPHASH HARRAY (FUNCTION (LAMBDA (VAL KEY)    ; Gather up contents of table
                                                (LET ((ROOT (|fetch| FMROOTNAME |of| VAL))
                                                      TEM)
                                                     (|if| (NOT (SETQ TEM (FASSOC ROOT ENTRIES)))
                                                         |then| (|push| ENTRIES (SETQ TEM (LIST ROOT))))
                                                     (|push| (CDR TEM)
                                                             (CONS (|if| (CDR (|fetch| FMFILECREATEDLST |of| VAL))
                                                                       |then|
                                                                       ; compiled file, don't keep if there is no other reason to
                                                                       keep
                                                                     |else| (FILENAMEFIELD KEY 'VERSION))
                                                                   (CONS KEY VAL))))))
                    ;; each element of ENTRIES is (root . versions), where each version is (vers# fullname . hashvalue)
                    (|for| GROUP |in| ENTRIES |bind| ONFILELST PAIR NFLUSH DATES
                       |do| (SETQ ONFILELST (MEMB (CAR GROUP)
                                                  FILELST))
                            (SETQ NFLUSH (- (LENGTH (CDR GROUP))
                                            *FILEMAP-VERSIONS*))
                            (|for| TAIL |on| (PROGN                 ; Sort files by increasing version
                                                    (SORT (CDR GROUP)
                                                          T))
                               |as| I |from| 1 |do| (SETQ PAIR (CDAR TAIL))
                                                    (|if| (AND (<= I NFLUSH)
                                                               (OR (NULL (SETQ DATES (GET (CAR GROUP)
                                                                                          'FILEDATES)))
                                                                   (NOT (STRING.EQUAL (CDAR DATES)
                                                                                      (CAR PAIR)))))
                                                        |then|
                                ;; flush old versions until we have gotten down to limit.  The STRING.EQUAL test is because the
                                ;; "current version" of a file might have a lower version number (being on a different directory) than
                                ;; the highest version you have looked at anywhere
                                                              (REMHASH (CAR PAIR)
                                                                       HARRAY)
                                                              (|add| NUMENTRIES -1)
                                                      |elseif| (|fetch| FMRECENT? |of| (CDR PAIR))
                                                        |then|          ; spare recently touched files, but clear the flag
                                                              (|replace| FMRECENT? |of| (CDR PAIR) |with| NIL)
                                                      |elseif| (OR (NOT ONFILELST)
                                                                   (CDR TAIL))
                                                        |then|          ; trim maps not looked at recently, but spare the highest version
                                                                        ; of anything on filelst
                                                              (REMHASH (CAR PAIR)
                                                                       HARRAY)
                                                              (|add| NUMENTRIES -1))))
                    ;; finally say how big to rehash the array.  Normally we want it not to change size.
                    (IMAX *FILEMAP-LIMIT* (FIXR (FTIMES NUMENTRIES 1.2)))))))
```

(FLUSHFILEMAPS

```
(LAMBDA (ROOTNAME)                                                  (* |bvm:| "26-Sep-86 11:37")
  (|if| (EQ ROOTNAME T)
      |then| (CLRHASH *FILEMAP-HASH*)
```

```
      |else| (MAPHASH *FILEMAP-HASH* (FUNCTION (LAMBDA (ME FULLNAME)
                                                 (|if| (STRING-EQUAL (|fetch| FMROOTNAME |of| ME)
                                                                     ROOTNAME)
                                                       |then| (REMHASH FULLNAME *FILEMAP-HASH*))))))
    ROOTNAME))
```

(LISPSOURCEFILEP

```
  (LAMBDA (FILE)                                                    (* |bvm:| "29-Sep-86 23:15")
```

;;; If the first few characters of FILE 'look like' those output by MAKEFILE then return the alleged address in the file of its FILEMAP expression.

```
    (RESETLST
        (|if| (NOT (STREAMP FILE))
            |then| (RESETSAVE NIL (LIST 'CLOSEF (SETQ FILE (OPENSTREAM FILE 'INPUT)))))
        (|if| (RANDACCESSP FILE)
            |then| (LET ((HERE (GETFILEPTR FILE)))
                        (PROG1 (CL:MULTIPLE-VALUE-BIND (ENV MAP)
                                   (\\PARSE-FILE-HEADER FILE (FUNCTION (LAMBDA (STREAM)
                                                                       ; Pointed now right after the FILECREATED expression
                                                                              (CAR (NLSETQ (SKREAD STREAM)
                                                                                           (SKREAD STREAM)
                                                                                           (FIXP (READ STREAM)))))))
                                   MAP)
                               (SETFILEPTR FILE HERE)))))
```

(GETFILEMAP

```
  (LAMBDA (STREAM FL)                                               (* |bvm:| "27-Aug-86 15:48")
```

;;; Value is map for STREAM either obtained from the file itself, or from its property list. STREAM is presumed open. FL is (NAMEFIELD STREAM T)

```
    (AND USEMAPFLG (CL:MULTIPLE-VALUE-BIND (ENV MAP)
                       (GET-ENVIRONMENT-AND-FILEMAP STREAM)
                   MAP))))
```

(PUTFILEMAP

```
  (LAMBDA (FILE FILEMAP FILCREATEDLST ENV FROMFILE? FCLOCATION)     (* |bvm:| "26-Sep-86 11:51")
                                                                    ; Called from: LOAD LOADFNS PRETTYDEF FILEMAP
```

;; As far as I can tell, the only use for FILCREATEDLST is to tell ADDFILE in LOADFNS that the file is a compiled file

```
    (|if| (NULL FILEMAP)
        |then| (REMHASH FILE *FILEMAP-HASH*)
      |elseif| BUILDMAPFLG
        |then| (LET* ((OLDENTRY (GETHASH FILE *FILEMAP-HASH*))
                      (NEWENTRY (|create| FILEMAPHASH |using| OLDENTRY FMFROMFILE? _ FROMFILE? FMRECENT? _ T)))
                     (|if| (NULL OLDENTRY)
                         |then| (|replace| FMROOTNAME |of| NEWENTRY |with| (ROOTFILENAME FILE (CDR FILCREATEDLST))))
                     (|if| ENV
                         |then| (|replace| FMENVIRONMENT |of| NEWENTRY |with| ENV)
                       |elseif| (NULL OLDENTRY)
                         |then| (|replace| FMENVIRONMENT |of| NEWENTRY |with| (MAKE-READER-ENVIRONMENT)))
                     (|if| (LISTP FILEMAP)
                         |then| (|replace| FMFILEMAP |of| NEWENTRY |with| FILEMAP))
                     (|if| FCLOCATION
                         |then| (|replace| FMFILECREATEDLOC |of| NEWENTRY |with| FCLOCATION))
                     (|if| FILCREATEDLST
                         |then| (|replace| FMFILECREATEDLST |of| NEWENTRY |with| FILCREATEDLST))
                     (PUTHASH FILE NEWENTRY *FILEMAP-HASH*))))
```

(UPDATEFILEMAP

```
  (LAMBDA (STREAM FILEMAP)                                          (* |bvm:| "24-Oct-86 17:15")
```

;;; Writes new FILEMAP on file currently open as STREAM. If we return T, the stream has been closed.

;;; This has little hope of working any more.

```
    (|if| NIL
        |then|                                                      ; This has little hope of working any more
             (LET ((DECLARESTRING (CONCAT "(DECLARE: DONTCOPY
                                          " "(FILEMAP"))
                   FILEMAPLOCADR TEM FILEMAPADR FILEMAPLOCLEN FULLNAME)
                  (SETFILEPTR STREAM 0)
                  (SKIPSEPRS STREAM)                                ; Could be some font shifts or other garbage
                  (READC STREAM)                                    ; Skip paren or bracket
                  (|if| (AND (EQ (RATOM STREAM)
                                 'FILECREATED)
                             (PROGN (SKREAD STREAM)                 ; Date
                                    (SKREAD STREAM)                 ; Name
                                    (|do| (COND
                                              ((EQ (SETQ TEM (READCCODE STREAM))
                                                   (CHARCODE SPACE)); found a space
                                               (RETURN T))
                                              ((NOT (SYNTAXP TEM 'SEPRCHAR))
                                                                    ; no spaces, lose
```

```
                                (RETURN)))))
                        (FIXP (SETQ FILEMAPADR (PROGN        ; skip over seprs
                                                     (SETQ FILEMAPLOCADR (GETFILEPTR STREAM))
                                                      ; Address of first character of file-map location
                                                     (PROG1 (RATOM STREAM)
                                                         (SETQ FILEMAPLOCLEN (IDIFFERENCE (GETFILEPTR STREAM
                                                                                                        )
                                                                                          FILEMAPLOCADR))))))
                        (SETQ FILEMAPADR (OR (FFILEPOS DECLARESTRING STREAM (FIX (TIMES FILEMAPADR 0.9)))
                                             (FFILEPOS DECLARESTRING STREAM 0)))
                        (EQ (PROGN (SKREAD STREAM)
                                   (RATOM STREAM))
                            'STOP)
                        (ILEQ (NCHARS FILEMAPADR T)
                              FILEMAPLOCLEN))
                |then| ;; normally, this will be called so that we are positioned at the filemap.  --- check for (FILECREATED & & number
                       ;; --) first to avoid searching compiled files for filemap.
                       (SETQ FULLNAME (CLOSEF STREAM))
                       (|if| (SETQ STREAM (CAR (NLSETQ (OPENSTREAM FULLNAME 'BOTH 'OLD NIL '(
                                                                                          DON\'T.CHANGE.DATE
                                                                                           )))))
                           |then| (RESETLST
                                      (RESETSAVE NIL (LIST 'CLOSEF STREAM))
                                      (SETFILEPTR STREAM FILEMAPADR)
                                      (PRIN3 "(DECLARE: DONTCOPY
                                             " STREAM)
                                      (SETQ FILEMAPADR (GETFILEPTR STREAM))
                                      (PRIN3 "(FILEMAP " STREAM)
                                      (POSITION STREAM (CONSTANT (NCHARS "(FILEMAP ")))
                                      (LET ((*PRINT-RADIX* 10))
                                           (PRIN2 FILEMAP STREAM))
                                      (PRIN1 "))" STREAM)
                                      (TERPRI STREAM)
                                      (PRINT 'STOP STREAM)
                                      (SETFILEPTR STREAM FILEMAPLOCADR)
                                      (PRINTNUM (LIST 'FIX FILEMAPLOCLEN)
                                                FILEMAPADR STREAM)
                                      (COND
                                         ((NEQ DFNFLG T)
                                          (PRIN3 "****rewrote file map for " T)
                                          (PRINT FULLNAME T T)))))
                        T)))))
```

(**PRINT-READER-ENVIRONMENT**

```
  (LAMBDA (ENV STREAM)                                    (* |bvm:| "24-Oct-86 15:53")
```

;;; If ENV is not the old default interlisp reader environment, writes a DEFINE-FILE-INFO expression on STREAM that will produce this environment
;;; when the file is loaded.

```
    (|if| (NOT (EQUAL-READER-ENVIRONMENT ENV *OLD-INTERLISP-READ-ENVIRONMENT*))
        |then| (LET ((*PACKAGE* *INTERLISP-PACKAGE*)
                     (*PRINT-BASE* 10)
                     PKG)
                    (PRINT (CONS 'DEFINE-FILE-INFO (OR (|fetch| RESPEC |of| ENV)
                                                       `(,@(AND (SETQ PKG (|fetch| REPACKAGE |of| ENV))
                                                                `(:PACKAGE ,(CL:PACKAGE-NAME PKG)))
                                                         :READTABLE
                                                         ,(READTABLEPROP (|fetch| REREADTABLE |of| ENV)
                                                                         'NAME)
                                                         :BASE
                                                         ,(|fetch| REBASE |of| ENV))))
                                 STREAM FILERDTBL)))))

)

(RPAQ? *FILEMAP-LIMIT* 20)

(RPAQ? *FILEMAP-VERSIONS* 2)

(RPAQ? *FILEMAP-HASH* (HASHARRAY *FILEMAP-LIMIT* (FUNCTION \\FILEMAP-HASHOVERFLOW)
                                 (FUNCTION STRING-EQUAL-HASHBITS)
                                 (FUNCTION STRING.EQUAL)))

(DECLARE\: EVAL@COMPILE DONTCOPY

(DECLARE\: EVAL@COMPILE

(RECORD FILEMAPHASH (FMENVIRONMENT FMROOTNAME FMFROMFILE? FMRECENT? FMFILECREATEDLOC FMFILECREATEDLST
                                                                                                . FMFILEMAP)
      )
)

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS *FILEMAP-LIMIT* *FILEMAP-VERSIONS* *FILEMAP-HASH*)
```

```
)
)


          (* * LVLPRINT)

(DEFINEQ
```

(LVLPRINT

```
  (LAMBDA (X FILE CARLVL CDRLVL TAIL)                    (* |wt:| 12-MAY-76 22 6)
    (LVLPRIN2 X FILE CARLVL CDRLVL TAIL)
    (TERPRI FILE)
    X))
```

(LVLPRIN1

```
  (LAMBDA (X FILE CARLVL CDRLVL TAIL)
    (DECLARE (SPECVARS FILE PRIN2FLG))
    (PROG (PRIN2FLG)
          (LVLPRIN X CARLVL CDRLVL TAIL)
          (RETURN X))))
```

(LVLPRIN2

```
  (LAMBDA (X FILE CARLVL CDRLVL TAIL)
    (DECLARE (SPECVARS FILE PRIN2FLG))                   (* |wt:| 12-MAY-76 22 6)
    (PROG ((PRIN2FLG T))
          (LVLPRIN X CARLVL CDRLVL TAIL)
          (RETURN X))))
```

(LVLPRIN

```
  (LAMBDA (X CARLVL CDRLVL TAIL)                         ; Edited 10-Nov-87 13:10 by jds
                                                         ; wt: 12-MAY-76 22 23

    (COND
       ((NLISTP X)
        (COND
           ((AND TAIL (EQ X (CDR (LAST TAIL)))
                 (NOT (MEMB X TAIL)))
            (PRIN1 '"...  . " FILE)
            (COND
               (PRIN2FLG (PRIN2 X FILE T))
               (T (PRIN1 X FILE)))
            ;; We use standard system read table for printing on grounds that even if this is going to a file, user is only dumping it with bpnt to look at
            ;; it, not to read it back in.
            (PRIN1 ")" FILE))
           (PRIN2FLG (PRIN2 X FILE T))
           (T (PRIN1 X FILE))))
       (T (PRIN1 (COND
                    ((AND TAIL (TAILP X TAIL))           ; Tail
                     '"... ")
                    (T "("))
                 FILE)
          (LVLPRIN0 X CARLVL CDRLVL)
          (PRIN1 ")" FILE)))))
```

(LVLPRIN0

```
  (LAMBDA (X CARLVL CDRLVL)                              ; Edited 10-Nov-87 13:11 by jds
                                                         ; LVLPRIN0 is like subprint.  it prints the interior segment of a list

    (AND (EQ (CAR X)
             CLISPTRANFLG)
         (SETQ X (CDDR X)))
    (PROG ((CDRLVL0 CDRLVL))
          (GO LP1)
      LP  (COND
             ((NULL (SETQ X (CDR X)))
              (RETURN))
             ((NLISTP X)
              (PRIN1 '" . " FILE)
              (COND
                 (PRIN2FLG (PRIN2 X FILE T))
                 (T (PRIN1 X FILE)))
              (RETURN))
             (T (SPACES 1 FILE)))
      LP1 (COND
             ((EQ CDRLVL 0)
              (PRIN1 "--" FILE)
              (RETURN))
             ((NLISTP (CAR X))
              (COND
                 (PRIN2FLG (PRIN2 (CAR X)
                                  FILE T T))
                 (T (PRIN1 (CAR X)
                           FILE))))
             ((OR (EQ CARLVL 0)
```

```
                (AND CDRLVL0 (EQ (SUB1 CDRLVL0)
                                 0)))                          ; the reason for the second check is that why bother to recurse
                                                               ; only to print (--).  & is better
             (PRIN1 '& FILE))
            ((AND (EQ FILE T)
                  (SUPERPRINTEQ (CAAR X)
                         COMMENTFLG)
                  **COMMENT**FLG)
             (PRIN1 **COMMENT**FLG FILE))
            (T (PRIN1 '\( FILE)
               (LVLPRIN0 (CAR X)
                      (AND CARLVL (IPLUS CARLVL (COND
                                                  ((MINUSP CARLVL)
                                                   1)
                                                  (T -1))))
                      (AND CDRLVL0 (SUB1 CDRLVL0)))
                  (PRIN1 '\) FILE)))
          (AND CDRLVL (SETQ CDRLVL (SUB1 CDRLVL)))
          (GO LP))))

)

;; used by PRINTOUT

(DEFINEQ
```

(FLUSHRIGHT

```
  (LAMBDA (POS X MIN P2FLAG CENTERFLAG FILE)                   (* |lmm| "10-Feb-86 12:10")
```

;; Right-flushes X at position POS. If P2FLAG, uses PRIN2-pname; if CENTERFLAG, centers X between current position and POS

```
    (SETQ POS (IDIFFERENCE (COND
                             ((MINUSP POS)
                              (IDIFFERENCE (POSITION FILE)
                                     POS))
                             ((ZEROP POS)
                              (LINELENGTH NIL FILE))
                             (T POS))
                      (NCHARS X P2FLAG)))
    (COND
       (CENTERFLAG (SETQ POS (QUOTIENT (IPLUS POS (POSITION FILE))
                                       2))))
    (TAB POS MIN FILE)
    (COND
       (P2FLAG (PRIN2 X FILE))
       (T (PRIN1 X FILE)))))
```

(PRINTPARA

```
  (LAMBDA (LMARG RMARG LIST P2FLAG PARENFLAG FILE)             (* |rmk:| "22-MAY-81 13:45")
```

;; Prints LIST in paragraph format. The first line starts at the current line position, but all subsequent lines begin at LMARG (0 is the left margin, NIL
;; is the current POSITION, negative LMARG is (POSITION) + LMARG). Printing is with PRIN2 if P2FLAG, otherwise PRIN1. The right margin is
;; at column RMARG if RMARG is positive, (LINELENGTH NIL FILE) minus RMARG for RMARG LEQ 0

```
    (DECLARE (SPECVARS LMARG RMARG P2FLAG FILE))
    (COND
       ((NULL LMARG)
        (SETQ LMARG (POSITION FILE)))
       ((MINUSP LMARG)
        (SETQ LMARG (IDIFFERENCE (POSITION FILE)
                          LMARG))))
    (COND
       ((ILEQ RMARG 0)
        (SETQ RMARG (IPLUS RMARG (LINELENGTH NIL FILE)))))
    (POSITION FILE (PRINTPARA1 LIST (POSITION FILE)
                          (COND
                             (PARENFLAG 1)
                             (T 0))
                          (COND
                             (PARENFLAG 1)
                             (T 0))))))
```

(PRINTPARA1

```
  (LAMBDA (LIST POS OPENCOUNT CLOSECOUNT)                      (* |wt:| " 9-SEP-78 09:54")
```

;; PRIN3 and PRIN4 are used here, so we don't have to set and unset LINELENGTH. We keep our own idea of the current line position in POS,
;; which is returned as the value of PRINTPARA1. OPENCOUNT is the number of open parens that must precede the first non-list we print,
;; CLOSECOUNT is the number of close parens that should follow the last non-list we print. They are passed as arguments so that their numbers
;; can be taken into account in deciding whether a non-list fits on the line or not.

```
    (PROG ($$VAL L LEN (CC 0))
      $$LP
          (SETQ L (CAR (OR (LISTP LIST)
                           (GO $$OUT))))                       ; POS is the correct column position at the end of each iteration
          (COND
             ((NLISTP (CDR LIST))
              (SETQ CC CLOSECOUNT)))                           ; The last iteration.  Now we really want to use CLOSECOUNT,
                                                               ; so we move it to CC.
```

```
          (COND
             ((LISTP L)
              (SETQ POS (PRINTPARA1 L POS (ADD1 OPENCOUNT)
                                (ADD1 CC)))
              (SETQ OPENCOUNT 0)                                    ; The lower call printed the open and closed parens, including
                                                                    ; the ones for this level, if any.
              (SETQ CC 0))
             (T (COND
                   ((ILESSP RMARG (IPLUS OPENCOUNT CC (SETQ POS (IPLUS POS (SETQ LEN (NCHARS L P2FLAG))))))
                    (TERPRI FILE)                                   ; TAB wouldn't work, cause POSITION doesn't know where we
                                                                    ; are.
                    (RPTQ LMARG (PRIN3 '\  FILE))
                    (SETQ POS (IPLUS LMARG LEN))))
                (COND
                   ((IGREATERP OPENCOUNT 0)
                    (RPTQ OPENCOUNT (PRIN3 '\( FILE))
                    (SETQ POS (IPLUS POS OPENCOUNT))
                    (SETQ OPENCOUNT 0)))
                (COND
                   (P2FLAG (PRIN4 L FILE))
                   (T (PRIN3 L FILE)))))
          (COND
             ((AND (IGREATERP RMARG (ADD1 POS))
                   (LISTP (CDR LIST)))
              (PRIN3 '\  FILE)
              (SETQ POS (ADD1 POS))))
       $$ITERATE
          (SETQ LIST (CDR LIST))
          (GO $$LP)
       $$OUT
          (RPTQ CC (COND
                      ((ILESSP RMARG (SETQ POS (ADD1 POS)))
                       (TERPRI FILE)                                ; We do the closes one-by-one, in case they won't fit on a line
                                                                    ; with only 1 atom
                       (RPTQ LMARG (PRIN3 '\  FILE))
                       (PRIN3 '\) FILE)
                       (SETQ POS (ADD1 LMARG)))
                      (T (PRIN3 '\) FILE))))
          (RETURN $$VAL))
     POS))

)

;; SUBLIS and friends

(DEFINEQ

(SUBLIS
  (LAMBDA (ALST EXPR FLG)
    (COND
       ((LISTP EXPR)
        ((LAMBDA (D A)
           (COND
              ((OR (NEQ A (CAR EXPR))
                   (NEQ D (CDR EXPR))
                   FLG)
               (CONS A D))
              (T EXPR)))
         (AND (CDR EXPR)
              (SUBLIS ALST (CDR EXPR)
                      FLG))
         (SUBLIS ALST (CAR EXPR)
                 FLG)))
       (T (LET ((Y (FASSOC EXPR ALST)))
               (COND
                  (Y (COND
                        (FLG (COPY (CDR Y)))
                        (T (CDR Y))))
                  (T EXPR)))))))


(SUBPAIR
  (LAMBDA (OLD NEW EXPR FLG)                                        (* |lmm| "25-FEB-82 15:29")
    (COND
       ((LISTP EXPR)
        ((LAMBDA (D A)
           (COND
              ((OR (NEQ A (CAR EXPR))
                   (NEQ D (CDR EXPR))
                   FLG)
               (CONS A D))
              (T EXPR)))
         (AND (CDR EXPR)
              (SUBPAIR OLD NEW (CDR EXPR)
                       FLG))
         (SUBPAIR OLD NEW (CAR EXPR)
                  FLG)))
```

```
       (T (PROG NIL
            LP  (RETURN (COND
                          ((NULL OLD)
                           EXPR)
                          ((NLISTP OLD)
                           (COND
                              ((EQ EXPR OLD)
                               (COND
                                  (FLG (COPY NEW))
                                  (T NEW)))
                              (T EXPR)))
                          ((EQ EXPR (CAR OLD))
                           (COND
                              (FLG (COPY (CAR NEW)))
                              (T (CAR NEW))))
                          (T (SETQ OLD (CDR OLD))
                             (SETQ NEW (CDR NEW))
                             (GO LP)))))))))
```

(**DSUBLIS**

```
  (LAMBDA (ALST EXPR FLG)
    (COND
       ((NLISTP EXPR)
        (SUBLIS ALST EXPR FLG))
       (T (LET ((A (DSUBLIS ALST (CAR EXPR)
                            FLG)))
             (OR (EQ A (CAR EXPR))
                 (RPLACA EXPR A)))
          (LET ((D (DSUBLIS ALST (CDR EXPR)
                            FLG)))
             (OR (EQ D (CDR EXPR))
                 (RPLACD EXPR D)))
          EXPR))))
```

)

(* * CONSTANTS)

```
(DEFINEQ
```

(**CONSTANTOK**

```
  (LAMBDA (X DEPTH)                                    (* |lmm| " 1-OCT-78 22:03")
    (OR DEPTH (SETQ DEPTH 100))
    (COND
       ((OR (SMALLP X)
            (STRINGP X)
            (FLOATP X))
        DEPTH)
       ((FIXP X)
        (AND (NOT (SMALLP (IPLUS X)))
             DEPTH))
       ((LITATOM X)
        (AND (IGREATERP (NCHARS X)
                        0)
             DEPTH))
       ((LISTP X)
        (AND (SETQ DEPTH (CONSTANTOK (CAR X)
                                     (SUB1 DEPTH)))
             (CONSTANTOK (CDR X)
                         DEPTH))))))
```

)

```
(MOVD? 'EVQ 'CONSTANT)

(MOVD? 'EVQ 'DEFERREDCONSTANT)

(MOVD? 'EVQ 'LOADTIMECONSTANT)
```

(* * SCRATCHLIST)

```
(PUTPROPS SCRATCHLIST MACRO ((SCRATCHLIST . FORMS)
                             ((LAMBDA (!SCRATCHLIST !SCRATCHTAIL)
                                (DECLARE (SPECVARS !SCRATCHLIST !SCRATCHTAIL))
                                (SETQ !SCRATCHTAIL !SCRATCHLIST)
                                (PROGN . FORMS)
                                (COND
                                   ((EQ !SCRATCHTAIL !SCRATCHLIST)
                                    NIL)
                                   (T (PROG ((L2 (CDR !SCRATCHLIST)))
                                            (RPLACD !SCRATCHLIST (PROG1 (CDR !SCRATCHTAIL)
                                                                        (RPLACD !SCRATCHTAIL NIL)))
                                            (FRPLACD (FLAST !SCRATCHLIST)
                                                   L2)
```

```
                                        (RETURN L2)))))
                                 (OR (LISTP SCRATCHLIST)
                                     (CONS))
                                 NIL)))

(PUTPROPS ADDTOSCRATCHLIST MACRO ((VALUE)
                                  (FRPLACA (SETQ !SCRATCHTAIL (OR (LISTP (CDR !SCRATCHTAIL))
                                                                  (CDR (FRPLACD !SCRATCHTAIL (CONS)))))
                                           VALUE)))

(PUTPROPS SCRATCHLIST INFO EVAL)

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS SYSFILES LOADOPTIONS LISPXCOMS CLISPTRANFLG COMMENTFLG HISTSTR4 LISPXREADFN REREADFLG HISTSTR0
       CTRLUFLG NOLINKMESS PROMPTCHARFORMS PROMPT#FLG FILERDTBL SPELLINGS2 USERWORDS BELLS CLISPARRAY)
)

(DEFINEQ
```

(NLAMBDA.ARGS

```
  (LAMBDA (X)                                                    (* |bvm:| "26-Apr-86 16:41")
```

;;; Standard function to take argument to NLAMBDA function, e.g. BREAK, and check to see if accidentally quoted.

;;; Handles both BREAK 'FOO as a command and (BREAK 'FOO 'BAR). In the former case, X is (QUOTE FOO), in the latter it is ((QUOTE FOO) ;;; (QUOTE BAR)).

```
    (COND
       ((NLISTP X)
        (AND X (LIST X)))
       ((AND (EQ (CAR X)
                 'QUOTE)
             (LISTP (CDR X))))
       ((AND (LISTP (CAR X))
             (EQ (CAAR X)
                 'QUOTE))
        (CONS (CADR (CAR X))
              (NLAMBDA.ARGS (CDR X))))
       (T X))))

)

(DECLARE\: DONTEVAL@LOAD DOCOPY

(ADDTOVAR CLISPARRAY )

(ADDTOVAR CLISPFLG )

(ADDTOVAR CTRLUFLG )

(ADDTOVAR EDITCALLS )

(ADDTOVAR EDITHISTORY )

(ADDTOVAR EDITUNDOSAVES )

(ADDTOVAR EDITUNDOSTATS )

(ADDTOVAR GLOBALVARS )

(ADDTOVAR LCASEFLG )

(ADDTOVAR LISPXBUFS )

(ADDTOVAR LISPXCOMS )

(ADDTOVAR LISPXFNS )

(ADDTOVAR LISPXHIST )

(ADDTOVAR LISPXHISTORY )

(ADDTOVAR LISPXPRINTFLG )

(ADDTOVAR NOCLEARSTKLST )

(ADDTOVAR NOFIXFNSLST )

(ADDTOVAR NOFIXVARSLST )

(ADDTOVAR P.A.STATS )

(ADDTOVAR PROMPTCHARFORMS )

(ADDTOVAR READBUF )
```

```
(ADDTOVAR READBUFSOURCE )

(ADDTOVAR REREADFLG )

(ADDTOVAR RESETSTATE )

(ADDTOVAR SPELLSTATS1 )

(RPAQQ CHCONLST (NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL))

(RPAQQ CHCONLST1 (NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL))

(RPAQQ CHCONLST2 (NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL))

(RPAQQ CLEARSTKLST T)

(RPAQQ CLISPTRANFLG CLISP\ )

(RPAQ HISTSTR0 "<c.r.>")

(RPAQ HISTSTR2 "repeat")

(RPAQ HISTSTR3 "from event:")

(RPAQ HISTSTR4 "ignore")

(RPAQQ LISPXREADFN READ)

(RPAQQ USEMAPFLG T)

(MAPC '((APPLY BLKAPPLY)
        (SETTOPVAL SETATOMVAL)
        (GETTOPVAL GETATOMVAL)
        (APPLY* BLKAPPLY*)
        (RPLACA FRPLACA)
        (RPLACD FRPLACD)
        (STKNTH FSTKNTH)
        (STKNAME FSTKNAME)
        (CHARACTER FCHARACTER)
        (STKARG FSTKARG)
        (CHCON DCHCON)
        (UNPACK DUNPACK)
        (ADDPROP /ADDPROP)
        (ATTACH /ATTACH)
        (DREMOVE /DREMOVE)
        (DSUBST /DSUBST)
        (NCONC /NCONC)
        (NCONC1 /NCONC1)
        (PUT /PUT)
        (PUTPROP /PUTPROP)
        (PUTD /PUTD)
        (REMPROP /REMPROP)
        (RPLACA /RPLACA)
        (RPLACD /RPLACD)
        (SET /SET)
        (SETATOMVAL /SETATOMVAL)
        (SETTOPVAL /SETTOPVAL)
        (SETPROPLIST /SETPROPLIST)
        (SET SAVESET)
        (PRINT LISPXPRINT)
        (PRIN1 LISPXPRIN1)
        (PRIN2 LISPXPRIN2)
        (SPACES LISPXSPACES)
        (TAB LISPXTAB)
        (TERPRI LISPXTERPRI)
        (PRINT SHOWPRINT)
        (PRIN2 SHOWPRIN2)
        (PUTHASH /PUTHASH)
        '*
        (FNCLOSER /FNCLOSER)
        (FNCLOSERA /FNCLOSERA)
        (FNCLOSERD /FNCLOSERD)
        (EVQ DELFILE)
        (NILL SMASHFILECOMS)
        (PUTASSOC /PUTASSOC)
        (LISTPUT1 PUTL)
        (NILL I.S.OPR)
        (NILL RESETUNDO)
        (NILL LISPXWATCH)
        'ADDSTATS
        (NILL FREEVARS)
        'USEDFREE
        (COPYBYTES COPYCHARS))
      (FUNCTION (LAMBDA (X)
                  (MOVD? (CAR X)
                         (CADR X)))))
```

```
(MAPC '((TIME PRIN1 LISPXPRIN1)
        (TIME SPACES LISPXSPACES)
        (TIME PRINT LISPXPRINT)
        (DEFC PRINT LISPXPRINT)
        (DEFC PUTD /PUTD)
        (DEFC PUTPROP /PUTPROP)
        (DOLINK FNCLOSERD /FNCLOSERD)
        (DOLINK FNCLOSERA /FNCLOSERA)
        (DEFLIST PUTPROP /PUTPROP)
        (SAVEDEF1 PUTPROP /PUTPROP)
        (MKSWAPBLOCK PUTD /PUTD))
      (FUNCTION (LAMBDA (X)
                  (AND (CCODEP (CAR X))
                       (APPLY 'CHANGENAME X)))))

(MAPC '((EVALQT (LAMBDA NIL
                  (PROG (TEM)
                        (RESETRESTORE NIL 'RESET)
                    LP  (PROMPTCHAR '_ T)
                        (LISPX (LISPXREAD T T))
                        (GO LP))))
        (LISPX (LAMBDA (LISPXX)
                 (PRINT (AND LISPXX (PROG (LISPXLINE LISPXHIST TEM)
                                          (RETURN (COND
                                                    ((AND (NLISTP LISPXX)
                                                          (SETQ LISPXLINE (READLINE T NIL T)))
                                                     (APPLY LISPXX (CAR LISPXLINE)))
                                                    (T (EVAL LISPXX))))))
                        T T)))
        (LISPXREAD (LAMBDA (FILE RDTBL)
                     (COND
                        (READBUF (PROG1 (CAR READBUF)
                                        (SETQ READBUF (CDR READBUF))))
                        (T (READ FILE RDTBL)))))
        (LISPXREADP (LAMBDA (FLG)
                      (COND
                         ((AND READBUF (SETQ READBUF (LISPXREADBUF READBUF)))
                          T)
                         (T (READP T FLG)))))
        (LISPXUNREAD (LAMBDA (LST)
                       (SETQ READBUF (APPEND LST (CONS HISTSTR0 READBUF)))))
        (LISPXREADBUF (LAMBDA (RDBUF)
                        (PROG NIL
                          LP  (COND
                                 ((NLISTP RDBUF)
                                  (RETURN NIL))
                                 ((EQ (CAR RDBUF)
                                      HISTSTR0)
                                  (SETQ RDBUF (CDR RDBUF))
                                  (GO LP))
                                 (T (RETURN RDBUF))))))
        (LISPX/ (LAMBDA (X)
                  X))
        (LOWERCASE (LAMBDA (FLG)
                     (PROG1 LCASEFLG
                         (RAISE (NULL FLG))
                         (RPAQ LCASEFLG FLG))))
        (FILEPOS (LAMBDA (STR FILE)
                   (PROG NIL
                     LP  (COND
                            ((EQ (PEEKC FILE)
                                 (NTHCHAR STR 1))
                             (RETURN T)))
                         (READC FILE)
                         (GO LP))))
        (FILEPKGCOM (NLAMBDA NIL NIL)))
      (FUNCTION (LAMBDA (L)
                  (OR (GETD (CAR L))
                      (PUTD (CAR L)
                            (CADR L))))))
)

(DECLARE\: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS

(ADDTOVAR NLAMA RESETBUFS DMPHASH FILESLOAD)

(ADDTOVAR NLAML FILEMAP)

(ADDTOVAR LAMA READFILE NLIST)
)

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(LOCALVARS . T)
)

(RPAQQ MACHINEINDEPENDENTCOMS
```

```
((COMS                                                  ; "File loader"
      (FNS LOAD? FILESLOAD DOFILESLOAD FINDFILE-WITH-EXTENSIONS)
      (INITVARS (*COMPILED-EXTENSIONS* (LIST FASL.EXT COMPILE.EXT))))
 (COMS                                                  ; random machine-independent utilities
      (FNS DMPHASH HASHOVERFLOW)
      (DECLARE\: EVAL@COMPILE DONTCOPY (MACROS HASHOVERFLOW.ARRAYTEST HASHOVERFLOW.UPDATEARRAY))
      (FNS BKBUFS CHANGENAME CHNGNM CLBUFS COMSNAME DEFINE FNS.PUTDEF EQMEMB EQUALN FNCHECK FNTYP1
           LCSKIP MAPRINT MKLIST NAMEFIELD NAMEFIELD-STRING NLIST PRINTBELLS PROMPTCHAR RAISEP READFILE
           READLINE REMPROPLIST RESETBUFS TAB UNSAVED1 WRITEFILE CLOSE-AND-MAYBE-DELETE UNSAFE.TO.MODIFY
           )
      (VARS UNSAFE.TO.MODIFY.FNS)
      (COMS                                             ; FILEDATE, for finding out the creation date of source files, from
                                                        ; the compiled files.
            ;; FASL isn't loaded when MACHINEINDEPENDENT is, so we have to fake the FASL checker for now.  It's defined in
            ;; FASLOAD.
            (FNS FILEDATE)
            (P (MOVD? 'NILL 'FASL-FILEDATE)))
      (P (MOVD? 'CL:FMAKUNBOUND 'UNDOABLY-FMAKUNBOUND))
                                                        ; used in FNS.PUTDEF before CMLUNDO loaded
      )
 (COMS                                                  ; Functions for retrieving and remembering FILEMAPs and file
                                                        ; reader environments
      (FNS FILEMAP \\PARSE-FILE-HEADER GET-ENVIRONMENT-AND-FILEMAP LOOKUP-ENVIRONMENT-AND-FILEMAP
           GET-FILEMAP-FROM-FILECREATED \\FILEMAP-HASHOVERFLOW FLUSHFILEMAPS LISPSOURCEFILEP GETFILEMAP
           PUTFILEMAP UPDATEFILEMAP PRINT-READER-ENVIRONMENT)
      (INITVARS (*FILEMAP-LIMIT* 20)
             (*FILEMAP-VERSIONS* 2)
             (*FILEMAP-HASH* (HASHARRAY *FILEMAP-LIMIT* (FUNCTION \\FILEMAP-HASHOVERFLOW)
                                (FUNCTION STRING-EQUAL-HASHBITS)
                                (FUNCTION STRING.EQUAL))))
      (DECLARE\: EVAL@COMPILE DONTCOPY (RECORDS FILEMAPHASH)
             (GLOBALVARS *FILEMAP-LIMIT* *FILEMAP-VERSIONS* *FILEMAP-HASH*)))
 (COMS (* * LVLPRINT)
      (FNS LVLPRINT LVLPRIN1 LVLPRIN2 LVLPRIN LVLPRIN0))
 (COMS                                                  ; used by PRINTOUT
      (FNS FLUSHRIGHT PRINTPARA PRINTPARA1))
 (COMS                                                  ; SUBLIS and friends
      (FNS SUBLIS SUBPAIR DSUBLIS))
 (COMS (* * CONSTANTS)
      (FNS CONSTANTOK)
      (P (MOVD? 'EVQ 'CONSTANT)
         (MOVD? 'EVQ 'DEFERREDCONSTANT)
         (MOVD? 'EVQ 'LOADTIMECONSTANT)))
 (COMS (* * SCRATCHLIST)
      (PROP MACRO SCRATCHLIST ADDTOSCRATCHLIST)
      (PROP INFO SCRATCHLIST))
 (GLOBALVARS SYSFILES LOADOPTIONS LISPXCOMS CLISPTRANFLG COMMENTFLG HISTSTR4 LISPXREADFN REREADFLG
       HISTSTR0 CTRLUFLG NOLINKMESS PROMPTCHARFORMS PROMPT#FLG FILERDTBL SPELLINGS2 USERWORDS BELLS
       CLISPARRAY)
 (FNS NLAMBDA.ARGS)
 (DECLARE\: DONTEVAL@LOAD DOCOPY                        ; initialization of variables used in many places
       (ADDVARS (CLISPARRAY)
              (CLISPFLG)
              (CTRLUFLG)
              (EDITCALLS)
              (EDITHISTORY)
              (EDITUNDOSAVES)
              (EDITUNDOSTATS)
              (GLOBALVARS)
              (LCASEFLG)
              (LISPXBUFS)
              (LISPXCOMS)
              (LISPXFNS)
              (LISPXHIST)
              (LISPXHISTORY)
              (LISPXPRINTFLG)
              (NOCLEARSTKLST)
              (NOFIXFNSLST)
              (NOFIXVARSLST)
              (P.A.STATS)
              (PROMPTCHARFORMS)
              (READBUF)
              (READBUFSOURCE)
              (REREADFLG)
              (RESETSTATE)
              (SPELLSTATS1))
        (VARS (CHCONLST '(NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL
                              ))
              (CHCONLST1 '(NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL
                               NIL))
              (CHCONLST2 '(NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL
                               NIL))
              (CLEARSTKLST T)
              (CLISPTRANFLG 'CLISP\ )
              (HISTSTR0 "<c.r.>")
              (HISTSTR2 "repeat")
```

```
                    (HISTSTR3 "from event:")
                    (HISTSTR4 "ignore")
                    (LISPXREADFN 'READ)
                    (USEMAPFLG T))
                (P (MAPC '((APPLY BLKAPPLY)
                           (SETTOPVAL SETATOMVAL)
                           (GETTOPVAL GETATOMVAL)
                           (APPLY* BLKAPPLY*)
                           (RPLACA FRPLACA)
                           (RPLACD FRPLACD)
                           (STKNTH FSTKNTH)
                           (STKNAME FSTKNAME)
                           (CHARACTER FCHARACTER)
                           (STKARG FSTKARG)
                           (CHCON DCHCON)
                           (UNPACK DUNPACK)
                           (ADDPROP /ADDPROP)
                           (ATTACH /ATTACH)
                           (DREMOVE /DREMOVE)
                           (DSUBST /DSUBST)
                           (NCONC /NCONC)
                           (NCONC1 /NCONC1)
                           (PUT /PUT)
                           (PUTPROP /PUTPROP)
                           (PUTD /PUTD)
                           (REMPROP /REMPROP)
                           (RPLACA /RPLACA)
                           (RPLACD /RPLACD)
                           (SET /SET)
                           (SETATOMVAL /SETATOMVAL)
                           (SETTOPVAL /SETTOPVAL)
                           (SETPROPLIST /SETPROPLIST)
                           (SET SAVESET)
                           (PRINT LISPXPRINT)
                           (PRIN1 LISPXPRIN1)
                           (PRIN2 LISPXPRIN2)
                           (SPACES LISPXSPACES)
                           (TAB LISPXTAB)
                           (TERPRI LISPXTERPRI)
                           (PRINT SHOWPRINT)
                           (PRIN2 SHOWPRIN2)
                           (PUTHASH /PUTHASH)
                           '*
                           (FNCLOSER /FNCLOSER)
                           (FNCLOSERA /FNCLOSERA)
                           (FNCLOSERD /FNCLOSERD)
                           (EVQ DELFILE)
                           (NILL SMASHFILECOMS)
                           (PUTASSOC /PUTASSOC)
                           (LISTPUT1 PUTL)
                           (NILL I.S.OPR)
                           (NILL RESETUNDO)
                           (NILL LISPXWATCH)
                           'ADDSTATS
                           (NILL FREEVARS)
                           'USEDFREE
                           (COPYBYTES COPYCHARS))
                       (FUNCTION (LAMBDA (X)
                                         (MOVD? (CAR X)
                                                (CADR X)))))
                   (MAPC '((TIME PRIN1 LISPXPRIN1)
                           (TIME SPACES LISPXSPACES)
                           (TIME PRINT LISPXPRINT)
                           (DEFC PRINT LISPXPRINT)
                           (DEFC PUTD /PUTD)
                           (DEFC PUTPROP /PUTPROP)
                           (DOLINK FNCLOSERD /FNCLOSERD)
                           (DOLINK FNCLOSERA /FNCLOSERA)
                           (DEFLIST PUTPROP /PUTPROP)
                           (SAVEDEF1 PUTPROP /PUTPROP)
                           (MKSWAPBLOCK PUTD /PUTD))
                       (FUNCTION (LAMBDA (X)
                                         (AND (CCODEP (CAR X))
                                              (APPLY 'CHANGENAME X)))))
                   (MAPC '((EVALQT (LAMBDA NIL (PROG (TEM)
                                                     (RESETRESTORE NIL 'RESET)
                                                     LP
                                                     (PROMPTCHAR '_ T)
                                                     (LISPX (LISPXREAD T T))
                                                     (GO LP))))
                           (LISPX (LAMBDA (LISPXX)
                                          (PRINT (AND LISPXX (PROG (LISPXLINE LISPXHIST TEM)
                                                                   (RETURN (COND ((AND (NLISTP LISPXX)
                                                                                       (SETQ LISPXLINE
                                                                                             (READLINE T NIL T)))
                                                                                  (APPLY LISPXX (CAR LISPXLINE)))
                                                                                 (T (EVAL LISPXX))))))
```

```
                                        T T)))
                            (LISPXREAD (LAMBDA (FILE RDTBL)
                                               (COND (READBUF (PROG1 (CAR READBUF)
                                                                     (SETQ READBUF (CDR READBUF))))
                                                     (T (READ FILE RDTBL)))))
                            (LISPXREADP (LAMBDA (FLG)
                                                (COND ((AND READBUF (SETQ READBUF (LISPXREADBUF READBUF)))
                                                       T)
                                                      (T (READP T FLG)))))
                            (LISPXUNREAD (LAMBDA (LST)
                                                 (SETQ READBUF (APPEND LST (CONS HISTSTR0 READBUF)))))
                            (LISPXREADBUF (LAMBDA (RDBUF)
                                                  (PROG NIL LP (COND ((NLISTP RDBUF)
                                                                      (RETURN NIL))
                                                                     ((EQ (CAR RDBUF)
                                                                          HISTSTR0)
                                                                      (SETQ RDBUF (CDR RDBUF))
                                                                      (GO LP))
                                                                     (T (RETURN RDBUF))))))
                            (LISPX/ (LAMBDA (X)
                                            X))
                            (LOWERCASE (LAMBDA (FLG)
                                               (PROG1 LCASEFLG
                                                      (RAISE (NULL FLG))
                                                      (RPAQ LCASEFLG FLG))))
                            (FILEPOS (LAMBDA (STR FILE)
                                             (PROG NIL LP (COND ((EQ (PEEKC FILE)
                                                                     (NTHCHAR STR 1))
                                                                 (RETURN T)))
                                                   (READC FILE)
                                                   (GO LP))))
                            (FILEPKGCOM (NLAMBDA NIL NIL)))
                           (FUNCTION (LAMBDA (L)
                                             (OR (GETD (CAR L))
                                                 (PUTD (CAR L)
                                                       (CADR L))))))))
        (DECLARE\: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA FILEMAP RESETBUFS DMPHASH
                                                                                FILESLOAD)
                                                                              (NLAML)
                                                                              (LAMA READFILE NLIST)))
        (LOCALVARS . T)))

(DECLARE\: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS

(ADDTOVAR NLAMA FILEMAP RESETBUFS DMPHASH FILESLOAD)

(ADDTOVAR NLAML )

(ADDTOVAR LAMA READFILE NLIST)
)

(PUTPROPS MACHINEINDEPENDENT COPYRIGHT ("Venue & Xerox Corporation" T 1983 1984 1985 1986 1987 1988 1989 1990
                                        1991 1992))
```

FUNCTION INDEX

VARIABLE INDEX

MACRO INDEX

PROPERTY INDEX

RECORD INDEX