File created: 12-Mar-2021 11:17:48 **{DSK}<home>larry>ilisp>med>sources>ACODE.;6**

changes to: (FNS PRINTCODENT)

previous date: 12-Mar-2021 09:50:45 {DSK}<home>larry>ilisp>med>sources>ACODE.;4

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
;;
;; Copyright (c) 1982-1988, 1990-1992, 1995, 2017, 2021 by Venue & Xerox Corporation.

(RPAQQ ACODECOMS
       ((COMS                                                    ; Printing compiled code
              (FNS PRINTCODE PRINTCODENT)
              (DECLARE%: EVAL@COMPILE DONTCOPY (MACROS PCVAR PRINJUMP NEXTBYTE PRINTCODEHEADERDECODE)
                     (GLOBALVARS \INITSUBRS \PRINTCODE.LEVEL \PRINTCODE.STKSTATE)))
        (COMS                                                    ; Analyzing compiled code
              (FNS CALLSCCODE RUNION)
              (FNS CHANGECCODE CCCSUBFN? \SUBFNDEF CCCSCAN \CODEBLOCKP)
              (FNS \MAP-CODE-POINTERS \MAP-CODE-LITERALS)
              (BLOCKS (CALLSCCODE CALLSCCODE RUNION)
                     (CHANGECCODE CHANGECCODE CCCSUBFN? CCCSCAN))
              ;; MACROS/OPTIMIZERS for getting and setting symbol entries in a compiled-code block.  These are parameterized to allow for 2-,
              ;; 3-, and 4-byte symbol representations.
              (DECLARE%: EVAL@COMPILE DONTCOPY (RECORDS REFMAP)
                     (MACROS CODEBASELT CODEBASELT2 CODEBASESETA CODEBASESETA2 CODEBASELT3 CODEBASELT4
                            CODEBASESETA3 CODEBASESETA4)
                     (OPTIMIZERS CODEBASESETATOM CODEBASEGETATOM CODEBASEGETNAME BYTESPERCODEATOM BIG-VMEM-HOST)
                     (FILES (LOADCOMP)
                            LLGC LLCODE LLBASIC MODARITH RENAMEMACROS))
              (ADDVARS (IGNOREFNS)))
        (COMS                                                    ; Maintaining ref count consistency in code
              (FNS \COPYCODEBLOCK \COPYFNHEADER \RECLAIMCODEBLOCK))
        (COMS                                                    ; Low-level break
              (FNS LLBREAK BROKENDEF))
        [COMS                                                    ; for TELERAID
              (DECLARE%: DONTCOPY (ADDVARS (RDCOMS (FNS PRINTCODE PRINTCODENT BROKENDEF))
                                           (EXPANDMACROFNS NEXTBYTE PCVAR PRINJUMP CODEBASELT CODEBASELT2
                                                  CODEBASESETA CODEBASESETA2 PRINTCODEHEADERDECODE]
        (COMS                                                    ; reference to opcodes symbolically
              (FNS PRINTOPCODES)
              (GLOBALVARS \OPCODES))
        (DECLARE%: EVAL@COMPILE DONTCOPY (LOCALVARS . T))))

;; Printing compiled code

(DEFINEQ

(PRINTCODE
  [LAMBDA (FN LVFLG RADIX OUTF FIRSTBYTE PC FN.IS.CODEBASE)        ; Edited 12-Mar-2021 09:48 by larry
                                                                   ; Edited 25-Feb-91 15:46
                                                                   ; by sybalsky

;;; WARNING: this code must run 'renamed' for TeleRaid Printcode to work.  However, it is pretty tricky to get it to run renamed because some of the
;;; constructs run in local space (e.g., the CARs and CADRs of the code list) and many run in remote space (e.g., the bytes of the code).

;;; It seems that frequently when modifying any part of PRINTCODE the renamed version stops working, so *BEWARE* and make sure you test any
;;; edits by doing a (DORENAME 'R) and checking TeleRaid's CodePrint command, as well as in normal PRINTCODE mode.

;;; All the CODEARRAY accesses are equivalent to FNHEADER accesses indirected thru the CCODEP object.  The reason it is done this awful crufty
;;; way, instead of fetching the code base, is so this works in Interlisp-10 as well.  Might want to punt that now.

    (DECLARE (SPECVARS OUTF))
    (OR RADIX (SETQ RADIX 16))
    (LET ([CODEBASE (COND
                       (FN.IS.CODEBASE FN)
                       (T (OR (\GET-COMPILED-CODE-BASE FN)
                              [AND (LITATOM FN)
                                   (\GET-COMPILED-CODE-BASE (GET FN 'CODE]
                              (ERROR FN "not compiled code"]
          (I4 (NUMFORMATCODE (LIST 'FIX (if (IGREATERP RADIX 15)
                                            then 3
                                          else 4)
                                   RADIX)))
          (I6 (NUMFORMATCODE (LIST 'FIX (if (IGREATERP RADIX 15)
                                            then 5
                                          else 6)
                                   RADIX)))
          NTSIZE STARTPC TAG TEMP OP# PVARS FVARS IVARS)
         (DECLARE (SPECVARS CODEBASE IVARS PVARS FVARS I4 I6))     ; Used by PRINTCODENT
         (LET ((*PRINT-BASE* RADIX))
```

```
          (for I from 0 by BYTESPERWORD while (ILESSP I (UNFOLD (fetch (FNHEADER OVERHEADWORDS) of T)
                                                                BYTESPERWORD))
             do (PRINTNUM I4 I OUTF)
                (PRIN1 ": " OUTF)
                (PRINTNUM I6 (CODEBASELT2 CODEBASE I)
                      OUTF)
                (PRINTCODEHEADERDECODE CODEBASE I OUTF)      ; Interpret header word
                (TERPRI OUTF)))
       (SETQ NTSIZE (fetch (FNHEADER NTSIZE) of CODEBASE))
       (PRINTCODENT "name table: " (UNFOLD (fetch (FNHEADER OVERHEADWORDS) of T)
                                           BYTESPERWORD)
             (UNFOLD NTSIZE BYTESPERWORD))
       (SETQ STARTPC (fetch (FNHEADER STARTPC) of CODEBASE))
       (COND
          ((GREATERP [SETQ NTSIZE (IDIFFERENCE (COND
                                                  ((fetch (FNHEADER NATIVE)
                                                          CODEBASE)
                                                   ;; native code has an extra 4 bytes
                                                   (- STARTPC 4))
                                                  (T STARTPC))
                                                (SETQ TEMP (IPLUS (UNFOLD (fetch (FNHEADER OVERHEADWORDS) of T)
                                                                          BYTESPERWORD)
                                                                  (COND
                                                                     ((EQ NTSIZE 0)
                                                                     ; No nametable, but there's a quad of zeros there anyway
                                                                      BYTESPERQUAD)
                                                                     (T (UNFOLD NTSIZE (ITIMES 2 BYTESPERWORD]
                      BYTESPERCELL)
           (PRINTCODENT "Local args: " TEMP (FOLDLO NTSIZE 2)))
          ((EQ NTSIZE BYTESPERCELL)                          ; Debugging info
           (printout OUTF T "Info: " .P2 (\GETBASEPTR CODEBASE (FOLDLO TEMP BYTESPERWORD))
                  T)))
       (printout OUTF T "----" T)
       (PROG ((CODELOC STARTPC)
              (LEVEL (AND LVFLG 0))
              B B1 B2 B3 B4 B5 FN LEN LEVADJ STK)
             [ALLOCAL (COND
                         (LEVEL (SETUPHASHARRAY '\PRINTCODE.LEVEL)
                                (SETUPHASHARRAY '\PRINTCODE.STKSTATE)
                                (CLRHASH \PRINTCODE.LEVEL)
                                (CLRHASH \PRINTCODE.STKSTATE]
         LP  (COND
                ((AND PC (IGEQ CODELOC PC))                  ; Caller asked to highlight this spot
                 (COND
                    ((NOT (IEQP CODELOC PC))
                     (PRINTOUT OUTF "(PC ")
                     (PRINTNUM I4 PC OUTF)
                     (PRINTOUT OUTF " not found)")))
                 (printout OUTF "------------------------------" T)
                 (SETQ PC)))
             (COND
                ((OR (NULL FIRSTBYTE)
                     (IGEQ CODELOC FIRSTBYTE))
                 (PRINTNUM I4 CODELOC OUTF)
                 (PRIN1 ": " OUTF)
                 [COND
                    (LVFLG (SETQ TEMP (GETHASH CODELOC \PRINTCODE.LEVEL))
                           [COND
                              [LEVEL (COND
                                        ([AND TEMP (OR (NEQ LEVEL TEMP)
                                                       (NOT (EQUAL STK (GETHASH CODELOC \PRINTCODE.STKSTATE]
                                         (PRIN1 "*" OUTF]
                              (T (SETQ LEVEL TEMP)
                                 (SETQ STK (GETHASH CODELOC \PRINTCODE.STKSTATE]
                           (COND
                              (LEVEL (TAB 7 NIL OUTF)
                                     (PRINTNUM I4 LEVEL OUTF]
                 (TAB 12 NIL OUTF))
                (T                                           ; Don't print code, but quietly process LEVEL etc
                   (SETQ TAG (\FINDOP (NEXTBYTE)))
                   (SELECTQ (ALLOCAL (OR (fetch OPPRINT of TAG)
                                         (fetch OPCODENAME of TAG)))
                        (-X- (TERPRI OUTF)
                             (RETURN))
                        (BIND [ALLOCAL (COND
                                          (LEVEL (push STK (SETQ LEVEL (ADD1 (IDIFFERENCE LEVEL
                                                                                 (LOGAND (CODEBASELT
                                                                                             CODEBASE
                                                                                             CODELOC)
                                                                                         15])
                        (UNBIND [ALLOCAL (AND LEVEL (SETQ LEVEL (pop STK])
                        (DUNBIND [ALLOCAL (AND LEVEL (SETQ LEVEL (SUB1 (pop STK])
                        (RETURN (SETQ LEVEL))
                        (SUBRCALL [AND LEVEL (SETQ LEVEL (ADD1 (IDIFFERENCE LEVEL (CODEBASELT CODEBASE
                                                                                              (ADD1 CODELOC])
                        (MISCN [AND LEVEL (SETQ LEVEL (ADD1 (IDIFFERENCE LEVEL (CODEBASELT CODEBASE
```

```
                                                                                (IPLUS 2 CODELOC])
                NIL)
             [COND
                ([AND LEVEL (ALLOCAL (SETQ LEVADJ (fetch LEVADJ of TAG]
                 [ALLOCAL (COND
                             ((LISTP LEVADJ)
                              (SETQ LEVADJ (CAR LEVADJ]
                 (SELECTQ LEVADJ
                      (FNX (add LEVEL (IDIFFERENCE 1 (CODEBASELT CODEBASE CODELOC))))
                      (POP.N (SETQ LEVEL (IDIFFERENCE LEVEL (CODEBASELT CODEBASE CODELOC))))
                      ((JUMP UNWIND)
                           (SETQ LEVEL))
                      ((CJUMP NCJUMP)
                           (add LEVEL -1))
                      (COND
                         ((NUMBERP LEVADJ)
                          (add LEVEL LEVADJ]
             (ALLOCAL (add CODELOC (fetch OPNARGS of TAG)))
             (GO LP)))
          [SETQ LEN (LOCAL (fetch OPNARGS of (SETQ TAG (\FINDOP (SETQ B (NEXTBYTE]
          (PRINTNUM I4 B OUTF)
          (COND
             ((IGREATERP LEN 0)
              (PRINTNUM I4 (SETQ B1 (NEXTBYTE))
                     OUTF)))
          (COND
             ((IGREATERP LEN 1)
              (PRINTNUM I4 (SETQ B2 (NEXTBYTE))
                     OUTF)))
          (COND
             ((IGREATERP LEN 2)
              (PRINTNUM I4 (SETQ B3 (NEXTBYTE))
                     OUTF)))
          (COND
             ((IGREATERP LEN 3)
              (PRINTNUM I4 (SETQ B4 (NEXTBYTE))
                     OUTF)))
          (COND
             ((IGREATERP LEN 4)
              (PRINTNUM I4 (SETQ B5 (NEXTBYTE))
                     OUTF)))
          [ALLOCAL (PROGN (printout OUTF 30 (fetch OPCODENAME of TAG))
                          (SETQ OP# (fetch OP# of TAG))
                          (SETQ LEVADJ (fetch LEVADJ of TAG]
          [ALLOCAL (COND
                      ((LISTP OP#)
                       (SETQ OP# (CAR OP#]
          [SELECTQ [SETQ TAG (ALLOCAL (OR (fetch OPPRINT of TAG)
                                          (fetch OPCODENAME of TAG]
               (-X- (TERPRI OUTF)
                    (RETURN))
               (IVAR (TAB 40 NIL OUTF)
                     (PCVAR (SELECTQ LEN
                                 (0 (IDIFFERENCE B OP#))
                                 (LRSH B1 1))
                            IVARS
                            'ivar))
               (PVAR (TAB 40 NIL OUTF)
                     (PCVAR (SELECTQ LEN
                                 (0 (IDIFFERENCE B OP#))
                                 (LRSH B1 1))
                            PVARS
                            'pvar))
               (FVAR (TAB 40 NIL OUTF)
                     (PCVAR (SELECTQ LEN
                                 (0 (IDIFFERENCE B OP#))
                                 (LRSH B1 1))
                            FVARS
                            'fvar))
               (JUMP (PRINJUMP (IPLUS (IDIFFERENCE B OP#)
                                      2)))
               (SIC (printout OUTF 40 .P2 B1))
               (SNIC (printout OUTF 40 .P2 (IDIFFERENCE B1 256)))
               (SICX (printout OUTF 40 .P2 (IPLUS (LLSH B1 8)
                                                  B2)))
               (JUMPX (PRINJUMP (COND
                                   ((IGEQ B1 128)
                                    (IDIFFERENCE B1 256))
                                   (T B1))))
               (FN ;; it's a function.  Print the name.
                   (NEW-SYMBOL-CODE (SETQ B (IPLUS (LLSH (IPLUS (LLSH (IPLUS (LLSH B1 8)
                                                                              B2)
                                                                         8)
                                                                    B3)
                                                               8)
                                                          B4))
```

```
                              (SETQ B (IPLUS (LLSH B1 8)
                                             B2)))
                          (printout OUTF 40 .P2 (\INDEXATOMDEF B)))
                     (BIND (TAB 40 NIL OUTF)
                           [ALLOCAL (PROG ((NNILS (LRSH B1 4))
                                           (NVALS (LOGAND B1 15)))
                                          (for I from (ADD1 (IDIFFERENCE B2 (IPLUS NNILS NVALS)))
                                             to (IDIFFERENCE B2 NNILS) do (SPACES 1 OUTF)
                                                                          (PCVAR I PVARS 'pvar))
                                          (PRIN1 '; OUTF)
                                          (for I from (ADD1 (IDIFFERENCE B2 NNILS)) to B2
                                             do (SPACES 1 OUTF)
                                                (PCVAR I PVARS 'pvar))
                                          (COND
                                             (LEVEL (push STK (SETQ LEVEL (ADD1 (IDIFFERENCE LEVEL NVALS])
                     (JUMPXX [PRINJUMP (IPLUS (LLSH B1 8)
                                              B2
                                              (COND
                                                 ((IGREATERP B1 127)
                                                  -65536)
                                                 (T 0])
                     (ATOM [printout OUTF 40 .P2
                                   (\INDEXATOMPNAME (NEW-SYMBOL-CODE
                                                      (IPLUS (LLSH (IPLUS (LLSH (IPLUS (LLSH B1 8)
                                                                                       B2)
                                                                                8)
                                                                          B3)
                                                                    8)
                                                              B4)
                                                      (IPLUS (LLSH B1 8)
                                                             B2])
                     (GCONST [printout OUTF 40 .P2 (1ST (\VAG2 (IPLUS (LLSH B1 8)
                                                                      B2)
                                                               (IPLUS (LLSH B3 8)
                                                                      B4])
                     (FNX [printout OUTF "(" B1 ")" 40 .P2
                                   (\INDEXATOMDEF (NEW-SYMBOL-CODE (IPLUS (LLSH (IPLUS (LLSH (IPLUS (LLSH B2 8)
                                                                                                    B3)
                                                                                             8)
                                                                                       B4)
                                                                                 8)
                                                                           B5)
                                                                     (IPLUS (LLSH B2 8)
                                                                            B3])
                     (TYPEP (printout OUTF "(" .P2 (OR (\TYPENAMEFROMNUMBER B1)
                                                       '?)
                                    ")"))
                     (UNBIND [ALLOCAL (AND LEVEL (SETQ LEVEL (pop STK])
                     (DUNBIND [ALLOCAL (AND LEVEL (SETQ LEVEL (SUB1 (pop STK])
                     (RETURN (SETQ LEVEL))
                     (SUBRCALL [ALLOCAL (printout OUTF 40 (for X in \INITSUBRS when (EQ B1 (CADR X))
                                                            do (RETURN (CAR X)) finally (RETURN "?"]
                               [AND LEVEL (SETQ LEVEL (ADD1 (IDIFFERENCE LEVEL B2])
                     (MISCN [ALLOCAL (printout OUTF 40 (for X in \USER-SUBR-LIST when (EQ B1 (CADR X))
                                                         do (RETURN (CAR X)) finally (RETURN "?"]
                            [AND LEVEL (SETQ LEVEL (ADD1 (IDIFFERENCE LEVEL B2])
                     (ALLOCAL (COND
                                 ((LISTP TAG)
                                  (printout OUTF 40 (CAR (NTH TAG (ADD1 B1]
                (TERPRI OUTF)
                [COND
                   ((AND LEVEL LEVADJ)
                    (SELECTQ LEVADJ
                         (FNX (add LEVEL (IDIFFERENCE 1 B1)))
                         (POP.N (SETQ LEVEL (IDIFFERENCE LEVEL B1)))
                         ((JUMP UNWIND)
                             (SETQ LEVEL))
                         ((CJUMP NCJUMP)
                              (add LEVEL -1))
                         (COND
                            ((NUMBERP LEVADJ)
                             (add LEVEL LEVADJ]
                (GO LP])
```

(**PRINTCODENT**

```
  [LAMBDA (STR START1 START2)
    (DECLARE (USEDFREE CODEBASE IVARS PVARS FVARS I4 I6 OUTF))  ; Edited 12-Mar-2021 11:17 by larry

    ;; Prints the name table identified with title STR that starts with names at START1 and codes at START2

    (LET (NAME TAG)
         (COND
            ((ILESSP START1 (SETQ START2 (IPLUS START2 START1)))
             (printout OUTF STR T)
             (for NT1 from START1 by (BYTESPERNAMEENTRY) while (ILESSP NT1 START2) as NT2 from START2
                by (BYTESPERNTOFFSETENTRY) do (PRINTNUM I4 NT1 OUTF)
                                              (PRIN1 ": " OUTF)
```

```
                                        (for I from 0 to (CONSTANT (SUB1 (BYTESPERNAMEENTRY)))
                                           do (PRINTNUM I4 (CODEBASELT CODEBASE (IPLUS NT1 I))
                                                     OUTF))
                                        (SPACES 2 OUTF)
                                        (PRINTNUM I4 NT2 OUTF)
                                        (PRIN1 ": " OUTF)
                                        (COND
                                           ((SETQ NAME (\INDEXATOMVAL (CODEBASEGETNAME CODEBASE NT1)))
                                            (SETQ TAG (GETNTOFFSET CODEBASE NT2))
                                            (printout OUTF .SP 1
                                                   (SELECTC (NTSLOT-VARTYPE (GETNTOFFSETENTRY CODEBASE NT2
                                                                                              ))
                                                        (IVARCODE (ALLOCAL (push IVARS (LIST TAG NAME)))
                                                                  'IVAR)
                                                        (PVARCODE (ALLOCAL (push PVARS (LIST TAG NAME)))
                                                                  'PVAR)
                                                        (PROGN (ALLOCAL (push FVARS (LIST TAG NAME)))
                                                               'FVAR))
                                                   " " TAG ": " .P2 NAME)))
                                        (TERPRI OUTF])

)

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(PUTPROPS PCVAR MACRO [(IND LST NAME)                          (* lmm "11-AUG-81 22:27")
                       (ALLOCAL (PROG NIL
                                      (PRIN2 [CADR (OR (ASSOC IND LST)
                                                       (RETURN (printout OUTF "[" NAME IND "]"]
                                             OUTF])

(PUTPROPS PRINJUMP MACRO [LAMBDA (N)
                          (PRIN1 "->" OUTF)
                          (PRINTNUM I4 [SETQ N (IPLUS N (IDIFFERENCE CODELOC (ADD1 LEN]
                                 OUTF)
                          (COND
                             (LEVEL (PUTHASH N (SELECTQ LEVADJ
                                                        ((NCJUMP JUMP)
                                                            LEVEL)
                                                        (SUB1 LEVEL))
                                                \PRINTCODE.LEVEL)
                                    (PUTHASH N STK \PRINTCODE.STKSTATE])

(PUTPROPS NEXTBYTE MACRO [NIL (CODEBASELT CODEBASE (PROG1 CODELOC (add CODELOC 1])

(PUTPROPS PRINTCODEHEADERDECODE DMACRO
          (DEFMACRO (CODEBASE INDEX OUTF) (LET
                                           (INDICES I THERE)
                                           [for NAME in (CDR (RECORDFIELDNAMES 'FNHEADER T))
                                              when (AND NAME (CL:SYMBOLP NAME))
                                              do [SETQ I (EVAL `(INDEXF (fetch (FNHEADER ,NAME]
                                                  (COND
                                                     ((EQ NAME '%#FRAMENAME)
                                                      (add I 1)))
                                                  (COND
                                                     ((SETQ THERE (ASSOC I INDICES))
                                                      (push (CDR THERE)
                                                            NAME))
                                                     (T (push INDICES (LIST I NAME]
                                           `(SELECTQ ,INDEX
                                                 (\,@ [for PAIR in INDICES
                                                         collect
                                                         (CONS
                                                          (UNFOLD (CAR PAIR)
                                                                  BYTESPERWORD)
                                                          (COND
                                                             [(CDDR PAIR)
                                                              (for NAME in (CDR PAIR)
                                                                 collect
                                                                 (SELECTQ NAME
                                                                      ((NATIVE CLOSUREP)
                                                                           `(AND (fetch (FNHEADER ,NAME)
                                                                                    of ,CODEBASE)
                                                                                 (PRIN1 ,(CONCAT "[" NAME "]")
                                                                                        ,OUTF)))
                                                                      `(printout
                                                                        ,OUTF
                                                                        ,(CONCAT " " (L-CASE (MKSTRING NAME))
                                                                                 ": ")
                                                                        (fetch (FNHEADER ,NAME)
                                                                           of ,CODEBASE]
                                                             [(EQ (CADR PAIR)
                                                                  '%#FRAMENAME)
                                                              `((printout ,OUTF " frame name: " .P2
                                                                          (1ST (fetch (FNHEADER %#FRAMENAME)
```

```
                                                                    of ,CODEBASE]
                                                              (T
                                                               `((PRIN1 ,[CONCAT " " (L-CASE (MKSTRING
                                                                                              (CADR PAIR]
                                                                        ,OUTF])
                                                        NIL))))
)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \INITSUBRS \PRINTCODE.LEVEL \PRINTCODE.STKSTATE)
)
)
```

;; Analyzing compiled code

```
(DEFINEQ
```

(**CALLSCCODE**

```
  [LAMBDA (DEF OPTION FNAPPLY)                                  (* DECLARATIONS%: (RECORD RESULT
                                                                (LNCALLED CALLED BOUND USEDFREE GLOBALS)))
                                                                ; Edited  1-Dec-92 00:51 by sybalsky:mv:envos
```

;;; Analyze DEF for function calls and variable references. Action depends on OPTION as follows:

;;; OPTION = NIL means return value of CALLSCCODE as described in IRM;

;;; OPTION = T means return list of free variable references;

;;; OPTION = APPLY, FNAPPLY, or VARAPPLY means call FNAPPLY on various references and return nothing. FNAPPLY takes two arguments: a
;;; symbol and a keyword indicating the type of reference, one of BOUND, USEDFREE, GLOBALS, or CALLED. If OPTION is FNAPPLY, only function
;;; references are noticed; if VARAPPLY, only variable bindings and references; otherwise all.

;;; For OPTION = NIL or T, CALLSCCODE descends into subfunctions.

```
    (PROG ((CODEBASE (OR (\GET-COMPILED-CODE-BASE DEF)
                         (\CODEBLOCKP DEF)
                         (ERROR DEF "not compiled code")))
           (IGNOREFNS IGNOREFNS)
           USEDFREE BOUND GLOBALS CALLED LNCALLED NTSIZE NAME TYPE TAG)
          (DECLARE (SPECVARS IGNOREFNS))
          [COND
             ((NEQ OPTION 'FNAPPLY)                             ; Get variables out of name table
              (SETQ NTSIZE (fetch (FNHEADER NTSIZE) of CODEBASE))
              (for NT1 from (UNFOLD (fetch (FNHEADER OVERHEADWORDS) of T)
                                    BYTESPERWORD)
                 by (BYTESPERNAMEENTRY) as NT2 from (IPLUS (CONSTANT (UNFOLD (fetch (FNHEADER OVERHEADWORDS)
                                                                                  of T)
                                                                              BYTESPERWORD))
                                                           (UNFOLD NTSIZE BYTESPERWORD))
                 by (BYTESPERNTOFFSETENTRY) until [NULL (SETQ NAME (\INDEXATOMVAL (GETNAMEENTRY CODEBASE NT1]
                 do (SETQ TYPE (SELECTQ (NTSLOT-VARTYPE (GETNTOFFSET CODEBASE NT2))
                                        ((IVARCODE PVARCODE)
                                             'BOUND)
                                        'USEDFREE))             ; Top two bits of the entry indicate kind of name:
                                                                ; 00(\NT.IVARCODE) = IVAR, 10(\NT.PVARCODE) = PVAR, 11
                                                                ; = FVAR
                    (SELECTQ OPTION
                         ((VARAPPLY APPLY)
                              (CL:FUNCALL FNAPPLY NAME TYPE))
                         (SELECTQ TYPE
                              (BOUND (pushnew BOUND NAME))
                              (pushnew USEDFREE NAME]
          (PROG ((CODELOC (fetch (FNHEADER STARTPC) of CODEBASE))
                 B B1 B2 B3 B4 B5 FN LEN)
            LP   (SETQ B (NEXTBYTE))
                 (SETQ B1 (AND [ILESSP 0 (SETQ LEN (fetch OPNARGS of (SETQ TAG (\FINDOP B]
                               (NEXTBYTE)))
                 (SETQ B2 (AND (ILESSP 1 LEN)
                               (NEXTBYTE)))
                 (SETQ B3 (AND (ILESSP 2 LEN)
                               (NEXTBYTE)))
                 (SETQ B4 (AND (ILESSP 3 LEN)
                               (NEXTBYTE)))
                 (SETQ B5 (AND (ILESSP 4 LEN)
                               (NEXTBYTE)))
                 (SELECTQ (fetch OPCODENAME of TAG)
                      (-X- (RETURN))
                      ((FN0 FN1 FN2 FN3 FN4)
                           [COND
                              [(FMEMB :4-BYTE COMPILER::*HOST-ARCHITECTURE*)
                               (SETQ NAME (\INDEXATOMDEF (IPLUS (LLSH (IPLUS (LLSH (IPLUS (LLSH B1 8)
                                                                                          B2)
                                                                                   8)
                                                                             B3)
                                                                       8)
                                                                 B4]
```

```
                        [(FMEMB :3-BYTE COMPILER::*HOST-ARCHITECTURE*)
                         (SETQ NAME (\INDEXATOMDEF (IPLUS (LLSH (IPLUS (LLSH B1 8)
                                                                       B2)
                                                                8)
                                                          B3]
                        (T (SETQ NAME (\INDEXATOMDEF (IPLUS (LLSH B1 8)
                                                            B2]
                    (GO FN))
                (FNX [COND
                        [(FMEMB :4-BYTE COMPILER::*HOST-ARCHITECTURE*)
                         (SETQ NAME (\INDEXATOMDEF (IPLUS (LLSH (IPLUS (LLSH (IPLUS (LLSH B2 8)
                                                                                    B3)
                                                                             8)
                                                                       B4)
                                                                8)
                                                          B5]
                        [(FMEMB :3-BYTE COMPILER::*HOST-ARCHITECTURE*)
                         (SETQ NAME (\INDEXATOMDEF (IPLUS (LLSH (IPLUS (LLSH B2 8)
                                                                       B3)
                                                                8)
                                                          B4]
                        (T (SETQ NAME (\INDEXATOMDEF (IPLUS (LLSH B2 8)
                                                            B3]
                    (GO FN))
                (GCONST [SETQ FN (BIG-VMEM-HOST (\VAG2 (IPLUS (LLSH B1 8)
                                                              B2)
                                                       (IPLUS (LLSH B3 8)
                                                              B4))
                                                (\VAG2 B1 (IPLUS (LLSH B2 8)
                                                                 B3]
                        (COND
                           ((AND (OR (type? COMPILED-CLOSURE FN)
                                     (\CODEBLOCKP FN))
                                 (NOT (FMEMB FN IGNOREFNS)))
                            (push IGNOREFNS FN)
                            (GO COMPILED-CLOSURE))))
                ((GVAR GVAR_)
                    [SELECTQ OPTION
                        (FNAPPLY)
                        ((VARAPPLY APPLY)
                            (CL:FUNCALL FNAPPLY
                                   [COND
                                      ((FMEMB :4-BYTE COMPILER::*HOST-ARCHITECTURE*)
                                       (\INDEXATOMVAL (IPLUS (LLSH (IPLUS (LLSH (IPLUS (LLSH B1 8)
                                                                                       B2)
                                                                                8)
                                                                          B3)
                                                                   8)
                                                             B4)))
                                      ((FMEMB :3-BYTE COMPILER::*HOST-ARCHITECTURE*)
                                       (\INDEXATOMVAL (IPLUS (LLSH (IPLUS (LLSH B1 8)
                                                                          B2)
                                                                   8)
                                                             B3)))
                                      (T (\INDEXATOMVAL (IPLUS (LLSH B1 8)
                                                               B2]
                                   'GLOBALS))
                        (pushnew GLOBALS (COND
                                            ((FMEMB :4-BYTE COMPILER::*HOST-ARCHITECTURE*)
                                             (\INDEXATOMVAL
                                              (IPLUS (LLSH (IPLUS (LLSH (IPLUS (LLSH B1 8)
                                                                               B2)
                                                                        8)
                                                                  B3)
                                                            8)
                                                      B4)))
                                            ((FMEMB :3-BYTE COMPILER::*HOST-ARCHITECTURE*)
                                             (\INDEXATOMVAL (IPLUS (LLSH (IPLUS (LLSH B1 8)
                                                                                B2)
                                                                         8)
                                                                   B3)))
                                            (T (\INDEXATOMVAL (IPLUS (LLSH B1 8)
                                                                     B2])
                    NIL)
                (GO LP)
            FN  [SELECTQ OPTION
                    ((FNAPPLY APPLY)
                        (CL:FUNCALL FNAPPLY NAME 'CALLED))
                    (VARAPPLY)
                    (COND
                       ((FMEMB NAME IGNOREFNS)                    ; Don't show calls to these

                        )
                       ((SETQ FN (\SUBFNDEF NAME))
                        (push IGNOREFNS NAME)
                        (GO COMPILED-CLOSURE))
                       ((EQ OPTION T)                             ; Only look at vars
```

```
                    )
                   (T (pushnew CALLED NAME]
              (GO LP)
          COMPILED-CLOSURE
                                                    ; Compiled subfunction, recursively analyze it
              [LET ((RESULT (CALLSCCODE FN OPTION FNAPPLY)))
                   (AND RESULT (COND
                                ((EQ OPTION T)              ; Just got free variables back
                                 (SETQ USEDFREE (RUNION RESULT USEDFREE)))
                                (T (SETQ LNCALLED (RUNION (fetch LNCALLED of RESULT)
                                                          LNCALLED))
                                   (SETQ BOUND (RUNION (fetch BOUND of RESULT)
                                                       BOUND))
                                   (SETQ USEDFREE (RUNION (fetch USEDFREE of RESULT)
                                                          USEDFREE))
                                   (SETQ GLOBALS (RUNION (fetch GLOBALS of RESULT)
                                                         GLOBALS))
                                   (SETQ CALLED (RUNION (fetch CALLED of RESULT)
                                                        CALLED]
              (GO LP))
          (RETURN (SELECTQ OPTION
                       ((FNAPPLY VARAPPLY APPLY)
                           NIL)
                       (T                                   ; All free var references
                          (RUNION USEDFREE GLOBALS))
                       (create RESULT
                               LNCALLED _ (REVERSE LNCALLED)
                               CALLED _ (REVERSE CALLED)
                               BOUND _ (REVERSE BOUND)
                               USEDFREE _ (REVERSE USEDFREE)
                               GLOBALS _ (REVERSE GLOBALS])
```

(RUNION

```
  [LAMBDA (L1 L2)                                           (* bvm%: "14-Mar-86 14:27")

;;; Fast UNION using EQ

    (for X in L1 unless (FMEMB X L2) do (push L2 X))
    L2])

)

(DEFINEQ
```

(CHANGECCODE

```
  [LAMBDA (NEWREF OLDREF FN)                                ; Edited 13-Nov-92 14:13 by sybalsky:mv:envos

;;; A reference map is a list ('refmap' E1 ... EN), where each element E has the form (CODEARRAY NAMELOCS CONSTLOCS DEFLOCS PTRLOCS).
;;; The first element is for the main function, and further elements are for compiler-generated subfunctions.  Each LOCS list is a list of byte locations in
;;; the code to be fixed up in the indicated way (i.e.  VALINDEX, LOLOC, DEFINDEX, and full 24-bit pointer in GCONST format respectively).

    (DECLARE (SPECVARS ALL-CODE-BASES))                     ; ALL-CODE-BASES is list of all code bases examined.  See
                                                            ; CCCSUBFN? for details.
    (PROG ((SEAL '"refmap")
           DEF MAP ALL-CODE-BASES)
          (SETQ DEF (OR (\GET-COMPILED-CODE-BASE FN)
                        (RETURN)))
          [COND
             [(NEQ (CAR (LISTP OLDREF))
                   SEAL)                                    ; Construct a reference map for OLDREF in DEF
              (COND
                 ((EQ (PROG1 OLDREF
                             (SETQ OLDREF (CONS SEAL (CCCSCAN DEF OLDREF))))
                      NEWREF)                               ; No change, just return reference map
                  (RETURN OLDREF]
             ((NEQ (fetch (REFMAP CODEARRAY) of (CADR OLDREF))
                   DEF)
              (ERROR '"Inconsistent reference map" (CONS OLDREF FN]
                                                            ; Change all references in the map OLDREF to refer to NEWREF
          [for MAP in (CDR OLDREF) do (SETQ DEF (fetch CODEARRAY of MAP))
                                      [COND
                                         ((OR (fetch NAMELOCS of MAP)
                                              (fetch CONSTLOCS of MAP)
                                              (fetch DEFLOCS of MAP))
                                          (OR (LITATOM NEWREF)
                                              (ERROR "Can't changename a symbol to a non-symbol in compiled
                                                     code" NEWREF]
                                      [for LC in (fetch NAMELOCS of MAP) do (CODEBASESETATOM DEF LC
                                                                                (NEW-SYMBOL-CODE NEWREF
                                                                                       (\ATOMVALINDEX NEWREF]
                                      [for LC in (fetch CONSTLOCS of MAP) do (CODEBASESETATOM DEF LC
                                                                                (NEW-SYMBOL-CODE NEWREF
                                                                                       (\ATOMPNAMEINDEX NEWREF]
                                      [for LC in (fetch DEFLOCS of MAP) do (CODEBASESETATOM DEF LC
                                                                                (NEW-SYMBOL-CODE NEWREF
```

```
                                                                    (\ATOMDEFINDEX NEWREF]
                                        (for LC in (fetch PTRLOCS of MAP)
                                           do (UNINTERRUPTABLY
                                        ;; Decrement ref count of old literal, add new.  Order here is such that the worst that happens if it is
                                        ;; somehow aborted (despite the UNINTERRUPTABLY) is that the old and new literals never get
                                        ;; collected
                                                  (\ADDREF NEWREF)
                                                  (\DELREF (PROG1 (CODEBASELT3 DEF LC)
                                                                  (CODEBASESETA3 DEF LC NEWREF))))]
          (RETURN OLDREF])
```

(CCCSUBFN?

```
  [LAMBDA (X)                                                       ; Edited  9-Jun-88 20:53 by drc:
    (DECLARE (USEDFREE ALL-CODE-BASES SUBMAPS OLDREF))
    ;; X is a literal found in the code.  If X denotes a compiled subfunction, adds X's analysis to SUBMAPS.  Subfunctions are either a symbol fnA0nnn
    ;; or a compiled function object produced by PavCompiler.
    (LET [(BASE (CL:TYPECASE X
                    (COMPILED-CLOSURE (\GET-COMPILED-CODE-BASE X))
                    (LITATOM (AND (SETQ X (\SUBFNDEF X))
                                  (\GET-COMPILED-CODE-BASE X)))
                    (T (\CODEBLOCKP X)))]
         (if (AND BASE (NOT (FMEMB BASE ALL-CODE-BASES)))
             then (push ALL-CODE-BASES BASE)
                  ;; break circles by remembering what we've already analyzed in ALL-CODE-BASES
                  (SETQ SUBMAPS (NCONC SUBMAPS (CCCSCAN BASE OLDREF])
```

(\SUBFNDEF

```
  [LAMBDA (X)                                                       (* bvm%: " 7-Jul-86 16:31")
    (AND (LITATOM X)
         (EQ (NTHCHARCODE X -5)
             (CHARCODE A))
         [NOT (find I C from -4 to -1 suchthat (OR (ILESSP (SETQ C (NTHCHARCODE X I))
                                                          (CHARCODE 0))
                                                   (IGREATERP C (CHARCODE 9]
         (\GET-COMPILED-DEFINITION X])
```

(CCCSCAN

```
  [LAMBDA (DEF OLDREF)
    (DECLARE (SPECVARS SUBMAPS OLDREF))                             ; Edited 13-Nov-92 14:09 by sybalsky:mv:envos
    ;; Scan the code block DEF for instances of the symbol OLDREF.  Return a list of the instances and their locations, for use in doing
    ;; CHANGENAME, e.g.
    (PROG ((CA DEF)
           CONSTLOCS DEFLOCS PTRLOCS SUBMAPS NAMELOCS TAG B NAME CODELOC)
          (SETQ CODELOC (fetch (FNHEADER STARTPC) of CA))
          [COND
             ((LITATOM OLDREF)
              (for NT1 from (UNFOLD (fetch (FNHEADER OVERHEADWORDS) of T)
                                    BYTESPERWORD)
                 by (CONSTANT (BYTESPERNAMEENTRY)) do (OR (SETQ NAME (\INDEXATOMVAL (CODEBASEGETNAME CA NT1)))
                                                          (RETURN))
                                                   (AND (EQ NAME OLDREF)
                                                        (push NAMELOCS NT1]
      LP  (SETQ B (CODEBASELT CA CODELOC))
          (SETQ TAG (\FINDOP B))
          (add CODELOC (fetch OPNARGS of TAG)
               1)
          (SELECTQ (OR (fetch OPPRINT of TAG)
                       (fetch OPCODENAME of TAG))
              (-X- (RETURN (CONS (create REFMAP
                                         CODEARRAY _ CA
                                         NAMELOCS _ NAMELOCS
                                         CONSTLOCS _ CONSTLOCS
                                         DEFLOCS _ DEFLOCS
                                         PTRLOCS _ PTRLOCS)
                                 SUBMAPS)))
              ((FN FNX)
                   [SETQ NAME (CODEBASEGETATOM CA (IDIFFERENCE CODELOC (BYTESPERCODEATOM]
                   [COND
                      ([AND (LITATOM OLDREF)
                            (EQP NAME (NEW-SYMBOL-CODE OLDREF (\ATOMDEFINDEX OLDREF]
                       (push DEFLOCS (IDIFFERENCE CODELOC (BYTESPERCODEATOM]
                   (CCCSUBFN? (\INDEXATOMDEF NAME)))
              (ATOM [SETQ NAME (CODEBASEGETATOM CA (IDIFFERENCE CODELOC (BYTESPERCODEATOM]
                    [COND
                       ([AND (LITATOM OLDREF)
                             (EQ NAME (NEW-SYMBOL-CODE OLDREF (\ATOMPNAMEINDEX OLDREF]
                        (push CONSTLOCS (IDIFFERENCE CODELOC (BYTESPERCODEATOM]
                    (CCCSUBFN? (\INDEXATOMPNAME NAME)))
              (GCONST [COND
                         ((EQ [SETQ NAME (CODEBASELT3 CA (IDIFFERENCE CODELOC (BYTESPERCODEATOM]
```

```
                            OLDREF)
                        (push PTRLOCS (IDIFFERENCE CODELOC (BYTESPERCODEATOM]
                    (CCCSUBFN? NAME))
            NIL)
        (GO LP])
```

(\CODEBLOCKP

```
[LAMBDA (PTR)                                          ; Edited  5-Apr-88 18:49 by bvm
```

;; Returns PTR if it is a pointer to a raw code block, else NIL. Code blocks come in two varieties: code hunks and code arrayblocks. Hunks are ;; easy to check, because they have a distinct type. Arrayblocks are tricky to check, because they are typeless. The code here assumes that if you ;; pass a typeless pointer, it is a pointer to the start of an object. If you pass a pointer to the middle of a bitmap, for example, you could, if you were ;; very unlucky, get a false positive.

```
(AND (LET ((TEM (NTYPX PTR)))
          (if (EQ TEM 0)
              then ;; Maybe arrayblock. Carefully check that: it is in the range for arrayspace; its header (the previous cell) exists and
                   ;; contains the magic arrayblock password, the block's type is code, the block is in use, and its trailer is well-formed.
                   (AND (>= (\HILOC PTR)
                            \FirstArraySegment)
                        (PROGN (SETQ TEM (\ADDBASE PTR (- \ArrayBlockHeaderWords)))
                               (OR (>= (fetch (POINTER WORDINPAGE) of PTR)
                                       \ArrayBlockHeaderWords)
                                   (\VALIDADDRESSP TEM)))
                        (EQ (fetch (ARRAYBLOCK PASSWORD) of TEM)
                            \ArrayBlockPassword)
                        (EQ (fetch (ARRAYBLOCK GCTYPE) of TEM)
                            CODEBLOCK.GCT)
                        (fetch (ARRAYBLOCK INUSE) of TEM)
                        (\VALIDADDRESSP (SETQ TEM (fetch (ARRAYBLOCK TRAILER) of TEM)))
                        (EQ (fetch (ARRAYBLOCK PASSWORD) of TEM)
                            \ArrayBlockPassword))
             elseif (fetch DTDHUNKP of (SETQ TEM (\GETDTD TEM)))
               then                                    ; It's a hunk, check the hunk's gc type
                    (EQ (fetch DTDGCTYPE of TEM)
                        CODEBLOCK.GCT)))
      PTR])

)

(DEFINEQ
```

(\MAP-CODE-POINTERS

```
[LAMBDA (CODEBLOCK MAPFN)                              ; Edited 13-Nov-92 14:11 by sybalsky:mv:envos
```

;; CODEBLOCK is pointer to base of compiled code block. We walk thru the code and apply MAPFN to each pointer we find (i.e., GCONST). ;; MAPFN is called with three args: the pointer, CODEBLOCK, and the byte offset in CODEBLOCK where the pointer lives.

```
(COND
   ((NEQ [LET ((TYPENO (NTYPX CODEBLOCK)))
              (COND
                 [(EQ TYPENO 0)
                  (fetch (ARRAYBLOCK GCTYPE) of (\ADDBASE CODEBLOCK (IMINUS \ArrayBlockHeaderWords]
                 (T (fetch DTDGCTYPE of (\GETDTD TYPENO]
         CODEBLOCK.GCT)
    (ERROR "ARG NOT Compiled Code Block" CODEBLOCK))
   (T (PROG ((CODELOC (fetch (FNHEADER STARTPC) of CODEBLOCK))
             TAG)
        LP  (SETQ TAG (\FINDOP (CODEBASELT CODEBLOCK CODELOC)))
            (add CODELOC 1)
            (SELECTQ (fetch OPCODENAME of TAG)
                 (-X- (RETURN))
                 (GCONST (CL:FUNCALL MAPFN (CODEBASELT3 CODEBLOCK CODELOC)
                                     CODEBLOCK CODELOC))
                 NIL)
            (add CODELOC (fetch OPNARGS of TAG))
            (GO LP])
```

(\MAP-CODE-LITERALS

```
[LAMBDA (CODEBLOCK MAPFN)                              ; Edited 13-Nov-92 15:35 by sybalsky:mv:envos
```

;; CODEBLOCK is pointer to base of compiled code block. We walk thru the code and apply MAPFN to each literal we find (i.e., GCONST). ;; MAPFN is called with four args: the literal, CODEBLOCK, the byte offset in CODEBLOCK where the literal lives, and the type of literal, one of ;; ATOM, FN or POINTER. If you're only interested in pointers, the speedier \MAP-CODE-POINTERS is more appropriate.

```
(COND
   ((NEQ [LET ((TYPENO (NTYPX CODEBLOCK)))
              (COND
                 [(EQ TYPENO 0)
                  (fetch (ARRAYBLOCK GCTYPE) of (\ADDBASE CODEBLOCK (IMINUS \ArrayBlockHeaderWords]
                 (T (fetch DTDGCTYPE of (\GETDTD TYPENO]
         CODEBLOCK.GCT)
    (ERROR "ARG NOT Compiled Code Block" CODEBLOCK))
   (T (PROG ((CODELOC (fetch (FNHEADER STARTPC) of CODEBLOCK))
             TAG)
            (for NT1 from (UNFOLD (fetch (FNHEADER OVERHEADWORDS) of T)
                                  BYTESPERWORD)
```

```
                  by (BYTESPERNAMEENTRY) do (CL:FUNCALL MAPFN (OR (\INDEXATOMVAL (GETNAMEENTRY CODEBLOCK NT1))
                                                                  (RETURN))
                                                          CODEBLOCK NT1 'ATOM))
            LP  (SETQ TAG (\FINDOP (CODEBASELT CODEBLOCK CODELOC)))
                (add CODELOC (fetch OPNARGS of TAG)
                     1)
                (SELECTQ (OR (fetch OPPRINT of TAG)
                             (fetch OPCODENAME of TAG))
                    (-X- (RETURN))
                    ((FN FNX)
                         (CL:FUNCALL MAPFN [\INDEXATOMDEF (CODEBASELT3 CODEBLOCK (IDIFFERENCE CODELOC (
                                                                                          BYTESPERCODEATOM
                                                                                                ]
                                                  CODEBLOCK
                                                  (IDIFFERENCE CODELOC (BYTESPERCODEATOM))
                                                  'FN))
                    (ATOM (CL:FUNCALL MAPFN [\INDEXATOMPNAME (CODEBASELT3 CODEBLOCK (IDIFFERENCE CODELOC (
                                                                                          BYTESPERCODEATOM
                                                                                                ]
                                                   CODEBLOCK
                                                   (IDIFFERENCE CODELOC (BYTESPERCODEATOM))
                                                   'ATOM))
                    (GCONST (CL:FUNCALL MAPFN (\VAG2 (CODEBASELT2 CODEBLOCK (IDIFFERENCE CODELOC 4))
                                                     (CODEBASELT2 CODEBLOCK (IDIFFERENCE CODELOC 2)))
                                    CODEBLOCK
                                    (IDIFFERENCE CODELOC 4)
                                    'POINTER))
                    NIL)
                (GO LP])

)

(DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY

(BLOCK%: CALLSCCODE CALLSCCODE RUNION)

(BLOCK%: CHANGECCODE CHANGECCODE CCCSUBFN? CCCSCAN)
)
```

;; MACROS/OPTIMIZERS for getting and setting symbol entries in a compiled-code block. These are parameterized to allow for 2-, 3-, and 4-byte
;; symbol representations.

```
(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(RECORD REFMAP (CODEARRAY NAMELOCS CONSTLOCS DEFLOCS PTRLOCS))
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS CODEBASELT MACRO [OPENLAMBDA (CODEBASE OFFSET)
                            (COND
                               ((fetch (FNHEADER BYTESWAPPED) of CODEBASE)
                                (\GETBASEBYTE CODEBASE (LOGXOR OFFSET 3)))
                               (T (\GETBASEBYTE CODEBASE OFFSET])

(PUTPROPS CODEBASELT2 MACRO [OPENLAMBDA (DEF LC)
                             (LOGOR (LLSH (CODEBASELT DEF LC)
                                          BITSPERBYTE)
                                    (CODEBASELT DEF (ADD1 LC])

(PUTPROPS CODEBASESETA MACRO [OPENLAMBDA (CODEBASE OFFSET NEWVALUE)
                              (COND
                                 ((fetch (FNHEADER BYTESWAPPED) of CODEBASE)
                                  (\PUTBASEBYTE CODEBASE (LOGXOR OFFSET 3)
                                                NEWVALUE))
                                 (T (\PUTBASEBYTE CODEBASE OFFSET NEWVALUE])

(PUTPROPS CODEBASESETA2 MACRO [OPENLAMBDA (DEF LC VALUE)
                               (CODEBASESETA DEF LC (LRSH VALUE BITSPERBYTE))
                               (CODEBASESETA DEF (ADD1 LC)
                                      (IMOD VALUE (CONSTANT (LLSH 1 BITSPERBYTE])

(PUTPROPS CODEBASELT3 MACRO [OPENLAMBDA (DEF LC)
                             (BIG-VMEM-CODE [\VAG2 (LOGOR (LLSH (CODEBASELT DEF LC)
                                                                BITSPERBYTE)
                                                          (CODEBASELT DEF (ADD1 LC)))
                                                   (LOGOR (LLSH (CODEBASELT DEF (IPLUS 2 LC))
                                                                BITSPERBYTE)
                                                          (CODEBASELT DEF (IPLUS 3 LC]
                                            (\VAG2 (CODEBASELT DEF LC)
                                                   (LOGOR (LLSH (CODEBASELT DEF (IPLUS 1 LC))
                                                                BITSPERBYTE)
                                                          (CODEBASELT DEF (IPLUS 2 LC])

(PUTPROPS CODEBASELT4 MACRO [OPENLAMBDA (DEF LC)
```

```
                                (BIG-VMEM-CODE [\VAG2 (LOGOR (LLSH (CODEBASELT DEF LC)
                                                                   BITSPERBYTE)
                                                             (CODEBASELT DEF (ADD1 LC)))
                                                      (LOGOR (LLSH (CODEBASELT DEF (IPLUS 2 LC))
                                                                   BITSPERBYTE)
                                                             (CODEBASELT DEF (IPLUS 3 LC]
                                       (\VAG2 (CODEBASELT DEF LC)
                                              (LOGOR (LLSH (CODEBASELT DEF (IPLUS 1 LC))
                                                           BITSPERBYTE)
                                                     (CODEBASELT DEF (IPLUS 2 LC])

(PUTPROPS CODEBASESETA3 MACRO [OPENLAMBDA (DEF LC VALUE)
                                 (CODEBASESETA DEF LC (\HILOC VALUE))
                                 (CODEBASESETA DEF (ADD1 LC)
                                        (LRSH (\LOLOC VALUE)
                                              BITSPERBYTE))
                                 (CODEBASESETA DEF (IPLUS 2 LC)
                                        (IMOD (\LOLOC VALUE)
                                              (CONSTANT (LLSH 1 BITSPERBYTE])

(PUTPROPS CODEBASESETA4 MACRO [OPENLAMBDA (DEF LC VALUE)
                                 (CODEBASESETA DEF LC (LRSH (\HILOC VALUE)
                                                            BITSPERBYTE))
                                 [CODEBASESETA DEF (ADD1 LC)
                                        (IMOD (\HILOC VALUE)
                                              (CONSTANT (LLSH 1 BITSPERBYTE]
                                 (CODEBASESETA DEF (IPLUS 2 LC)
                                        (LRSH (\LOLOC VALUE)
                                              BITSPERBYTE))
                                 (CODEBASESETA DEF (IPLUS 3 LC)
                                        (IMOD (\LOLOC VALUE)
                                              (CONSTANT (LLSH 1 BITSPERBYTE])
)

(DEFOPTIMIZER CODEBASESETATOM (DEFINITION OFFSET SYMBOL &ENVIRONMENT ENV)
                              [COND
                                 [(FMEMB :4-BYTE (COMPILER::ENV-TARGET-ARCHITECTURE ENV))
                                  `(CODEBASESETA4 ,DEFINITION ,OFFSET ,SYMBOL]
                                 [(FMEMB :3-BYTE (COMPILER::ENV-TARGET-ARCHITECTURE ENV))
                                  `(CODEBASESETA3 ,DEFINITION ,OFFSET ,SYMBOL]
                                 (T `(CODESETA2 ,DEFINITION ,OFFSET ,SYMBOL])

(DEFOPTIMIZER CODEBASEGETATOM (DEFINITION OFFSET SYMBOL &ENVIRONMENT ENV)
                              ;; Get an atom out of a compiled function definition.
                              [COND
                                 [(FMEMB :4-BYTE (COMPILER::ENV-TARGET-ARCHITECTURE ENV))
                                  `(CODEBASELT4 ,DEFINITION ,OFFSET]
                                 [(FMEMB :3-BYTE (COMPILER::ENV-TARGET-ARCHITECTURE ENV))
                                  `(CODEBASELT3 ,DEFINITION ,OFFSET]
                                 (T `(CODEBASELT2 ,DEFINITION ,OFFSET ,SYMBOL])

(DEFOPTIMIZER CODEBASEGETNAME (BASE OFFSET &ENVIRONMENT ENV)
                              [COND
                                 [(FMEMB :4-BYTE (COMPILER::ENV-TARGET-ARCHITECTURE ENV))
                                  `(CODEBASEGETATOM ,BASE ,OFFSET]
                                 [(FMEMB :3-BYTE (COMPILER::ENV-TARGET-ARCHITECTURE ENV))
                                  `(CODEBASEGETATOM ,BASE ,OFFSET]
                                 (T `(CODEBASELT2 ,BASE ,OFFSET])

(DEFOPTIMIZER BYTESPERCODEATOM (&ENVIRONMENT ENV)
                               [COND
                                  ((FMEMB :4-BYTE (COMPILER::ENV-TARGET-ARCHITECTURE ENV))
                                   `(CONSTANT 4))
                                  ((FMEMB :3-BYTE (COMPILER::ENV-TARGET-ARCHITECTURE ENV))
                                   `(CONSTANT 3))
                                  (T `(CONSTANT 2])

(DEFOPTIMIZER BIG-VMEM-HOST (NEW-SYMBOL-FORM OLD-SYMBOL-FORM &ENVIRONMENT ENV)
                            ;; Allow for differences between 4-byte pointers and 3-byte pointers..
                            `(COND
                                ((FMEMB :4-BYTE COMPILER::*HOST-ARCHITECTURE*)
                                 ,NEW-SYMBOL-FORM)
                                (T ,OLD-SYMBOL-FORM)))

(FILESLOAD (LOADCOMP)
       LLGC LLCODE LLBASIC MODARITH RENAMEMACROS)
)

(ADDTOVAR IGNOREFNS )
```

```
;; Maintaining ref count consistency in code

(DEFINEQ

(\COPYCODEBLOCK
  [LAMBDA (NEWCA OLDCA NWORDS NEWFN)                    ; Edited  3-Mar-87 22:28 by bvm:
    ;; Copies code from an old code block OLDCA to a new block NEWCA.  Length of the code in words is NWORDS.  NEWFN is optional new frame
    ;; name for the code.
    (UNINTERRUPTABLY
        (\BLT NEWCA OLDCA NWORDS)
        ;; now have to fix up ref counts.  First increment ref count of everything in a GCONST
        [\MAP-CODE-POINTERS NEWCA (FUNCTION (LAMBDA (PTR)
                                              (\ADDREF PTR]
        ;; Then ref count the frame name (usually a no-op, if it's a symbol, but be careful anyway).
        (\ADDREF (IF NEWFN
                     THEN (replace (FNHEADER %#FRAMENAME) of NEWCA with NEWFN)
                          NEWFN
                   ELSE (fetch (FNHEADER %#FRAMENAME) of NEWCA)))
        NEWCA)])

(\COPYFNHEADER
  [LAMBDA (FNHD)                                        ; Edited  3-Mar-87 22:39 by bvm:
    ;; Returns a copy of just the header portion of FNHD -- the fixed header plus name table.  This is useable as a NAMETABLE on the stack, but not
    ;; as code.
    (PROG ((HEADWORDS (UNFOLD (fetch (FNHEADER NTSIZE) of FNHD)
                              2))
           NEWFNHD)
          [SETQ HEADWORDS (IPLUS (fetch (FNHEADER OVERHEADWORDS) of T)
                                 (COND
                                    ((EQ HEADWORDS 0)            ; No name table, but still need to copy quad of zeros
                                     WORDSPERQUAD)
                                    (T HEADWORDS]
          (SETQ NEWFNHD (\ALLOC.CODE.BLOCK (UNFOLD HEADWORDS BYTESPERCELL)
                                           HEADWORDS))           ; make it a code block, not just a regular block, so gc knows how
                                                                 ; to reclaim it
          (UNINTERRUPTABLY
              (\BLT NEWFNHD FNHD HEADWORDS)
              (replace (FNHEADER STARTPC) of NEWFNHD with 0)     ; make it unexecutable.  \RECLAIMCODEBLOCK also cares
                                                                 ; about this.
              (\ADDREF (fetch (FNHEADER FRAMENAME) of NEWFNHD)))
          (RETURN NEWFNHD])

(\RECLAIMCODEBLOCK
  [LAMBDA (CODEBASE)                                    ; Edited  6-May-88 13:01 by amd
    ;; Finalization for code hunks; also called by RECLAIMCODEBLOCK.  Decrements the reference count of all the literals in the block.
    (COND
       ((AND SI::*CLOSURE-CACHE-ENABLED* (XCL::GET-IMPLICIT-KEY-HASH CODEBASE SI::*CLOSURE-CACHE*))
        ;; clear cache entry
        (CL:SETF (XCL::GET-IMPLICIT-KEY-HASH CODEBASE SI::*CLOSURE-CACHE*)
                 NIL)
        ;; and don't reclaim (code block will be reclaimed next time 'round)
        T)
       (T (\DELREF (fetch (FNHEADER FRAMENAME) of CODEBASE))
          [IF (NEQ (fetch (FNHEADER STARTPC) of CODEBASE)
                   0)
              THEN ;; Code block never got filled in, or it's a vestigial one from \COPYFNHEADER
                   (\MAP-CODE-POINTERS CODEBASE (FUNCTION (LAMBDA (PTR)
                                                            (OR (EQ PTR CODEBASE)
                                                                (\DELREF PTR]
          ;; Return NIL to say it's ok to reclaim it now
          NIL])

)

;; Low-level break

(DEFINEQ

(LLBREAK
  [LAMBDA (FN WHEN)
    (DECLARE (GLOBALVARS BROKENFNS))                    ; Edited 15-Apr-87 18:33 by bvm:
    (PROG (NUFN DEF)
          (COND
             ((GETPROP FN 'BROKEN)
              (XCL:UNBREAK-FUNCTION FN)))
```

```
          (OR (SETQ DEF (\GET-COMPILED-DEFINITION FN))
              (ERROR FN "is not compiled code"))
          (/SETATOMVAL 'BROKENFNS (CONS FN BROKENFNS))
          (/PUTD [SETQ NUFN (PACK* FN (GENSYM 'L]
                 DEF T)
          (/PUTPROP FN 'BROKEN NUFN)
          (/PUTD FN (create COMPILED-CLOSURE using DEF FNHEADER _ (BROKENDEF DEF WHEN)))
          (RETURN FN])
```

(BROKENDEF

```
  [LAMBDA (DEF WHEN)                                          ; Edited 25-Jun-2017 22:16 by rmk:
    (PROG ((CA (\GET-COMPILED-CODE-BASE DEF))
           BEFORE AFTER SIZE FIRSTBYTE NEWCA)
          (SETQ FIRSTBYTE (fetch (FNHEADER STARTPC) of CA))
          (UNLESSRDSYS (SELECTQ WHEN
                           (BEFORE (SETQ BEFORE T))
                           (AFTER (SETQ AFTER T))
                           ((NIL BOTH)
                                (SETQ BEFORE T)
                                (SETQ AFTER T))
                           (LISPERROR "ILLEGAL ARG" WHEN)))      ; Check validity of WHEN before going uninterruptable
          (UNINTERRUPTABLY                                     ; Uninterruptable because of ref count modification
              (UNLESSRDSYS (PROGN                              ; Locally, create new code block and copy into it
                                (SETQ SIZE (UNFOLD (\#BLOCKDATACELLS CA)
                                                   BYTESPERCELL))
                                (SETQ NEWCA (\ALLOC.CODE.BLOCK (+ (COND
                                                                      (BEFORE 3)
                                                                      (T 0))
                                                                  SIZE)
                                                               (CEIL (ADD1 (FOLDHI FIRSTBYTE BYTESPERCELL))
                                                                     CELLSPERQUAD)))
                                (COND
                                   (BEFORE                     ; Need to insert preamble code
                                           (\MOVEBYTES CA 0 NEWCA 0 FIRSTBYTE)
                                                               ; Copy header
                                           [PROGN              ; insert call to RAID followed by a POP
                                                  [CODEBASESETA NEWCA FIRSTBYTE (CAR (\FINDOP '%'NIL]
                                                  [CODEBASESETA NEWCA (+ FIRSTBYTE 1)
                                                                (CAR (\FINDOP 'RAID]
                                                  (CODEBASESETA NEWCA (+ FIRSTBYTE 2)
                                                                (CAR (\FINDOP 'POP]
                                           (\MOVEBYTES CA FIRSTBYTE NEWCA (+ FIRSTBYTE 3)
                                                       (- SIZE FIRSTBYTE))
                                           (add FIRSTBYTE 3))
                                   (T                          ; Just copy verbatim
                                      (\MOVEBYTES CA 0 NEWCA 0 SIZE)))
                                (\ADDREF (fetch (FNHEADER FRAMENAME) of NEWCA))
                                                               ; count reference to framename

                                )
                     (PROGN                                    ; For Teleraid, can't create new code blocks, so can only make
                                                               ; break AFTER
                            (SETQ NEWCA CA)
                            (SETQ AFTER T)))                   ; rmk:  Remove (GO DOSCAN), since there is no place to go.
                                                               ; Seems reasonable to fall through to the AFTER test, if AFTER
                                                               ; was just set.
              [COND
                 (AFTER                                        ; Change all RETURNs to \RETURN
                         (bind OP do (SELECTQ [fetch (OPCODE OPCODENAME) of (SETQ OP (\FINDOP (CODEBASELT NEWCA
                                                                                                   FIRSTBYTE]
                                     (-X- (RETURN))
                                     (GCONST [UNLESSRDSYS (\ADDREF (\VAG2 (CODEBASELT NEWCA (+ FIRSTBYTE 1))
                                                                          (CODEBASELT2 NEWCA (+ FIRSTBYTE 2])
                                     (RETURN [CODEBASESETA NEWCA FIRSTBYTE (CAR (\FINDOP '\RETURN])
                                     NIL)
                                (add FIRSTBYTE 1 (fetch (OPCODE OPNARGS) of OP])
          (RETURN NEWCA])

)

;; for TELERAID

(DECLARE%: DONTCOPY

(ADDTOVAR RDCOMS (FNS PRINTCODE PRINTCODENT BROKENDEF))

(ADDTOVAR EXPANDMACROFNS NEXTBYTE PCVAR PRINJUMP CODEBASELT CODEBASELT2 CODEBASESETA CODEBASESETA2
                         PRINTCODEHEADERDECODE)
)

;; reference to opcodes symbolically

(DEFINEQ
```

(PRINTOPCODES

```
  [LAMBDA (SINGLE)                                                    (* lmm "22-Mar-85 10:34")
    (printout NIL "  #" 9 "name" 24 "len-1" 34 "format" 43 "stk effect" 55 "UFN table entry" T T)
    (for X in (COND
                (SINGLE (LIST (\FINDOP SINGLE)))
                (T \OPCODES))
       do [LET ((OP (fetch OP# of X)))
               (COND
                  ((LISTP OP)
                   (printout NIL .I3.8 (CAR OP)
                          "-"
                          (CADR OP)))
                  (T (printout NIL .I3.8 OP]
          (TAB 9)
          (PRIN1 (fetch OPCODENAME of X))
          [COND
             ((NEQ (fetch OPCODENAME of X)
                   'unused)
              (printout NIL 26 (OR (fetch OPNARGS of X)
                                   '?)
                     35
                     (OR (fetch OPPRINT of X)
                         '?)
                     44
                     (OR (fetch LEVADJ of X)
                         '?)
                     55
                     (OR (fetch UFNFN of X)
                         ""]
          (TERPRI])

)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \OPCODES)
)

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(LOCALVARS . T)
)
)

(PUTPROPS ACODE COPYRIGHT ("Venue & Xerox Corporation" 1982 1983 1984 1985 1986 1987 1988 1990 1991 1992 1995
                                    2017 2021))
```

FUNCTION INDEX

MACRO INDEX

OPTIMIZER INDEX

VARIABLE INDEX

RECORD INDEX