

File created: 19-Oct-94 12:30:11 {DSK}<lispcore>sources>LLGC.;3

changes to: (VARS LLGCCOMS)

previous date: 9-Feb-93 14:29:47 {DSK}<lispcore>sources>LLGC.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

;;
;; Copyright (c) 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1990, 1992, 1993, 1994 by Venue & Xerox Corporation. All rights reserved.

```
(RPAQQ LLGCCOMS
  ((PROPS (LLGC FILETYPE))
    (COMS                                     ; Reference counting
      (FNS \HTFIND \GC.HANDLEOVERFLOW \GCMAPTABLE))
    (COMS                                     ; Overflowed reference counts
      (FNS \GC.ENTER.BIGREFCNT \GC.MODIFY.BIGREFCNT \GC.LOOKUP.BIGREFCNT \GC.BIGREFCNT.MISSING)
      (GLOBALVARS \HTBIGCOUNT))
    (COMS                                     ; GC
      (FNS \GCMAPSCAN \GCMAPUNSCAN \GCRECLAIMCELL \FREELISTCELL \GCSCAN1 \GCSCAN2 \REFCNT \EQREFCNT1
        \SET.FINALIZATION.FUNCTION))
    [COMS                                     ; User entries
      (FNS RECLAIM \DORECLAIM \MAIKO.DORECLAIM RECLAIMMIN GCMESS GCGAG GCTRP)
      (ADDVARS (\MAIKO.MOVDS (\MAIKO.DORECLAIM \DORECLAIM]
    (COMS                                     ; Turning off GC
      (FNS DISABLEGC \DISABLEGC1 \MAIKO.DISABLEGC \DOGCDISABLEDINTERRUPT)
      (ADDVARS (\MAIKO.MOVDS (\MAIKO.DISABLEGC \DISABLEGC1)))
      (INITVARS (\GCDISABLED))
      (GLOBALVARS \GCDISABLED))
    (DECLARE%: EVAL@COMPILE DONTCOPY
      (EXPORT (MACROS ADDRDEF \ADDRDEF DELETeref \DELREF SCANREF \STKREF UNSCANREF CREATeref \CREATeref
        .INCREMENT.ALLOCATION.COUNT. .CHECK.ALLOCATION.COUNT. \GCDISABLED)
        (RECORDS HTOVERFLOW GC HTCOLL))
      (RECORDS GCOVFL MDSTYPEWORD GCPTR)
    ;; WORDSPERGENTRY should be 1 for non-BIGVM sysouts. Affects offsets into HTMAIN and HTCOLL.
    (CONSTANTS \HTBIGENTRYSIZE (\HT2CNT (IPLUS \HT1CNT \HT1CNT))
      (\HTCNTSHIFT 10)
      (\HTNOSTKBIT (LOGXOR 65535 \HTSTKBIT))
      (\HTSTK1 (LOGOR \HTSTKBIT \HT1CNT))
      (\HTSTKCNT (LOGOR \HTCNTMASK \HTSTKBIT))
      \HTHIMASK
      (\MAXHTCNT 32767)
      (WORDSPERGENTRY 2))
    (CONSTANTS \HTCOLLTHRESHOLD \HTCOLLMAX)
    (MACROS .GETLINK. .DELINK. .FREELINK. .MODENTRY. .NEWENTRY. .GCRECLAIMLP.)
    (GLOBALVARS \RECLAIMMIN \RECLAIM.COUNTDOWN \GCTIME1 \GCTIME2 \FINALIZATION.FUNCTIONS)
    (CONSTANTS \ADDRFCASE \DELREFCASE \SCANREFCASE \UNSCANREFCASE))
  [DECLARE%: DONTVAL@LOAD DOCOPY (INITVARS (\RECLAIMMIN 3000)
    (\RECLAIM.COUNTDOWN 3000)
    (GCMESS T)
    (\GCTIME1 (CREATECELL \FIXP))
    (\GCTIME2 (CREATECELL \FIXP]
  (FNS \GCERROR)
  [COMS                                     ; for MAKEINIT
    (FNS INITGC)
    (DECLARE%: DONTCOPY (ADDVARS (MKI.SUBFNS (ADDRDEF . PROGN)
      (\ADDRDEF . PROGN)
      (\DELREF . PROGN)
      (CREATeref . PROGN)
      (\CREATeref . PROGN)
      (DELETeref . PROGN)
      (.INCREMENT.ALLOCATION.COUNT. . PROGN)
      (.CHECK.ALLOCATION.COUNT. . PROGN)))
      (ADDVARS (INEWCMS (FNS INITGC)))
      EVAL@COMPILE
      (ADDVARS (DONTCOMPILEFNS INITGC]
    (LOCALVARS . T)))
  (PUTPROPS LLGC FILETYPE :BCOMPL)

;; Reference counting

(DEFINEQ

(\HTFIND
  [LAMBDA (PTR CASE)                                     ; Edited 1-Feb-87 15:05 by jop
    ;; Modify reference count of the constants ptr according to case --- Returns PTR if result is 0 ref cnt, NIL otherwise --- CASE is one of
    ;; (\ADDRFCASE \DELREFCASE \SCANREFCASE \UNSCANREFCASE)
    (PROG ((PROBE PTR)
      ENTRY LINK PREV)
```

```

(CHECK (NOT \INTERRUPTABLE))
[COND
  ((fetch (MDSTYPEWORD NOREFCNT) of (\ADDBASE \MDSTypeTable (LRSH (fetch (POINTER PAGE#) of PTR)
    1)))
    ; PTR not to be ref counted. Also true when GC disabled
  (RETURN))
  (\GCDISABLED
    ; Shouldn't happen
    (RETURN (NIL))
  (CHECK (EVENP (\LOLOC PTR)))
  (SETQ ENTRY (\ADDBASE \HTMAIN (LRSH (\LOLOC PROBE)
    1)))
  [COND
    ((fetch (GC EMPTY) of ENTRY)
      ; create new entry
      (RETURN (.NEWENTRY. ENTRY PTR CASE])
    [COND
      ((fetch (GC LINKP) of ENTRY)
        ; chase down the link
        (GO FINDLINK)))
    [COND
      ((EQ (\HILOC PTR)
        (fetch (GC HIBITS) of ENTRY))
        ; matches pointer in main table
        (RETURN (COND
          ((.MODENTRY. ENTRY CASE PTR)
            (replace (GC EMPTY) of ENTRY with T)
            NIL)
          ((EQ (fetch (GC STKCNT) of ENTRY)
            0)
            PTR)
          (T NIL]
    ]
  ]

```

;;; new collision

```

NEWCOLLISION
  (.GETLINK. LINK)
  (.GETLINK. PREV)
  (replace (GC NXTPTR) of PREV with (\LOLOC LINK))
  (replace (GC CONTENTS) of PREV with (fetch (GC CONTENTS) of ENTRY))
  (CHECK (EVENP (\LOLOC PREV)))
  (replace (GC LINKPTR) of ENTRY with (\LOLOC PREV))
  (replace (GC NXTPTR) of LINK with 0)
  (replace (GC EMPTY) of LINK with T)
  (RETURN (.NEWENTRY. LINK PTR CASE))
FINDLINK
  (SETQ LINK (\ADDBASE \HTCOLL (fetch (GC LINKPTR) of ENTRY)))
LINKLOOP
  (CHECK (SELECTC (fetch (GC HIBITS) of LINK)
    ((LIST \SmallPosHi \SmallNegHi \AtomHi)
      NIL)
    T))
  [COND
    ((EQ (fetch (GC HIBITS) of LINK)
      (\HILOC PTR))
      ; found the link entry
      (RETURN (COND
        ((.MODENTRY. LINK CASE PTR)
          ; reference count went to 1, delete list entry
          (.DELLINK. LINK PREV ENTRY)
          NIL)
        ((EQ 0 (fetch (GC STKCNT) of LINK)
          PTR)
          (T NIL]
      (SETQ PREV LINK)
      [COND
        ((NEQ (SETQ LINK (fetch (GC NXTPTR) of LINK))
          0)
          (SETQ LINK (\ADDBASE \HTCOLL LINK))
          (GO LINKLOOP)))
    ]

```

;;; Didn't find an entry on this chain

```

  (.GETLINK. LINK)
  (replace (GC NXTPTR) of LINK with 0)
  (CHECK PREV)
  (replace (GC NXTPTR) of PREV with (\LOLOC LINK))
  (RETURN (.NEWENTRY. LINK PTR CASE])

```

(\GC.HANDLEOVERFLOW

[LAMBDA (ARG) ; Edited 2-Feb-87 10:30 by jop

;; called as PUNT after microcode has put some things in the overflow table

```

(UNINTERRUPTABLY
  [PROG ((CELL \HTOVERFLOW)
    PTR)
    LP (COND
      ((SETQ PTR (fetch (HTOVERFLOW PTR) of CELL))
        (\HTFIND PTR (fetch (HTOVERFLOW CASE) of CELL))
        (replace (HTOVERFLOW CLEAR) of CELL with T)
        (SETQ CELL (\ADDBASE CELL WORDSPERCELL))
        (GO LP)))
    ]

```

```

      (PROGN (SETQ PTR (\GETDTD \LISTP))
        (COND
          ((IGREATERP (SETQ CELL (fetch DTD CNT0 of PTR))
            1024)
            (.INCREMENT.ALLOCATION.COUNT. CELL)
            (.BOXIPLUS. (fetch DTD CNTLOC of PTR)
              (fetch DTD CNT0 of PTR))
            (replace DTD CNT0 of PTR with 0]
        ARG) ] )

```

(\GC.MAPTABLE

```

[LAMBDA (ARG)
  (DECLARE (GLOBALVARS \MaxTypeNumber))
  ;; Called as a punt after microcode has done a CREATECELL and the count got big enough. Used to also be called when free list got empty.
  (UNINTERRUPTABLY
    [PROG ((CELL \HTOVERFLOW)
      PTR)
      LP (COND
        ((SETQ PTR (fetch (HTOVERFLOW PTR) of CELL))
          (\HTFIND PTR (fetch (HTOVERFLOW CASE) of CELL))
          (replace (HTOVERFLOW CLEAR) of CELL with T)
          (SETQ CELL (\ADDBASE CELL WORDSPERCELL))
          (GO LP]
        [COND
          (NIL (LET* ((DTD (\GETDTD (NTYPX ARG)))
            (N (fetch DTD CNT0 of DTD)))
            (.BOXIPLUS. (fetch DTD CNTLOC of DTD)
              N)
            (replace DTD CNT0 of DTD with 0)
            (.INCREMENT.ALLOCATION.COUNT. N)))
          (T
            ;; Generally we know that ARG's type caused the punt. At present we clean up EVERY counter so that the cumulative effect of all the
            ;; different types of CREATECELL contribute to deciding whether to gc. Not sure this is entirely necessary, and it gets slower as
            ;; more datatypes get allocated. Fortunately, \GC.MAPTABLE is only called when the count gets big, so is infrequent.
            (bind DTD N for I from 1 to \MaxTypeNumber when (NEQ [SETQ N (fetch DTD CNT0 of (SETQ DTD (\GETDTD
              I]
              0)
              do (.BOXIPLUS. (fetch DTD CNTLOC of DTD)
                N)
                (replace DTD CNT0 of DTD with 0)
                (.INCREMENT.ALLOCATION.COUNT. N]
            ARG) ] )
    )
  )

```

;; Overflowed reference counts

(DEFINEQ

(\GC.ENTER.BIGREFCNT

```

[LAMBDA (PTR ENTRY)
  ;; Called when the ref cnt of PTR is incremented to \MAXHTCNT. PTR is inserted in a simple table pointed to by \HTBIGCOUNT until its ref cnt
  ;; comes back down
  (PROG ((OENTRY \HTBIGCOUNT)
    TMP)
    [COND
      ((ODDP (\LOLOC PTR))
        (SETQ PTR (\ADDBASE PTR -1]
      LP [SELECTQ (SETQ TMP (fetch OVFLPTR of OENTRY))
        (T
          )
        (NIL
          [COND
            ((EVENP (\LOLOC (\ADDBASE OENTRY \HTBIGENTRYSIZE))
              WORDSPERPAGE)
              (\NEWPAGE (\ADDBASE OENTRY \HTBIGENTRYSIZE])
            (COND
              ((EQ TMP PTR)
                (\MP.ERROR \MP.BIGREFCNTALREADYPRESENT "PTR already in overflow table" PTR ENTRY)
                (add (fetch OVFLCNTHI of OENTRY)
                  1)
                (RETURN))
              (T (SETQ OENTRY (\ADDBASE OENTRY \HTBIGENTRYSIZE))
                (GO LP]
            (replace OVFLCNTLO of OENTRY with \MAXHTCNT)
            (replace OVFLCNTHI of OENTRY with 0)
            (replace OVFLPTR of OENTRY with PTR)
            (replace (GC CNT) of ENTRY with \MAXHTCNT])
    )
  )

```

(\GC.MODIFY.BIGREFCNT

```

[LAMBDA (ENTRY CASE PTR)
  ; Edited 1-Feb-87 15:00 by jop

```

;; Called from .MODENTRY. to do one of the 4 reference counting cases on PTR. ENTRY is the gc table entry whose CNT field is \MAXHTCNT

```
(PROG ((OENTRY \HTBIGCOUNT)
  TMP CNT)
  [COND
    ((ODDP (\LOLOC PTR))
      (SETQ PTR (\ADDBASE PTR -1]
      ; This should be an error, but accomodate it for now
    LP (COND
      ((NEQ (SETQ TMP (fetch OVFLPTR of OENTRY))
        PTR)
        (COND
          ((NULL TMP)
            (\GC.BIGREFCNT.MISSING PTR ENTRY)
            (RETURN)))
          (SETQ OENTRY (\ADDBASE OENTRY \HTBIGENTRYSIZE))
          (GO LP)))
      (SELECT CASE
        (\ADDREFCASE (replace OVFLCNTLO of OENTRY with (COND
          ((ILESSP (SETQ TMP (fetch OVFLCNTLO of OENTRY))
            MAX.SMALLP)
            (ADD1 TMP))
            (T (add (fetch OVFLCNTHI of OENTRY)
              1)
              0))))
          (\DELREFCASE (replace OVFLCNTLO of OENTRY with (COND
            ((IGEQ (SETQ TMP (SUB1 (fetch OVFLCNTLO
              of OENTRY)))
              \MAXHTCNT)
              TMP)
            (EQ 0 (fetch OVFLCNTHI of OENTRY))
            ; Ref cnt has fallen below max, bring it out
            (replace (GC CNT) of ENTRY with TMP)
            (replace OVFLPTR of OENTRY with T)
            ; mark deleted
            TMP)
            ((ILESSP TMP 0)
              (add (fetch OVFLCNTHI of OENTRY)
                -1)
              MAX.SMALLP)
            (T TMP))))
          (\SCANREFCASE (replace (GC STKBIT) of ENTRY with T))
          (\UNSCANREFCASE
            (replace (GC STKBIT) of ENTRY with NIL))
          NIL)
      (RETURN NIL]))
  ; Value is NIL to tell .MODENTRY. that cnt ~= 1
```

(\GC.LOOKUP.BIGREFCNT

[LAMBDA (PTR ENTRY)

; Edited 2-Feb-87 10:31 by jop

;; Returns ref cnt of PTR from the big table. ENTRY is the main or collision hashtable entry, but is used only for informational purposes to RAID

```
(PROG ((OENTRY \HTBIGCOUNT)
  TMP)
  [COND
    ((ODDP (\LOLOC PTR))
      (SETQ PTR (\ADDBASE PTR -1]
      ; This should be an error, but accomodate it for now
    LP (COND
      ((NEQ PTR (SETQ TMP (fetch OVFLPTR of OENTRY)))
        (COND
          ((NULL TMP)
            (\GC.BIGREFCNT.MISSING PTR ENTRY)
            (RETURN \MAXHTCNT))
          (SETQ OENTRY (\ADDBASE OENTRY \HTBIGENTRYSIZE))
          (GO LP)))
      (RETURN (\MAKENUMBER (fetch OVFLCNTHI of OENTRY)
        (fetch OVFLCNTLO of OENTRY]))
```

(\GC.BIGREFCNT.MISSING

[LAMBDA (PTR ENTRY)

(* JonL "14-Sep-84 00:46")

(\MP.ERROR \MP.BIGREFCNTMISSING "PTR refcnt previously overflowed, but not found in table." PTR ENTRY))

)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \HTBIGCOUNT)

)

;; GC

(DEFINEQ

(\GCMAPSCAN

[LAMBDA NIL

; Edited 2-Feb-87 10:31 by jop

;; scan gc tables looking for reclaimable items

```

(PROG (ENTRY PTR (PROBE \HTMAINSIZE)
      LINK PREV)
NEXTENTRY
[SETQ ENTRY (\ADDBASE \HTMAIN (SETQ PROBE (OR (\GCSCAN1 PROBE)
                                                (RETURN)]
RETRY
(COND
  ((fetch (GC LINKP) of ENTRY) ; trace down collision table
   (SETQ PREV NIL)
   (SETQ LINK (\ADDBASE \HTCOLL (fetch (GC LINKPTR) of ENTRY)))
   [PROG NIL
    LINKLOOP
      (CHECK (EVENP (\LOLOC LINK))
              (SELECTC (fetch (GC HIBITS) of LINK)
                        ((LIST \AtomHI \SmallPosHi \SmallNegHi)
                         NIL)
                        T)
              (NOT (fetch (GC LINKP) of LINK)))
      [COND
        ((EQ (fetch (GC STKCNT) of LINK)
              0)
         (SETQ PTR (\VAG2 (fetch (GC HIBITS) of LINK)
                           (LLSH PROBE 1)))
         (.DELLINK. LINK PREV ENTRY)
         (.GCRECLAIMLP. PTR)
         (COND
           ((fetch (GC EMPTY) of ENTRY)
            (GO NEXTENTRY))
           (T (GO RETRY]
         (SETQ PREV LINK)
         (COND
           ((NEQ 0 (SETQ LINK (fetch (GC NXTPTR) of LINK))
            (SETQ LINK (\ADDBASE \HTCOLL LINK))
            (GO LINKLOOP]
           (GO NEXTENTRY)))
         (CHECK (SELECTC (fetch (GC HIBITS) of ENTRY)
                   ((LIST \AtomHI \SmallPosHi \SmallNegHi)
                    NIL)
                   T))
         (COND
           ((EQ 0 (fetch (GC STKCNT) of ENTRY)) ; REF CNT WENT TO 0 -- ERASE ENTRY IN MAIN TABLE,
            (SETQ PTR (\VAG2 (fetch (GC HIBITS) of ENTRY) ; AND RECLAIM POINTER
                              (LLSH PROBE 1)))
            (replace (GC EMPTY) of ENTRY with T)
            (.GCRECLAIMLP. PTR)))
         (GO NEXTENTRY)])

```

(\GCMAPUNSCAN

```

[LAMBDA NIL ; Edited 2-Feb-87 10:32 by jop

```

```

;; scan gc tables turning of stack bits

```

```

(PROG ((PROBE \HTMAINSIZE)
      ENTRY)
LP [SETQ ENTRY (\ADDBASE \HTMAIN (SETQ PROBE (OR (\GCSCAN2 PROBE)
                                                (RETURN)]
RETRY
(COND
  [(fetch (GC LINKP) of ENTRY) ; LINK -- trace down collision table
   (PROG ((LNK (\ADDBASE \HTCOLL (fetch (GC LINKPTR) of ENTRY))
          PREV)
          SCNLP
            [COND
              ((fetch (GC STKBIT) of LNK)
               (COND
                 ((EQ (fetch (GC CNT) of LNK)
                      1) ; Ref count 1 with no stack bit => no entry
                  (.DELLINK. LNK PREV ENTRY) ; .DELLINK. smashes the chain, so don't try to follow it further
                  (GO RETRY))
                 (T (replace (GC STKBIT) of LNK with NIL]
              (COND
                ([NEQ 0 (SETQ LNK (fetch (GC NXTPTR) of (SETQ PREV LNK]
                 (SETQ LNK (\ADDBASE \HTCOLL LNK))
                 (GO SCNLP]
                ((fetch (GC STKBIT) of ENTRY)
                 (COND
                   ((EQ (fetch (GC CNT) of ENTRY)
                        1)
                    (replace (GC EMPTY) of ENTRY with T))
                   (T (replace (GC STKBIT) of ENTRY with NIL]
                 (GO LP]))

```

(\GCRECLAIMCELL

```

[LAMBDA (CELL) ; Edited 25-Mar-87 11:48 by bvm:

```

```

;; Called with CELL a pointer being freed. It has just had its refcount bumped from zero to one. We need to decrement the refcnt of anything it

```

```

;; points at, and if possible reclaim any of those that are now at zero count.
;; This is the new \GCRECLAIMCELL -- old version lives on as \OLDGCRECLAIMCELL if anyone wants the old behavior (uses microcode but
;; doesn't reclaim bushy structures)

(PROG ((PTR CELL)
      (DTD VAL TYPE INDEX DONEXT PTRFIELDS CODE FINAL)
      LP (CHECK (EQ 1 (\REFCNT PTR)))
      (SELECTC (SETQ TYPE (NTYPX PTR))
        (\LISTP (COND
          ((EQ CDRCODING 0)
           (GO NORMAL)))
        [COND
          ((EQ (SETQ CODE (fetch CDRCODE of PTR))
              \CDR.INDIRECT) ; Dispose of indirection first
           (SETQ PTR (PROG1 (fetch CARFIELD of PTR)
                           (\FREELISTCELL PTR)))
           (SETQ CODE (fetch CDRCODE of PTR))
           (CHECK (NEQ CODE \CDR.INDIRECT)
                  (ILEQ CODE \CDR.MAXINDIRECT)
          [COND
            (INDEX ; We've already decremented the CAR, start with the CDR
              (SETQ INDEX NIL))
            (T (COND
              ((SETQ VAL (\DELREF (CAR PTR))) ; CAR went to zero, start working on it
               (replace (GCPTR FULLLINKFIELD) of PTR with DONEXT)
               (replace CDRCODE of PTR with CODE) ; Keep CDR Code, which was smashed by FULLLINKFIELD
               (SETQ DONEXT PTR)
               (GO DOVAL)
              (SETQ VAL (\DELREF (CDR PTR)))
              [COND
                ((ILEQ CODE \CDR.MAXINDIRECT) ; indirect
                 ; local indirect
                 (\FREELISTCELL (\ADDBASE (fetch PAGEBASE of PTR)
                                           (LLSH (IDIFFERENCE CODE \CDR.INDIRECT)
                                                1]
                (\FREELISTCELL PTR)
                (GO DOVAL)
              (if (AND (NOT INDEX)
                      (SETQ FINAL (\GETBASEPTR \FINALIZATION.FUNCTIONS (UNFOLD TYPE WORDSPERCELL)))
                      (CL:FUNCALL FINAL PTR))
                  then ;; Type has a finalization that can perform cleanups. If returns T, says not to reclaim now. Don't do this when INDEX is
                  ;; true, because in that case we have already half reclaimed the object.
                  (GO TRYNEXT)))
                (GO TRYNEXT)))
        NORMAL
        (SETQ DTD (\GETDTD TYPE))
        (SETQ PTRFIELDS (fetch DTDPTRS of DTD))
        (COND
          (INDEX ;; We have half reclaimed PTR already. INDEX is the cell offset of the first field we haven't decremented yet
            (SETQ INDEX (UNFOLD INDEX WORDSPERCELL))
            (do (SETQ PTRFIELDS (CDR PTRFIELDS))
              (CHECK PTRFIELDS) repeatuntil (EQ (CAR PTRFIELDS)
                                                INDEX))
            (SETQ INDEX NIL)))
        [while PTRFIELDS do (COND
          ((SETQ VAL (\DELREF (\GETBASEPTR PTR (pop PTRFIELDS))
                              ; Suspend work on PTR, go chase VAL
          (COND
            (PTRFIELDS ; There is more to do
              (replace (GCPTR FULLLINKFIELD) of PTR with DONEXT)
              (CHECK (EVENP (CAR PTRFIELDS))
                     (ILESSP (CAR PTRFIELDS)
                             (UNFOLD (LLSH 1 BITSPERBYTE)
                                      WORDSPERCELL)))
              (replace (GCPTR OFFSETCODE) of PTR with (FOLDLO (CAR PTRFIELDS)
                                                                WORDSPERCELL))
              ; This assumes that no datatype is longer than 2^8 cells long
              (SETQ DONEXT PTR)
              (GO DOVAL)
            (T ; That was the last pointer field anyway, so finish up
              (GO ADDTOFREELIST]
        ADDTOFREELIST
        (\PUTBASEPTR PTR 0 (fetch DTDFREE of DTD))
        (replace DTDFREE of DTD with PTR)
        DOVAL
        (COND
          (VAL (\ADDRF (SETQ PTR VAL))
            (SETQ VAL NIL)
            (GO LP)))
        TRYNEXT
        (COND
          (DONEXT (SETQ PTR DONEXT)
                  (SETQ DONEXT (fetch (GCPTR LINKFIELD) of PTR))
                  (SETQ INDEX (fetch (GCPTR OFFSETCODE) of PTR))
                  (GO LP)))
          (RETURN NIL))

```

(FREELISTCELL

```

[LAMBDA (X)                                     (* Imm "1-JAN-82 23:54")
  (PROG ((BASE (fetch (POINTER PAGEBASE) of X))
    (CHECK (LISTP X)
      (EVENP (\LOLOC X)))
    (replace CDRCODE of X with (fetch NEXTCELL of BASE))
    (replace NEXTCELL of BASE with (fetch (POINTER WORD#) of X))
    (COND
      ((AND (IGREATERP (add (fetch (CONSPAGE CNT) of BASE)
        1)
          2)
        (EQ (fetch NEXTPAGE of BASE)
          \CONSPAGE.LAST))
        (replace NEXTPAGE of BASE with (fetch DTDNEXTPAGE of \LISTPDTD))
        (replace DTDNEXTPAGE of \LISTPDTD with (PAGELOC BASE]))

```

(GCSCAN1

```

[LAMBDA (PROBE)                                     ; Edited 2-Feb-87 10:27 by jop
  (PROG (ENT)
    LP (COND
      ((ILEQ PROBE 0)
        (RETURN NIL)))
    [SETQ ENT (\ADDBASE \HTMAIN (SETQ PROBE (SUB1 PROBE)
    (COND
      ([AND (NOT (fetch (GC EMPTY) of ENT))
        (OR (fetch (GC LINKP) of ENT)
          (EQ 0 (fetch (GC STKCNT) of ENT))
        (RETURN PROBE))
      (T (GO LP])

```

(GCSCAN2

```

[LAMBDA (PROBE)                                     (* Imm "23-DEC-81 22:48")
  (PROG NIL
    LP (COND
      ((ILEQ PROBE 0)
        (RETURN NIL))
      ((NEQ [LOGAND (CONSTANT (LOGOR \HTSTKBIT 1))
        (\GETBASE \HTMAIN (SETQ PROBE (SUB1 PROBE)
          0)
        (RETURN PROBE))
      (T (GO LP])

```

(REFCNT

```

[LAMBDA (PTR)                                     ; Edited 9-Feb-93 14:27 by jds
  (PROG (ENTRY LINK CNT)
    (COND
      ((fetch (MDSTYPEWORD NOREFCNT) of (\ADDBASE \MDSTypeTable (LRSH (fetch (POINTER PAGE#) of PTR)
        1)))
        ; PTR is not reference counted
      (RETURN 1)))
    (CHECK (EVENP (\LOLOC PTR)))
    (SETQ ENTRY (\ADDBASE \HTMAIN (UNFOLD (LRSH (\LOLOC PTR)
      1)
      WORDSPERGENTRY)))
    [COND
      ((fetch (GC EMPTY) of ENTRY)
        (RETURN 1))
      ((fetch (GC LINKP) of ENTRY)
        (GO FINDLINK))
        ; chase down the link
      ((NEQ (\HILOC PTR)
        (fetch (GC HIBITS) of ENTRY))
        ; Doesn't match ptr in table, so no entry
      (RETURN 1))
      ((ILESSP (SETQ CNT (fetch (GC CNT) of ENTRY))
        \MAXHTCNT)
      (RETURN CNT))
      (T
        ; Look in overflow table
        (RETURN (\GC.LOOKUP.BIGREFCNT PTR ENTRY]
    FINDLINK
      (SETQ LINK (\ADDBASE \HTCOLL (UNFOLD (fetch (GC LINKPTR) of ENTRY)
        WORDSPERGENTRY)))
    LINKLOOP
      [COND
        ((EQ (fetch (GC HIBITS) of LINK)
          (\HILOC PTR))
          ; found the link entry
        (RETURN (COND
          ((ILESSP (SETQ CNT (fetch (GC CNT) of LINK))
            \MAXHTCNT)
            CNT)
          (T (\GC.LOOKUP.BIGREFCNT PTR]
      (COND
        ((EQ (SETQ LINK (fetch (GC NXTPTR) of LINK))
          0)
          ; Didn't find an entry on this chain

```

```

(RETURN 1))
(T (SETQ LINK (\ADDBASE \HTCOLL (UNFOLD LINK WORDSPERGENTRY)))
  (GO LINKLOOP))

```

(EQREFCNT1

[LAMBDA (PTR)

; Edited 9-Feb-93 14:28 by jds

;; True if PTR's refcnt is definitely one -- this differs from (EQ (\REFCNT PTR) 1) because it is false for objects that are not reference counted, and
 ;; also for objects whose stack bit is on (during gc)

```

(PROG (ENTRY LINK)
  (COND
    ((fetch (MDSTYPEWORD NOREFCNT) of (\ADDBASE \MDSTypeTable (LRSH (fetch (POINTER PAGE#) of PTR)
      1)))
      ; PTR is not reference counted--ref cnt is indeterminate
    (RETURN NIL)))
  (CHECK (EVENP (\LOLOC PTR)))
  (SETQ ENTRY (\ADDBASE \HTMAIN (UNFOLD (LRSH (\LOLOC PTR)
    1)
    WORDSPERGENTRY)))
  [COND
    ((NOT (fetch (GC LINKP) of ENTRY))
      ; Ref cnt is 1 if there's no entry, or this entry is not for PTR
    (RETURN (OR (fetch (GC EMPTY) of ENTRY)
      (NEQ (\HILOC PTR)
        (fetch (GC HIBITS) of ENTRY)
        ; chase down the link
      (SETQ LINK (\ADDBASE \HTCOLL (UNFOLD (fetch (GC LINKPTR) of ENTRY)
        WORDSPERGENTRY))))
    LINKLOOP
  (COND
    ((EQ (fetch (GC HIBITS) of LINK)
      (\HILOC PTR))
      ; found the link entry, so must not be 1
    (RETURN NIL))
    ((EQ (SETQ LINK (fetch (GC NXTPTR) of LINK))
      0)
      ; Didn't find an entry on this chain
    (RETURN T))
    (T (SETQ LINK (\ADDBASE \HTCOLL (UNFOLD LINK WORDSPERGENTRY)))
      (GO LINKLOOP))
  )

```

(\SET.FINALIZATION.FUNCTION

[LAMBDA (TYPE FN)

; Edited 4-Mar-87 11:29 by bvm:

;; Make FN be the finalization fn for specified TYPE (number or name). Finalization fn is a function of one argument, a pointer whose ref count is
 ;; zero and about to be reclaimed. Fn returns NIL if ok to reclaim, T if not.

```

(LET [(TYPENO (OR (FIXP TYPE)
  (\TYPENUMBERFROMNAME TYPE)
  (IF (NOT (AND TYPENO (<= TYPENO \MaxTypeNumber))
    THEN (\ILLEGAL.ARG TYPE)
    ELSEIF (NOT (FNTYP FN))
    THEN (\ILLEGAL.ARG FN)
    ELSE (\PUTBASEPTR \FINALIZATION.FUNCTIONS (UNFOLD TYPENO WORDSPERCELL)
      FN))
  )

```

;; User entries

(DEFINEQ

(RECLAIM

```

[LAMBDA NIL
  (DORECLAIM)
  0])

```

(* Imm " 1-JUN-81 20:06")

(\DORECLAIM

```

[LAMBDA NIL
  (DECLARE (GLOBALVARS GCMESSE \RECLAIM.COUNTDOWN))
  (COND
    ((NOT (\GCDISABLED))
      (UNINTERRUPTABLY
        (SETQ \RECLAIM.COUNTDOWN NIL)
        (PROG ((GCTIME1 (CLOCK 2 \GCTIME1)))
          (AND GCMESSE (FLIPCUSOR))
          (\CONTEXTSWITCH \GCFXP)
          (AND GCMESSE (FLIPCUSOR))
          (\BOXIPLUS (LOCF (fetch GCTIME of \MISCSTATS))
            (\BOXIDIFFERENCE (CLOCK 2 \GCTIME2)
              GCTIME1)))
          (SETQ \RECLAIM.COUNTDOWN \RECLAIMMIN)))))
  )

```

(* Imm "15-OCT-82 12:12")

(\MAIKO.DORECLAIM

```

[LAMBDA NIL
  (SUBRCALL DORECLAIM)]

```

; Edited 12-Oct-88 12:01 by krivacic

(RECLAIMMIN

```

[LAMBDA (N)
  (PROG1 (OR \RECLAIMMIN T)
    [COND
      (N (SETQ \RECLAIM.COUNTDOWN (SETQ \RECLAIMMIN (COND
        ((AND (NOT \GCDISABLED)
          (NEQ N T))
        (IMIN (IMAX N 100)
          MAX.SMALL.INTEGER]))))
    (* bvm%: " 3-Sep-85 22:20")

```

(GCMESS

```

[LAMBDA (NUM STR)
  NIL])
(* Imm " 1-JUN-81 20:08")

```

(GCGAG

```

[LAMBDA (MESSAGE)
  (DECLARE (GLOBALVARS GCMESS))
  (PROG1 GCMESS (SETQ GCMESS MESSAGE))
(* rrb "11-JUN-81 10:13")

```

(GCTRP

```

[LAMBDA NIL
  ;; returns the number of storage allocations before the next gc
  (OR (FIXP \RECLAIM.COUNTDOWN)
    0])
; Edited 2-Feb-87 10:28 by jop

```

)

```

(ADDTovar \MAIKO.MOVDS (\MAIKO.DORECLAIM \DORECLAIM) )

```

```

;; Turning off GC

```

```

(DEFINEQ

```

(DISABLEGC

```

[LAMBDA (NOERROR)
  (UNINTERRUPTABLY
    (\DISABLEGC1 NOERROR)))
(* bvm%: " 3-Sep-85 21:49")

```

(\DISABLEGC1

```

[LAMBDA (NOERROR)
  ;; Do all the things necessary when GC must be turned off
  [LET ((TYPEBASE \MDSTypeTable))
    (FRPTQ (UNFOLD \MDSTTsize WORDSPERPAGE)
      (replace (MDSTYPEWORD NOREFCNT) of TYPEBASE with T)
      (SETQ TYPEBASE (\ADDBASE TYPEBASE 1]
    (SETQ \RECLAIM.COUNTDOWN (SETQ \RECLAIMMIN))
    (PROGN (COND
      ((AND (NOT NOERROR)
        (NOT \GCDISABLED))
        (replace GCDISABLED of \INTERRUPTSTATE with T)
        (SETQ \PENDINGINTERRUPT T)))
      (SETQ \GCDISABLED T))
    NIL])
; Edited 2-Feb-87 10:29 by jop
; Mark every type entry in the world 'don't ref count'
; Cause an interrupt and warning at next opportune time

```

(\MAIKO.DISABLEGC

```

[LAMBDA NIL
  (SUBRCALL DISABLEGC))
; Edited 7-Jun-90 19:04 by nm

```

(\DOGCDISABLEDINTERRUPT

```

[LAMBDA NIL
  ;; Called while interruptable after GC disabled. So informs user.
  (LET ((W (CREATEW (CREATEREGION 300 (IDIFFERENCE SCREENHEIGHT 100)
    450 100)
    "GC Disabled Warning")))
    (printout W T "Internal garbage collector tables have overflowed, due
      to too many pointers with reference count greater than 1.
      *** The garbage collector is now disabled. ***
      Save your work and reload as soon as possible.")
    (replace GCDISABLED of \INTERRUPTSTATE with NIL)
    (FLASHWINDOW W 4)
    (HELP "GC Disabled" "
      Save and reload a.s.a.p.")]
)
; Edited 2-Feb-87 10:29 by jop

```

```

(ADDTovar \MAIKO.MOVDS (\MAIKO.DISABLEGC \DISABLEGC1) )

```

```

(RPAQ? \GCDISABLED )

```

```
(DECLARE%: DOEVAL@COMPILE DONTCOPY
```

```
(GLOBALVARS \GCDISABLED)
)
```

```
(DECLARE%: EVAL@COMPILE DONTCOPY
```

```
:: FOLLOWING DEFINITIONS EXPORTED
```

```
(DECLARE%: EVAL@COMPILE
```

```
(PUTPROPS ADDRF MACRO (OPENLAMBDA (PTR)
                          (PROG1 PTR (\ADDRF PTR))))
```

```
(PUTPROPS ADDRF DMACRO ((X)
                          ((OPCODES GCREF 0)
                           X)))
```

```
(PUTPROPS DELETEREF MACRO (OPENLAMBDA (PTR)
                                   (PROG1 PTR (\DELRERF PTR))))
```

```
(PUTPROPS DELRERF DMACRO ((X)
                            ((OPCODES GCREF 1)
                             X)))
```

```
(PUTPROPS SCANREF MACRO (= . \STKREF))
```

```
(PUTPROPS STKREF DMACRO ((X)
                           ((OPCODES GCREF 2)
                            X)))
```

```
(PUTPROPS UNSCANREF MACRO ((PTR)
                              (\HTFIND PTR 3)))
```

```
(PUTPROPS CREATEREF MACRO (= . \CREATEREF))
```

```
(PUTPROPS CREATEREF MACRO (OPENLAMBDA (PTR)
                                   (PROG1 (\DELRERF PTR)
                                   (. INCREMENT.ALLOCATION.COUNT. 1))))
```

```
(PUTPROPS .INCREMENT.ALLOCATION.COUNT. MACRO [OPENLAMBDA (N)
                                                (DECLARE (GLOBALVARS \RECLAIM.COUNTDOWN))
                                                (AND \RECLAIM.COUNTDOWN (COND
                                                                ((IGREATERP \RECLAIM.COUNTDOWN N)
                                                                 (SETQ \RECLAIM.COUNTDOWN
                                                                    (IDIFFERENCE \RECLAIM.COUNTDOWN N)
                                                                    ))
                                                                (T (SETQ \RECLAIM.COUNTDOWN
                                                                    (\DORECLAIM]))
                                                                ))
                                                ]
```

```
(PUTPROPS .CHECK.ALLOCATION.COUNT. MACRO [OPENLAMBDA (N)
                                                (DECLARE (GLOBALVARS \RECLAIM.COUNTDOWN))
                                                (AND \RECLAIM.COUNTDOWN (COND
                                                                ((NOT (IGREATERP \RECLAIM.COUNTDOWN N))
                                                                 (SETQ \RECLAIM.COUNTDOWN
                                                                    (\DORECLAIM]))
                                                                ))
                                                ]
```

```
(PUTPROPS \GCDISABLED MACRO (NIL (PROGN (DECLARE (GLOBALVARS \GCDISABLED))
                                         \GCDISABLED)))
)
```

```
(DECLARE%: EVAL@COMPILE
```

```
[BLOCKRECORD HTOVERFLOW ((CASE BITS
                            4)
                           (PTR XPOINTER))
 (ACCESSFNS HTOVERFLOW ((CLEAR NIL (\PUTBASEFIXP DATUM 0 NIL]
```

```
[BLOCKRECORD GC ((CNT BITS 15)
                  (STKBIT FLAG)
                  (HIBITS BITS 15)
                  (LINKP FLAG)
                  (NXTPTR FIXP))
 (BLOCKRECORD GC ((STKCNT WORD)))
 (ACCESSFNS GC ((EMPTY (EQ 0 (\GETBASEFIXP DATUM 0))
                     (\PUTBASEFIXP DATUM 0 0))
                (CONTENTS (\GETBASEFIXP DATUM 0)
                           (\PUTBASEFIXP DATUM 0 NEWVALUE))
                (LINKPTR (LOGAND (\GETBASEFIXP DATUM 0)
                                  -2)
                          (\PUTBASEFIXP DATUM 0 (LOGOR NEWVALUE 1]
```

```
(BLOCKRECORD HTCOLL (;; An entry in the GC collision table. NEXTFREE is initialized to 2 by INITGC, as part of the MAKEINIT.
                     (FREEPTR FIXP) ; The GC table entry
                     (NEXTFREE FIXP) ; If the entry is in use, points to the next entry in this collision
```

; chain. If not, offset (in 1/2-entries) of the next free one on the
; chain.

)
)

:: END EXPORTED DEFINITIONS

(DECLARE%: EVAL@COMPILE

(BLOCKRECORD GCOVFL ((OVFLPTR FULLXPOINTER)
(OVFLCNTHI WORD)
(OVFLCNTLO WORD)))

(BLOCKRECORD MDSTPEWORD ((NOREFCNT FLAG)
(NIL BITS 15)))

[BLOCKRECORD GCPTR ((OFFSETCODE BITS 4)
(LINKFIELD XPOINTER)

; What to do next
; Link to next thing to work on after this

)
(BLOCKRECORD GCPTR ((FULLLINKFIELD FULLXPOINTER]
)

(DECLARE%: EVAL@COMPILE

(RPAQQ \HTBIGENTRYSIZE 4)

(RPAQ \HT2CNT (IPLUS \HT1CNT \HT1CNT))

(RPAQQ \HTCNTSHIFT 10)

(RPAQ \HTNOSTKBIT (LOGXOR 65535 \HTSTKBIT))

(RPAQ \HTSTK1 (LOGOR \HTSTKBIT \HT1CNT))

(RPAQ \HTSTKCNT (LOGOR \HTCNTMASK \HTSTKBIT))

(RPAQQ \HTHIMASK 510)

(RPAQQ \MAXHTCNT 32767)

(RPAQQ \WORDSPERGENTRY 2)

(CONSTANTS \HTBIGENTRYSIZE (\HT2CNT (IPLUS \HT1CNT \HT1CNT))
(\HTCNTSHIFT 10)
(\HTNOSTKBIT (LOGXOR 65535 \HTSTKBIT))
(\HTSTK1 (LOGOR \HTSTKBIT \HT1CNT))
(\HTSTKCNT (LOGOR \HTCNTMASK \HTSTKBIT))
\HTHIMASK
(\MAXHTCNT 32767)
(WORDSPERGENTRY 2))
)

(DECLARE%: EVAL@COMPILE

(RPAQQ \HTCOLLTHRESHOLD 65528)

(RPAQQ \HTCOLLMAX 65534)

(CONSTANTS \HTCOLLTHRESHOLD \HTCOLLMAX)
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS .GETLINK. MACRO [(VAR) ; get a new cell from free list into VAR
(SETQ VAR (fetch (HTCOLL FREEPTR) of \HTCOLL))
(COND
(EQ 0 VAR)
[COND
((IGE (SETQ VAR (fetch (HTCOLL NEXTFREE) of \HTCOLL))
\HTCOLLTHRESHOLD)
(\DISABLEGC1)
(COND
(EQ VAR \HTCOLLMAX) ; Don't wrap it around. Should never get here -- stop ref counting
; if gc is disabled!
(SETQ VAR (IDIFFERENCE VAR 2]
(replace (HTCOLL NEXTFREE) of \HTCOLL with (IPLUS VAR 2))
(SETQ VAR (\ADDBASE \HTCOLL VAR)))
(T (replace (HTCOLL FREEPTR) of \HTCOLL with (fetch (GC NXTPTR)
of (SETQ VAR (\ADDBASE \HTCOLL VAR))

(PUTPROPS .DELLINK. MACRO [(LINK PREV ENTRY)
(PROGN [COND
(PREV (replace (GC NXTPTR) of PREV with (fetch (GC NXTPTR) of LINK)))
(T (replace (GC LINKPTR) of ENTRY with (fetch (GC NXTPTR) of LINK)
; skip over this guy

```

        (.FREELINK. LINK) ; put him on the free list
        (COND
          ([EQ 0 (fetch (GC NXTPTR) of (SETQ LINK (\ADDBASE \HTCOLL
                                                    (fetch (GC LINKPTR)
                                                          of ENTRY]
                                                    ; if there is now only one entry on this chain, put him back on the
                                                    ; free list too
          (replace (GC CONTENTS) of ENTRY with (fetch (GC CONTENTS) of LINK))
          (.FREELINK. LINK])

(PUTPROPS .FREELINK. DMACRO (OPENLAMBDA (LINKCELL) ; put LINKCELL back on HTCOLL freelist
  (replace (GC CONTENTS) of LINKCELL with 0)
  (replace (GC NXTPTR) of LINKCELL with (fetch (HTCOLL FREEPTR) of \HTCOLL))
  (replace (HTCOLL FREEPTR) of \HTCOLL with (\LOLOC LINKCELL)))

(PUTPROPS .MODENTRY. DMACRO [(ENTRY CASE PTR)
  (PROG ((GCCNT (fetch (GC CNT) of ENTRY)))
    (DECLARE (LOCALVARS GCCNT))
    (COND
      [(NEQ GCCNT \MAXHTCNT)
        (SELECTC CASE
          (\ADDREFCASE [COND
            ((EQ GCCNT (SUB1 \MAXHTCNT))
              (\GC.ENTER.BIGREFCNT PTR ENTRY))
            (T (replace (GC CNT) of ENTRY with (ADD1 GCCNT)))
          (\DELREFCASE (OR (NEQ 0 GCCNT)
              (\MP.ERROR \MP.DELREF0 "DELREF on PTR with 0
                refcnt" PTR ENTRY))
              (replace (GC CNT) of ENTRY with (SUB1 GCCNT)))
          (\SCANREFCASE (replace (GC STKBIT) of ENTRY with T))
          (\UNSCANREFCASE
            (replace (GC STKBIT) of ENTRY with NIL))
          (\GCERROR))
        (RETURN (EQ (fetch (GC STKCNT) of ENTRY)
          (LLSH 1 1)
        (T (\GC.MODIFY.BIGREFCNT ENTRY CASE PTR]))

(PUTPROPS .NEWENTRY. MACRO [(ENTRY PTR CASE)
  (PROGN (CHECK (fetch (GC EMPTY) of ENTRY))
    (replace (GC HIBITS) of ENTRY with (\HILOC PTR))
    (SELECTC CASE
      (\ADDREFCASE (replace (GC CNT) of ENTRY with 2)
        NIL)
      (\DELREFCASE PTR)
      (\SCANREFCASE (replace (GC CNT) of ENTRY with 1)
        (replace (GC STKBIT) of ENTRY with T)
        NIL)
      (\GCERROR]))

(PUTPROPS .GCRECLAIMLP. DMACRO [(X)
  (PROG NIL
    LP (COND
      ((SETQ X (\GCRECLAIMCELL X))
        (\ADDREF X)
        (GO LP))
    )

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS \RECLAIMMIN \RECLAIM.COUNTDOWN \GCTIME1 \GCTIME2 \FINALIZATION.FUNCTIONS)

(DECLARE%: EVAL@COMPILE

(RPAQQ \ADDREFCASE 0)

(RPAQQ \DELREFCASE 1)

(RPAQQ \SCANREFCASE 2)

(RPAQQ \UNSCANREFCASE 3)

(CONSTANTS \ADDREFCASE \DELREFCASE \SCANREFCASE \UNSCANREFCASE)
)

(DECLARE%: DONTVAL@LOAD DOCOPY

(RPAQ? \RECLAIMMIN 3000)

(RPAQ? \RECLAIM.COUNTDOWN 3000)

(RPAQ? GCMESST)

(RPAQ? \GCTIME1 (CREATECELL \FIXP))

(RPAQ? \GCTIME2 (CREATECELL \FIXP))

```

```

)

(DEFINEQ
  (\GCERROR
    [LAMBDA (REASON FLG)
      (PROG NIL
        (COND
          ((AND FLG REASON (\GCDISABLED))
            (RETURN)))
          (until (RAID (OR REASON "Bad CASE arg to \HTFIND"))
            (DISABLEGC]))
        )
      )
    (* Imm "8-DEC-81 14:21")
  )

```

```

;; for MAKEINIT

```

```

(DEFINEQ
  (INITGC
    [LAMBDA NIL
      (CREATEPAGES \HTMAIN (FOLDHI \HTMAINSIZE WORDSPERPAGE)
        T T)
      (CREATEPAGES \HTOVERFLOW 1 T T)
      (CREATEPAGES \HTBIGCOUNT 1 T)
      (CREATEPAGES \HTCOLL 1 NIL T)
      (CREATEPAGES (\ADDBASE \HTCOLL WORDSPERPAGE)
        (SUB1 (FOLDHI \HTCOLLSIZE WORDSPERPAGE))
        T)
      (replace (HTCOLL FREEPTR) of \HTCOLL with 0)
      (replace (HTCOLL NEXTFREE) of \HTCOLL with 2])
    )
    (* bvm%: "13-Feb-84 18:14")
  )

```

```

(DECLARE%: DONTCOPY

```

```

(ADDTOVAR MKI.SUBFNS (ADDRF . PROGN)
  (\ADDRF . PROGN)
  (\DELREF . PROGN)
  (CREATEREF . PROGN)
  (\CREATEREF . PROGN)
  (DELETEREF . PROGN)
  (.INCREMENT.ALLOCATION.COUNT. . PROGN)
  (.CHECK.ALLOCATION.COUNT. . PROGN))

```

```

(ADDTOVAR INEWCOMS (FNS INITGC))

```

```

(ADDTOVAR DONTCOMPILEFNS INITGC)
)

```

```

(DECLARE%: DOEVAL@COMPILE DONTCOPY

```

```

(LOCALVARS . T)
)

```

```

(PUTPROPS LLGC COPYRIGHT ("Venue & Xerox Corporation" 1981 1982 1983 1984 1985 1986 1987 1988 1990 1992 1993
  1994))

```

FUNCTION INDEX

DISABLEGC	9	\EQREFCNT1	8	\GCMAPUNSCAN	5
GCGAG	9	\FREELISTCELL	7	\GCRECLAIMCELL	5
GCMESS	9	\GC.BIGREFCNT.MISSING	4	\GCSCAN1	7
GCTRP	9	\GC.ENTER.BIGREFCNT	3	\GCSCAN2	7
INITGC	13	\GC.HANDLEOVERFLOW	2	\HTFIND	1
RECLAIM	8	\GC.LOOKUP.BIGREFCNT	4	\MAIKO.DISABLEGC	9
RECLAIMMIN	9	\GC.MODIFY.BIGREFCNT	3	\MAIKO.DORECLAIM	8
\DISABLEGC1	9	\GCERROR	13	\REFCNT	7
\DOGCDISABLEDINTERRUPT	9	\GCMAPSCAN	4	\SET.FINALIZATION.FUNCTION	8
\DORECLAIM	8	\GCMAPTABLE	3		

MACRO INDEX

.CHECK.ALLOCATION.COUNT.	10	.MODENTRY.	12	UNSCANREF	10
.DELLINK.	11	.NEWENTRY.	12	\ADDRF	10
.FREELINK.	12	ADDRF	10	\CREATEREF	10
.GCRECLAIMLP.	12	CREATEREF	10	\DELREF	10
.GETLINK.	11	DELETEREF	10	\GCDISABLED	10
.INCREMENT.ALLOCATION.COUNT.	10	SCANREF	10	\STKREF	10

CONSTANT INDEX

WORDSPERGENTRY ..11	\HT2CNT	11	\HTCOLLMAX	11	\HTNOSTKBIT	11	\MAXHTCNT	11
\ADDRFCASE	12	\HTBIGENTRYSIZE ..11	\HTCOLLTHRESHOLD ..11	\HTSTK1	11	\SCANREFCASE	12	
\DELREFCASE	12	\HTCNTSHIFT	11	\HTSTKCNT	11	\UNSCANREFCASE	12	

VARIABLE INDEX

DONTCOMPILEFNS	13	MKI.SUBFNS	13	\GCTIME2	12	\RECLAIMMIN	12
GCMESS	12	\GCDISABLED	9	\MAIKO.MOVDS	9		
INWCMS	13	\GCTIME1	12	\RECLAIM.COUNTDOWN	12		

RECORD INDEX

GC	10	GCOVFL	11	GCPTR	11	HTCOLL	10	HTOVERFLOW ...	10	MDSTYPEWORD ..	11
----------	----	--------------	----	-------------	----	--------------	----	----------------	----	----------------	----

PROPERTY INDEX

LLGC	1
------------	---
