File created: 18-Oct-93 16:35:37 **{Pele:mv:envos}<LispCore>Sources>CLTL2>LLMVS.;2**

previous date: 3-Sep-91 18:10:25 {Pele:mv:envos}<LispCore>Sources>CLTL2>LLMVS.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

;;


```
(RPAQQ LLMVSCOMS
       [
```

;;; Runtime support for multiple value passing. This file must be present for compiled multiple values to work.

```
        (FNS CL:VALUES CL:VALUES-LIST \MVLIST \SIMULATE.UNBIND)
        (DECLARE%: DONTCOPY (MACROS \VALUES \VALUES-UFN)
               (LOCALVARS . T))
        (VARIABLES CL:MULTIPLE-VALUES-LIMIT)
```

;; UFNs for the CL:VALUES and CL:VALUES-LIST sub-opcodes of MISCN:

```
        (FNS CL::VALUES-UFN CL::VALUES-LIST-UFN)
        (PROP FILETYPE LLMVS)
        (DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA)
                                                                                (NLAML)
                                                                                (LAMA CL:VALUES])
```

;;; Runtime support for multiple value passing. This file must be present for compiled multiple values to work.

```
(DEFINEQ
```

(CL:VALUES

```
  [LAMBDA ARGS                                                    ; Edited 30-May-90 16:01 by jds
```

;; Return multiple values to a caller.

```
    (\VALUES (for I from 1 to ARGS collect (ARG ARGS I))
           (AND (IGEQ ARGS 1)
                (ARG ARGS 1])
```

(CL:VALUES-LIST

```
  [LAMBDA (CL:VALUES)                                             ; Edited 30-May-90 16:02 by jds
```

;; Given a list of values, return them as multiple values to a caller.

```
    (\VALUES CL:VALUES (CAR CL:VALUES])
```

(\MVLIST

```
  [LAMBDA (X)
    (LIST X])
```

(\SIMULATE.UNBIND

```
  [LAMBDA (FRAME N RETURNER)                                      ; Edited 25-Nov-87 12:54 by bvm:
```

;; Simulate the action of N applications of UNBIND occurring in specified FRAME. RETURNER is the frame that will return to FRAME, and hence
;; must be made slow (NIL if my caller). Must be called uninterruptably.

```
    (LET* [(NEXT (fetch (FX NEXTBLOCK) of FRAME))
           (SP NEXT)
           (PVAR0BASE (STACKADDBASE (fetch (FX FIRSTPVAR) of FRAME]
          [TO N DO (do                                            ; Pop stack until a bind mark is encountered
                     (SETQ SP (- SP WORDSPERCELL)) REPEATUNTIL (fetch BINDMARKP of (STACKADDBASE SP))
                   FINALLY                                        ; Unbind stuff. Bind mark says how many pvars were bound,
                                                                  ; and gives the offset of the last of them
                       (LET [(LASTPVAR (fetch BINDLASTPVAR of (STACKADDBASE SP]
                            (to (fetch BINDNVALUES of (STACKADDBASE SP)) do (\PUTBASE PVAR0BASE LASTPVAR
                                                                                      65535)
                                                                            (SETQ LASTPVAR (- LASTPVAR
                                                                                              WORDSPERCELL]
          (replace (FX NEXTBLOCK) of FRAME with SP)
          (\MAKEFREEBLOCK SP (- NEXT SP))
```

;; Now explicitly slow return to FRAME, since we have violated the fast return assumptions by blowing away stack between here and there

```
          (replace (FX FASTP) of (OR RETURNER (\MYALINK)) with NIL])
```

```
)

(DECLARE%: DONTCOPY

(DECLARE%: EVAL@COMPILE

(PUTPROPS \VALUES MACRO [(MANY ONE CALLER-FRAME)
```

```
(PROG* ((IMMEDIATE-CALLER (OR CALLER-FRAME (\MYALINK)))
        (CALLER IMMEDIATE-CALLER)
        PREVFRAME)
```

;; NB: THIS MACRO MUST TRACK \VALUES-UFN, EXCEPT FOR THE PC-SETTING CODE. THIS ONE IS ;; USED IN THE FUNCTIONS CL:VALUES AND CL:VALUES-LIST.

;; This macro is used by VALUES and VALUES-LIST to possibly return multiple values. It works by examining the ;; caller to see if the next instruction is MVLIST (currently in the form of a FN1 \MVLIST), which is present in all ;; multiple-value receivers. If so, it bumps the pc past there and returns the MANY expression, whose value is a list ;; of all the values. If it encounters RETURN instead, the call was tail-recursive, so procedure repeats with caller's ;; caller, etc. Otherwise, multiple values are not expected, and the macro returns just ONE value (the first) to the ;; caller.

```
  NEWFRAME
      (RETURN (PROG ((PC (fetch (FX PC) of CALLER))
                     (CODE (fetch (FX FNHEADER) of CALLER))
                     (NUNBINDS 0)
                     BYTE)
                NEWPC
                    [SELECTC (SETQ BYTE (\GETBASEBYTE CODE PC))
                        ((LIST (OP# RETURN)
                               (OP# \RETURN))
                                ; Call is tail-recursive, so iterate. \RETURN is for LLBREAKing.
                            (SETQ PREVFRAME CALLER)
                            (SETQ CALLER (fetch (FX CLINK) of CALLER))
                            (GO NEWFRAME))
                        ((OP# FN1)   ; Could be MVLIST
                            (SELECTQ [\INDEXATOMDEF
                                      (NEW-SYMBOL-CODE
                                       [\VAG2 (\GETBASEBYTE CODE (+ PC 1))
                                              (create WORD
                                                      HIBYTE _
                                                      (\GETBASEBYTE CODE
                                                            (+ PC 2))
                                                      LOBYTE _
                                                      (\GETBASEBYTE CODE
                                                            (+ PC 3]
                                       (create WORD
                                              HIBYTE _ (\GETBASEBYTE CODE
                                                            (+ PC 1))
                                              LOBYTE _ (\GETBASEBYTE CODE
                                                            (+ PC 2]
                                (\MVLIST
                                    ; Bump PC past the call, and return the values list
                                          (UNINTERRUPTABLY
                                              (COND
                                                 ((NEQ NUNBINDS 0)
                                    ; Sigh.  We have to simulate the unbinding, since we need to get
                                    ; past the MVLIST.
                                                       (\SIMULATE.UNBIND CALLER NUNBINDS
                                                            PREVFRAME)))
                                       ;; Update the PC to skip over the FN1 opcode 1+(# of bytes in
                                       ;; a symbol in the code stream):
                                                       (replace (FX PC) of CALLER
                                                          with (NEW-SYMBOL-CODE (+ PC 4)
                                                                       (+ PC 3))))
                                                  (RETURN MANY))
                                          NIL))
                        ((OP# UNBIND)
                                    ; UNBIND appears.  This preserves the top of stack, so it should
                                    ; also preserve multiple values.
                            (add PC 1)
                            (add NUNBINDS 1)
                            (GO NEWPC))
                        ((OP# JUMPX)
                                    ; Follow the jump (yecch)
                            (add PC (COND
                                        ((>= (SETQ BYTE (\GETBASEBYTE CODE
                                                                (+ PC 1)))
                                             128)
                                         (- BYTE 256))
                                        (T BYTE)))
                            (GO NEWPC))
                        ((OP# JUMPXX)
                            (add PC (SIGNED (create WORD
                                                    HIBYTE _ (\GETBASEBYTE
                                                              CODE
                                                              (+ PC 1))
                                                    LOBYTE _ (\GETBASEBYTE
                                                              CODE
                                                              (+ PC 2)))
                                            BITSPERWORD))
                            (GO NEWPC))
                        (LET [(JUMPBASE (CONSTANT (CAAR (\FINDOP 'JUMP]
                             (COND
                                ([<= JUMPBASE BYTE (CONSTANT
                                                    (CADAR (\FINDOP 'JUMP]
```

```
                                                    (add PC (+ (- BYTE JUMPBASE)
                                                               2))
                                                    (GO NEWPC]
                                            (RETURN ONE])

(PUTPROPS \VALUES-UFN MACRO
          [(MANY ONE CALLER-FRAME RESULT-IVAR)
           (PROG* ((IMMEDIATE-CALLER (OR CALLER-FRAME (\MYALINK)))
                   (CALLER IMMEDIATE-CALLER)
                   PREVFRAME)
```

;; NB: THIS MACRO MUST TRACK \VALUES, EXCEPT FOR THE PC SETTING CODE. THIS ONE IS USED IN THE UFNs FOR ;; VALUES AND VALUES-LIST.

;; This macro is used by VALUES and VALUES-LIST to possibly return multiple values. It works by examining the caller to see if the ;; next instruction is MVLIST (currently in the form of a FN1 \MVLIST), which is present in all multiple-value receivers. If so, it bumps the ;; pc past there and returns the MANY expression, whose value is a list of all the values. If it encounters RETURN instead, the call was ;; tail-recursive, so procedure repeats with caller's caller, etc. Otherwise, multiple values are not expected, and the macro returns just ;; ONE value (the first) to the caller.

```
             NEWFRAME
                  (RETURN (PROG ((PC (fetch (FX PC) of CALLER))
                                 (CODE (fetch (FX FNHEADER) of CALLER))
                                 (NUNBINDS 0)
                                 BYTE)
                           NEWPC
                               [SELECTC (SETQ BYTE (\GETBASEBYTE CODE PC))
                                    ((LIST (OP# RETURN)
                                           (OP# \RETURN))        ; Call is tail-recursive, so iterate. \RETURN is for LLBREAKing.
                                         (SETQ PREVFRAME CALLER)
                                         (SETQ CALLER (fetch (FX CLINK) of CALLER))
                                         (GO NEWFRAME))
                                    ((OP# FN1)                   ; Could be MVLIST
                                         (SELECTQ [\INDEXATOMDEF (create WORD
                                                                         HIBYTE _ (\GETBASEBYTE CODE
                                                                                        (+ PC 1))
                                                                         LOBYTE _ (\GETBASEBYTE CODE
                                                                                        (+ PC 2]
                                             (\MVLIST            ; Bump PC past the call, and return the values list
                                                      (LET (VALS)
                                                           (SETQ VALS MANY)
```

;; This LET & SETQ forces MANY to be computed before we dink with the stack ;; (which seems to destroy some of the values!)

```
                                                           (REPLACE (FX NEXTBLOCK) OF IMMEDIATE-CALLER
                                                              WITH (LOLOC RESULT-IVAR))
                                                           (UNINTERRUPTABLY
                                                               (COND
                                                                  ((NEQ NUNBINDS 0)
                                                                  ; Sigh. We have to simulate the unbinding, since we need to get
                                                                  ; past the MVLIST.
                                                                     (\SIMULATE.UNBIND CALLER NUNBINDS PREVFRAME))
                                                                  )
                                                               [COND
                                                                  ((EQ CALLER IMMEDIATE-CALLER)
```

;; If the immediate caller has the MVLIST, then the PC has already been ;; bumped, courtesy of the microcode.

```
                                                                    (replace (FX PC) of CALLER
                                                                       with (+ PC 3)))
                                                                   (T ;; Otherwise, we should skip over the FN1 \MVLIST.
                                                                      (replace (FX PC) of CALLER
                                                                         with (+ PC 3])
                                                           (SI::UNWIND IMMEDIATE-CALLER)
                                                           (RETURN VALS)))
                                                    NIL))
                                    ((OP# UNBIND)                ; UNBIND appears. This preserves the top of stack, so it should
                                                                 ; also preserve multiple values.
                                         (add PC 1)
                                         (add NUNBINDS 1)
                                         (GO NEWPC))
                                    ((OP# JUMPX)                 ; Follow the jump (yecch)
                                         (add PC (COND
                                                    ((>= (SETQ BYTE (\GETBASEBYTE CODE (+ PC 1)))
                                                         128)
                                                     (- BYTE 256))
                                                    (T BYTE)))
                                         (GO NEWPC))
                                    ((OP# JUMPXX)
                                         (add PC (SIGNED (create WORD
                                                                 HIBYTE _ (\GETBASEBYTE CODE (+ PC 1))
                                                                 LOBYTE _ (\GETBASEBYTE CODE (+ PC 2)))
                                                         BITSPERWORD))
                                         (GO NEWPC))
                                    (LET [(JUMPBASE (CONSTANT (CAAR (\FINDOP 'JUMP]
                                         (COND
                                            ([<= JUMPBASE BYTE (CONSTANT (CADAR (\FINDOP 'JUMP]
                                             (add PC (+ (- BYTE JUMPBASE)
```

```
                                                    2))
                                              (GO NEWPC]
                                    (RETURN ONE])
)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(LOCALVARS . T)
)
)
```

(CL:DEFCONSTANT **CL:MULTIPLE-VALUES-LIMIT** 512)

;; UFNs for the CL:VALUES and CL:VALUES-LIST sub-opcodes of MISCN:

(DEFINEQ

(CL::VALUES-UFN

```
  [LAMBDA (CL::INDEX CL::ARGCOUNT CL::ARG-PTR)                    ; Edited 5-Jun-90 15:21 by jds
```

;; This is the UFN for the VALUES MISCN opcode. Its definition must be analogous to that for CL:VALUES, in case anything changes.

;; * * * * * * *

;; Architectural note: This function assumes that it is called by an unwind-protect from \miscn.ufn. Therefore, it skips two frames before deciding ;; whether to pass back one valur or many.

```
    (\VALUES-UFN (for I from 0 to (LLSH (SUB1 CL::ARGCOUNT)
                                       1)
                    by 2 collect (\GETBASEPTR CL::ARG-PTR I))
           (AND (IGEQ CL::ARGCOUNT 1)
                (\GETBASEPTR CL::ARG-PTR 0))
           (fetch (FX CLINK) of (fetch (FX CLINK) of (\MYALINK)))
           CL::ARG-PTR])
```

(CL::VALUES-LIST-UFN

```
  [LAMBDA (CL::INDEX CL::ARGCOUNT CL::ARG-PTR)                    ; Edited 5-Jun-90 15:21 by jds
```

;; This is the UFN for the VALUES-LIST MISCN opcode. Its definition must be analogous to that for CL:VALUES-LIST, in case anything changes.

;; * * * * * * *

;; Architectural note: This function assumes that it is called by an unwind-protect from \miscn.ufn. Therefore, it skips two frames before deciding ;; whether to pass back one value or many.

```
    (LET ((CL:VALUES (\GETBASEPTR CL::ARG-PTR 0)))
         (\VALUES-UFN CL:VALUES (CAR CL:VALUES)
                (fetch (FX CLINK) of (fetch (FX CLINK) of (\MYALINK)))
                CL::ARG-PTR])

)

(PUTPROPS LLMVS FILETYPE :FAKE-COMPILE-FILE)

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

(ADDTOVAR NLAMA )

(ADDTOVAR NLAML )

(ADDTOVAR LAMA CL:VALUES)
)

(PUTPROPS LLMVS COPYRIGHT ("Xerox Corporation" 1986 1987 1989 1990 1991 1993))
```

FUNCTION INDEX

MACRO INDEX

PROPERTY INDEX

CONSTANT INDEX