File created: 16-May-90 11:46:37 **{DSK}<usr>local>lde>lispcore>sources>ADDARITH.;2**

changes to: (VARS ADDARITHCOMS)

previous date: 30-Mar-89 11:13:59 {DSK}<usr>local>lde>lispcore>sources>ADDARITH.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

;;


```
(RPAQQ ADDARITHCOMS
       ((LOCALVARS . T)
                                                          ; OK
        (MACROS MASK.1'S MASK.0'S BITTEST BITSET BITCLEAR)
        (COMS (OPTIMIZERS LOGNOT)
              (FNS LOGNOT))
        (COMS                                             ; BYTE hacking functions
              (MACROS LOADBYTE DEPOSITBYTE)
              (MACROS BYTESIZE BYTEPOSITION))
        (COMS (OPTIMIZERS IMOD)
              (FNS IMODLESSP)
              (MACROS IMODPLUS IMODDIFFERENCE))
        (COMS (FNS ROT)
              (MACROS .ROT.))
        (COMS ;; Primitive Functions for extracting fields as integers
              (MACROS \XLOADBYTEWORD)
              (FNS \PUTBASEBITS)
              ;; Primitive functions, especially needed for CommonLisp array package.
              (DECLARE%: DONTCOPY (MACROS .HIHALFWORDLO. .HIHALFWORDHI. .LOHALFWORDLO. .LOHALFWORDHI.)))
        (COMS ;; Beginning of rewrite of some LLARITH things, modularly using the macros of this file
              (DECLARE%: DONTCOPY (EXPORT (CONSTANTS MASK0WORD1'S MASK1WORD0'S MASKWORD1'S MASKHALFWORD1'S
                                                     BITSPERHALFWORD)
                                          (MACROS EQZEROP)
                                          (MACROS \MOVETOBOX .XUNBOX. .XLLSH. .XLLSH1. .XLRSH.
                                                  .ADD.2WORD.INTEGERS. .SUB.2WORD.INTEGERS. .32BITMUL.)
                                          (MACROS .SUMSMALLMOD. .DIFFERENCESMALLMOD.)
                                          (MACROS \GETBASENIBBLE \PUTBASENIBBLE \GETBASEBIT \PUTBASEBIT))
                     (MACROS .ADD.2WORD.INTEGERS. .SUB.2WORD.INTEGERS. .32BITMUL.)))
        (PROP (MAKEFILE-ENVIRONMENT FILETYPE)
              ADDARITH)))

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(LOCALVARS . T)
)

;; OK

(DECLARE%: EVAL@COMPILE

(PUTPROPS MASK.1'S MACRO (OPENLAMBDA (POSITION SIZE)
                           (LSH (SUB1 (LSH 1 SIZE))
                                POSITION)))

(PUTPROPS MASK.0'S MACRO (OPENLAMBDA (POSITION SIZE)
                           (LOGNOT (MASK.1'S POSITION SIZE))))

(PUTPROPS BITTEST MACRO ((N MASK)
                         (NEQ 0 (LOGAND N MASK))))

(PUTPROPS BITSET MACRO (= . LOGOR))

(PUTPROPS BITCLEAR MACRO ((X MASK)
                          (LOGAND X (LOGNOT MASK))))
)


(DEFOPTIMIZER LOGNOT (INTEGER)
                     `(LOGXOR -1 ,INTEGER))

(DEFINEQ

(LOGNOT
  [LAMBDA (INTEGER)                                       (* kbr%: "12-Jul-86 17:05")
    (LOGXOR -1 INTEGER])

)
```

```
;; BYTE hacking functions

(DECLARE%: EVAL@COMPILE

(PUTPROPS LOADBYTE MACRO ((N POS SIZE)
                          (LOGAND (LSH N (IMINUS POS))
                                  (MASK.1'S 0 SIZE))))

(PUTPROPS DEPOSITBYTE MACRO (OPENLAMBDA (N POS SIZE VAL)
                              (LOGOR (BITCLEAR N (MASK.1'S POS SIZE))
                                     (LSH (LOGAND VAL (MASK.1'S 0 SIZE))
                                          POS))))
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS BYTESIZE MACRO ((BYTESPEC)
                          (BYTE-SIZE BYTESPEC)))

(PUTPROPS BYTEPOSITION MACRO ((BYTESPEC)
                              (CL:BYTE-POSITION BYTESPEC)))
)


(DEFOPTIMIZER IMOD (&REST L)
                   [PROG [(N (CONSTANTEXPRESSIONP (CADR L]
                         (if (NULL N)
                             then (RETURN 'IGNOREMACRO))
                         (SETQ N (CAR N))
                         (RETURN (COND
                                    ((NOT (POWEROFTWOP N))
                                     'IGNOREMACRO)
                                    (T (LIST 'LOGAND (CAR L)
                                             (SUB1 N])

(DEFINEQ

(IMODLESSP
  [LAMBDA (X Y MODULUS)                                    (* lmm "12-Apr-85 12:43")
    (ILESSP (IMODDIFFERENCE Y X MODULUS)
            (FOLDHI MODULUS 2])

)

(DECLARE%: EVAL@COMPILE

(PUTPROPS IMODPLUS MACRO ((X Y MODULUS)
                          (IMOD (IPLUS X Y)
                                MODULUS)))

(PUTPROPS IMODDIFFERENCE MACRO ((X Y MODULUS)
                                (IMOD (IDIFFERENCE X Y)
                                      MODULUS)))
)

(DEFINEQ

(ROT
  [LAMBDA (X N FIELDSIZE)                                  (* Pavel " 7-Oct-86 15:26")
    ;; Normalize N, the shift factor, into the half-open interval of 0 to FIELDSIZE and transform a negative N (rotating rightwards) into a positive form.
    (LET* ((N (IMOD N FIELDSIZE))
           (N.B (IDIFFERENCE FIELDSIZE N)))
          (DEPOSITBYTE (LOADBYTE X N.B N)
                 N N.B X])

)

(DECLARE%: EVAL@COMPILE

(PUTPROPS .ROT. MACRO ((XFORM N FIELDSIZE)
                       ((OPENLAMBDA (X)
                          (DEPOSITBYTE (LOADBYTE X (IDIFFERENCE FIELDSIZE N)
                                                 N)
                                 N
                                 (IDIFFERENCE FIELDSIZE N)
                                 X))
                        XFORM)))
)

;; Primitive Functions for extracting fields as integers

(DECLARE%: EVAL@COMPILE

(PUTPROPS \XLOADBYTEWORD DMACRO [(N POS SIZE)               ; N is constrained to be a SMALLP
                                 (LOGAND (\XLRSHWORD N POS)
```

```
                                        (MASK.1'S 0 (IMIN BITSPERWORD SIZE])
)

(DEFINEQ

(\PUTBASEBITS
  [LAMBDA (ADDR POSITION SIZE VAL)                                    (* lmm "12-Apr-85 15:18")
    (if (GREATERP POSITION BITSPERWORD)
        then (\PUTBASEBITS (\ADDBASE ADDR (FOLDLO POSITION BITSPERWORD))
                    (IMOD POSITION BITSPERWORD)
                    SIZE VAL)
      elseif (GREATERP SIZE (DIFFERENCE BITSPERWORD POSITION))
        then                                                          ; more than one word
             [\PUTBASEBITS ADDR POSITION (DIFFERENCE BITSPERWORD POSITION)
                    (RSH VAL (SETQ SIZE (DIFFERENCE SIZE (DIFFERENCE BITSPERWORD POSITION]
             (\PUTBASEBITS (\ADDBASE ADDR 1)
                    0 SIZE VAL)
      else                                                            ; a single word
           (\PUTBASE ADDR 0 (DEPOSITBYTE (\GETBASE ADDR 0)
                                 (DIFFERENCE (SUB1 BITSPERWORD)
                                       POSITION)
                                 SIZE VAL])

)

;; Primitive functions, especially needed for CommonLisp array package.

(DECLARE%: DONTCOPY

(DECLARE%: EVAL@COMPILE

(PUTPROPS .HIHALFWORDLO. MACRO ((X)
                                 (LRSH X BITSPERHALFWORD)))

(PUTPROPS .HIHALFWORDHI. MACRO [(X)
                                (LOGAND X (CONSTANT (LSH MASKHALFWORD1'S BITSPERHALFWORD])

(PUTPROPS .LOHALFWORDLO. MACRO ((X)
                                 (LOGAND X MASKHALFWORD1'S)))

(PUTPROPS .LOHALFWORDHI. MACRO ((X)
                                 (LLSH (LOGAND X MASKHALFWORD1'S)
                                       BITSPERHALFWORD)))
)
)

;; Beginning of rewrite of some LLARITH things, modularly using the macros of this file

(DECLARE%: DONTCOPY

;; FOLLOWING DEFINITIONS EXPORTED

(DECLARE%: EVAL@COMPILE

(RPAQQ MASK0WORD1'S 32767)

(RPAQQ MASK1WORD0'S 32768)

(RPAQQ MASKWORD1'S 65535)

(RPAQQ MASKHALFWORD1'S 255)

(RPAQQ BITSPERHALFWORD 8)

(CONSTANTS MASK0WORD1'S MASK1WORD0'S MASKWORD1'S MASKHALFWORD1'S BITSPERHALFWORD)
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS EQZEROP MACRO ((X)
                         (EQ 0 X)))
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS \MOVETOBOX DMACRO (OPENLAMBDA (N D)
                              (SELECTC (NTYPX N)
                                   (\SMALLP (replace (FIXP HINUM) of D with 0)
                                            (replace (FIXP LONUM) of D with N))
                                   (\FIXP (replace (FIXP HINUM) of D with (fetch (FIXP HINUM) of N))
                                          (replace (FIXP LONUM) of D with (fetch (FIXP LONUM) of N)))
                                   (\ILLEGAL.ARG N))))

(PUTPROPS .XUNBOX. MACRO [(X HX LX)
                          (until (SETQ LX (SELECTC (NTYPX X)
                                                   (\SMALLP (COND
```

```
                                                ((IGEQ X 0)
                                                 (SETQ HX 0)
                                                 X)
                                                (T (SETQ HX MASKWORD1'S)
                                                   (\LOLOC X))))
                                        (\FIXP (SETQ HX (fetch (FIXP HINUM) of X))
                                               (fetch (FIXP LONUM) of X))
                                        NIL))
                                do (SETQ X (LISPERROR "ILLEGAL ARG" X T])

(PUTPROPS .XLLSH. MACRO [(HI LO N)
                         (if (IGEQ N BITSPERWORD)
                             then                                    ; Jump 16 bits in a single bound!
                                  (SETQ HI LO)
                                  (SETQ LO 0)
                                  (SETQ N (IDIFFERENCE N BITSPERWORD)))
                         (if (IGEQ N BITSPERHALFWORD)
                             then                                    ; Jump 8 bits in a single bound!
                                  (SETQ HI (LOGOR (.LOHALFWORDHI. HI)
                                                  (.HIHALFWORDLO. LO)))
                                  (SETQ LO (.LOHALFWORDHI. LO))
                                  (SETQ N (IDIFFERENCE N BITSPERHALFWORD)))
                         (if (IGEQ N 4)
                             then                                    ; Jump 4 bits in a single bound!
                                  (SETQ HI (LOGOR (LRSH LO (CONSTANT (IDIFFERENCE BITSPERWORD 4)))
                                                  (LLSH [LOGAND HI (CONSTANT (MASK.1'S 0 (IDIFFERENCE
                                                                                         BITSPERWORD 4]
                                                                4)))
                                  (SETQ LO (LLSH [LOGAND LO (CONSTANT (MASK.1'S 0 (IDIFFERENCE BITSPERWORD 4]
                                                 4))
                                  (SETQ N (IDIFFERENCE N 4)))    ; MASK0WORD1'S should be same as (SUB1 (LSH 1 (SUB1
                                                                 ; BITSPERWORD)))
                         (FRPTQ N (SETQ HI (LLSH (LOGAND HI MASK0WORD1'S)
                                                 1))
                                  (SETQ LO (LLSH (if (IGEQ LO MASK1WORD0'S)
                                                     then (add HI 1)
                                                          (LOGAND LO MASK0WORD1'S)
                                                     else LO)
                                                 1])

(PUTPROPS .XLLSH1. MACRO ((HI LO)
                          (SETQ HI (LLSH (LOGAND HI MASK0WORD1'S)
                                         1))
                          (SETQ LO (LSH (COND
                                           ((IGEQ LO MASK1WORD0'S)
                                            (SETQ HI (LOGOR HI 1))
                                            (LOGAND LO MASK0WORD1'S))
                                           (T LO))
                                        1))))

(PUTPROPS .XLRSH. MACRO [(HI LO N)
                         (if (IGEQ N BITSPERWORD)
                             then                                    ; Jump 10 bits in a single bound!
                                  (SETQ LO HI)
                                  (SETQ HI 0)
                                  (SETQ N (IDIFFERENCE N BITSPERWORD)))
                         (if (IGEQ N BITSPERHALFWORD)
                             then                                    ; Jump 8 bits in a single bound!
                                  (SETQ LO (LOGOR (.HIHALFWORDLO. LO)
                                                  (.LOHALFWORDHI. HI)))
                                  (SETQ HI (.HIHALFWORDLO. HI))
                                  (SETQ N (IDIFFERENCE N BITSPERHALFWORD)))
                         (if (IGEQ N 4)
                             then                                    ; Jump 4 bits in a single bound!
                                  (SETQ LO (LOGOR (LLSH (LOGAND HI (CONSTANT (MASK.1'S 0 4)))
                                                        (CONSTANT (IDIFFERENCE BITSPERWORD 4)))
                                                  (LRSH LO 4)))
                                  (SETQ HI (LRSH HI 4))
                                  (SETQ N (IDIFFERENCE N 4)))    ; MASK1WORD0'S should be same as \SIGNBIT
                         (FRPTQ N (SETQ LO (if (ODDP HI)
                                               then (LOGOR (LRSH LO 1)
                                                           MASK1WORD0'S)
                                               else (LRSH LO 1)))
                                  (SETQ HI (LRSH HI 1])

(PUTPROPS .ADD.2WORD.INTEGERS. MACRO [(HX LX HY LY)              ; Ignores carry out of high-order word
                                      (SETQ HX (.SUMSMALLMOD. HX HY))
                                      (SETQ LX (.SUMSMALLMOD. LX LY (SETQ HX (if (EQ HX MAX.SMALL.INTEGER)
                                                                                 then 0
                                                                                 else (ADD1 HX])

(PUTPROPS .SUB.2WORD.INTEGERS. MACRO [(HX LX HY LY)              ; Ignores carry out of high-order word
                                      (SETQ HX (.DIFFERENCESMALLMOD. HX HY))
                                      (SETQ LX (.DIFFERENCESMALLMOD. LX LY (SETQ HX
                                                                                 (if (EQ HX 0)
                                                                                     then MAX.SMALL.INTEGER
                                                                                     else (SUB1 HX])
```

```
(PUTPROPS .32BITMUL. MACRO ((HR LR X Y)
                            (PROG (HX LX HY LY)
                                  (if (ILESSP X Y)
                                      then (swap X Y))          ; Y is the lesser of the two now
                                  (.XUNBOX. X HX LX)
                                  (.XUNBOX. Y HY LY)
                                LP (if (ODDP LY)
                                       then (.ADD.2WORD.INTEGERS. HR LR HX LX))
                                  (if (EQ HY 0)
                                      then (SETQ LY (LRSH LY 1))
                                           (if (EQ LY 0)
                                               then (RETURN))
                                       else (.LRSH1. HY LY))        ; Trim off highest bits, so that left-shifting doesn't generate FIXPs
                                  (SETQ HX (LOGAND HX MASK0WORD1'S))
                                  (.LLSH1. HX LX)
                                  (GO LP))))
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS .SUMSMALLMOD. MACRO ((X Y OVERFLOWFORM)
                               ([LAMBDA (\SumSmallModVar)
                                  (DECLARE (LOCALVARS \SumSmallModVar))
                                  (IF (ILEQ X \SumSmallModVar)
                                      THEN (IPLUS X Y)
                                    ELSE OVERFLOWFORM (IDIFFERENCE X (ADD1 \SumSmallModVar]
                                (IDIFFERENCE MAX.SMALL.INTEGER Y))))

(PUTPROPS .DIFFERENCESMALLMOD. MACRO [(X Y BORROWFORM)
                                      (IF (NOT (IGREATERP Y X))
                                          THEN (IDIFFERENCE X Y)
                                        ELSE BORROWFORM (ADD1 (IDIFFERENCE MAX.SMALL.INTEGER (IDIFFERENCE
                                                                                               Y X])
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS \GETBASENIBBLE DMACRO [OPENLAMBDA (BASE OFFST)
                                  ([LAMBDA (\Byte)
                                     (DECLARE (LOCALVARS \Byte))
                                     (if (ODDP OFFST)
                                         then (LOGAND \Byte (CONSTANT (MASK.1'S 0 BITSPERNIBBLE)))
                                      else (LRSH \Byte BITSPERNIBBLE]
                                   (\GETBASEBYTE BASE (FOLDLO OFFST NIBBLESPERBYTE])

(PUTPROPS \PUTBASENIBBLE DMACRO (OPENLAMBDA (BASE OFFST VAL)
                                  ([LAMBDA (\ByteNo)
                                     (DECLARE (LOCALVARS \ByteNo))
                                     ([LAMBDA (\Byte)
                                        (DECLARE (LOCALVARS \Byte))
                                        (\PUTBASEBYTE BASE \ByteNo
                                                (if (ODDP OFFST)
                                                    then (LOGOR (LOGAND \Byte (CONSTANT (MASK.1'S BITSPERNIBBLE
                                                                                                  BITSPERNIBBLE)))
                                                                VAL)
                                                  else (LOGOR (LOGAND \Byte (CONSTANT (MASK.1'S 0 BITSPERNIBBLE)
                                                                                      ))
                                                              (LLSH VAL BITSPERNIBBLE]
                                        (\GETBASEBYTE BASE \ByteNo]
                                   (FOLDLO OFFST NIBBLESPERBYTE))))

(PUTPROPS \GETBASEBIT DMACRO (OPENLAMBDA (BASE OFFST)
                               ([LAMBDA (\ByteNo \BitMask)
                                  (DECLARE (LOCALVARS \ByteNo \BitMask))
                                  (if (EQ 0 (LOGAND \BitMask (\GETBASEBYTE BASE \ByteNo)))
                                      then 0
                                    else 1]
                                (FOLDLO OFFST BITSPERBYTE)
                                (MASK.1'S (IDIFFERENCE (CONSTANT (SUB1 BITSPERBYTE))
                                                       (IMOD OFFST BITSPERBYTE))
                                      1))))

(PUTPROPS \PUTBASEBIT DMACRO (OPENLAMBDA (BASE OFFST VAL)
                               ([LAMBDA (\ByteNo \BitMask \Byte)
                                  (DECLARE (LOCALVARS \ByteNo \BitMask \Byte))
                                  (SETQ \Byte (\GETBASEBYTE BASE \ByteNo))
                                  (if (if (EQ 0 (LOGAND \BitMask \Byte))
                                          then (NOT (EQ 0 VAL))
                                       else (EQ 0 VAL))
                                      then (\PUTBASEBYTE BASE \ByteNo (LOGXOR \BitMask \Byte)))
                                  VAL]
                                (FOLDLO OFFST BITSPERBYTE)
                                (MASK.1'S (IDIFFERENCE (CONSTANT (SUB1 BITSPERBYTE))
                                                       (IMOD OFFST BITSPERBYTE))
                                      1))))
)
```

;; END EXPORTED DEFINITIONS

```
(DECLARE%: EVAL@COMPILE

(PUTPROPS .ADD.2WORD.INTEGERS. MACRO [(HX LX HY LY)            ; Ignores carry out of high-order word
                                      (SETQ HX (.SUMSMALLMOD. HX HY))
                                      (SETQ LX (.SUMSMALLMOD. LX LY (SETQ HX (if (EQ HX MAX.SMALL.INTEGER)
                                                                                 then 0
                                                                               else (ADD1 HX])

(PUTPROPS .SUB.2WORD.INTEGERS. MACRO [(HX LX HY LY)            ; Ignores carry out of high-order word
                                      (SETQ HX (.DIFFERENCESMALLMOD. HX HY))
                                      (SETQ LX (.DIFFERENCESMALLMOD. LX LY (SETQ HX
                                                                                 (if (EQ HX 0)
                                                                                     then MAX.SMALL.INTEGER
                                                                                   else (SUB1 HX])

(PUTPROPS .32BITMUL. MACRO ((HR LR X Y)
                            (PROG (HX LX HY LY)
                                  (if (ILESSP X Y)
                                      then (swap X Y))          ; Y is the lesser of the two now
                                  (.XUNBOX. X HX LX)
                                  (.XUNBOX. Y HY LY)
                                    LP  (if (ODDP LY)
                                            then (.ADD.2WORD.INTEGERS. HR LR HX LX))
                                  (if (EQ HY 0)
                                      then (SETQ LY (LRSH LY 1))
                                           (if (EQ LY 0)
                                               then (RETURN))
                                    else (.LRSH1. HY LY))       ; Trim off highest bits, so that left-shifting doesn't generate FIXPs
                                  (SETQ HX (LOGAND HX MASK0WORD1'S))
                                  (.LLSH1. HX LX)
                                  (GO LP))))
)
)

(PUTPROPS ADDARITH MAKEFILE-ENVIRONMENT (:PACKAGE "INTERLISP" :READTABLE "INTERLISP" :BASE 10))

(PUTPROPS ADDARITH FILETYPE CL:COMPILE-FILE)

(PUTPROPS ADDARITH COPYRIGHT ("Venue & Xerox Corporation" 1982 1983 1984 1985 1986 1987 1989 1990))
```

FUNCTION INDEX

MACRO INDEX

CONSTANT INDEX

OPTIMIZER INDEX

PROPERTY INDEX