File created: 16-May-90 20:28:24 **{DSK}<usr>local>lde>lispcore>sources>MACROS.;2**

changes to: (VARS MACROSCOMS)

previous date: 17-Feb-88 14:13:34 {DSK}<usr>local>lde>lispcore>sources>MACROS.;1

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
;;
```


```
(RPAQQ MACROSCOMS
       [(OPTIMIZERS ADD1 CONSTANT DEFERREDCONSTANT EVENP GEQ IGEQ ILEQ IMAX IMIN LEQ LIST* NCONC1 NEQ NLISTP
              ODDP RPTQ SELECT SELECTC SETQQ SUB1 ZEROP)
        (PROP MACRO RESETBUFS FLESSP PROG2 SIGNED UNSIGNED)
        (COMS                                                    ; obsolete Interlisp macro functions
              (FNS EXPANDMACRO MACROEXPANSION EXPAND-DEFMACRO COMPUTE-MACRO-ARGS MACROS.GETDEF GETMACROPROP
                   EXPANDOPENLAMBDA)
              (GLOBALVARS NOFIXFNSLST BYTECOMPFLG CLISPARRAY BYTEMACROPROP))
        (PROP MACRO LOADTIMECONSTANT)
        (FUNCTIONS CSELECT)
        (COMS (FNS PRINTCOMSTRAN)
              (GLOBALVARS COMMENTFLG LCASEFLG PRINTOUTMACROS)
              (ADDVARS (PRINTOUTMACROS))
              (VARS PRINTOUTTOKENS)
              (PROP INFO PRINTOUT printout)
              (PROP MACRO PRINTOUT printout))
        (ADDVARS * (LIST (CONS 'SYSPROPS MACROPROPS)))
        (PROP PROPTYPE * (PROGN MACROPROPS))
        (PROP SETFN GETTOPVAL)
        (PROP FILETYPE MACROS)
        (DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA)
                                                                              (NLAML)
                                                                              (LAMA])

(DEFOPTIMIZER ADD1 (X)
              `(IPLUS ,X 1))

(DEFOPTIMIZER CONSTANT (&REST MACROX)
              [PROG ((VAL (APPLY 'PROG1 MACROX)))
                    (RETURN (COND
                                ((CONSTANTOK VAL)
                                 (KWOTE VAL))
                                (T (CONS 'LOADTIMECONSTANT MACROX])

(DEFOPTIMIZER DEFERREDCONSTANT (X)
                              `((CL:LAMBDA (MACROX)
                                       (DECLARE (LOCALVARS MACROX))
                                       (OR (CDR MACROX)
                                           (FRPLACD (FRPLACA MACROX (EVQ ,X))
                                                    T))
                                       (CAR MACROX))
                                (LOADTIMECONSTANT (CONS NIL NIL))))

(DEFOPTIMIZER EVENP (N &OPTIONAL (MODULUS 2))
              `(EQ 0 (IMOD ,N ,MODULUS)))

(DEFOPTIMIZER GEQ (X Y)
              `(NOT (LESSP ,X ,Y)))

(DEFOPTIMIZER IGEQ (X Y)
              `(NOT (ILESSP ,X ,Y)))

(DEFOPTIMIZER ILEQ (X Y)
              `(NOT (IGREATERP ,X ,Y)))

(DEFOPTIMIZER IMAX (&OPTIONAL (ARG1 NIL ARG1GIVEN)
                              (ARG2 NIL ARG2GIVEN)
                              &REST RESTARGS)
              [COND
                 ((NOT ARG1GIVEN)
                  'MIN.INTEGER)
                 ((NOT ARG2GIVEN)
                  `(FIX %, ARG1))
```

```
                    (RESTARGS `(IMAX (IMAX2 %, ARG1 %, ARG2)
                                     ., RESTARGS))
                    (T `(IMAX2 %, ARG1 %, ARG2])


(DEFOPTIMIZER IMIN (&OPTIONAL (ARG1 NIL ARG1GIVEN)
                              (ARG2 NIL ARG2GIVEN)
                              &REST RESTARGS)
                   (COND
                      ((NOT ARG1GIVEN)
                       'MAX.INTEGER)
                      ((NOT ARG2GIVEN)
                       `(FIX %, ARG1))
                      (RESTARGS `(IMIN (IMIN2 %, ARG1 %, ARG2)
                                       ., RESTARGS))
                      (T (LIST 'IMIN2 ARG1 ARG2))))


(DEFOPTIMIZER LEQ (X Y)
                  `(NOT (GREATERP ,X ,Y)))


(DEFOPTIMIZER LIST* (&REST X)
                    [COND
                       ((NULL X)
                        NIL)
                       ((NULL (CDR X))
                        (CAR X))
                       ((NULL (CDDR X))
                        (CONS 'CONS X))
                       (T (LIST 'CONS (CAR X)
                                (CONS 'LIST* (CDR X])


(DEFOPTIMIZER NCONC1 (LST X)
                     `(NCONC ,LST (CONS ,X)))


(DEFOPTIMIZER NEQ (X Y)
                  `(NULL (EQ ,X ,Y)))


(DEFOPTIMIZER NLISTP (X)
                     `(NULL (LISTP ,X)))


(DEFOPTIMIZER ODDP (X . TAIL)
                   `(NOT (EVENP ,X \, TAIL)))


(DEFOPTIMIZER RPTQ (N . FORMS)
                   `(PROG ((RPTN ,N)
                           RPTV)
                          (DECLARE (LOCALVARS RPTN RPTV))
                    RPTQLAB
                          (COND
                             ((IGREATERP RPTN 0)
                              (SETQ RPTV (PROGN \, FORMS))
                              (SETQ RPTN (SUB1 RPTN))
                              (GO RPTQLAB)))
                          (RETURN RPTV)))


(DEFOPTIMIZER SELECT (&REST X)
                     (CSELECT X))


(DEFOPTIMIZER SELECTC (EXPR &REST CLAUSES)
                      `[SELECTQ ,EXPR
                           ,@(FOR TAIL ON CLAUSES COLLECT (CL:IF (CDR TAIL)
                                                                 `(,(EVAL (CAAR TAIL))
                                                                   ,@(CDAR TAIL))
                                                                 (CAR TAIL))])


(DEFOPTIMIZER SETQQ (X V)
                    `(SETQ ,X ',V))


(DEFOPTIMIZER SUB1 (X)
                   `(IDIFFERENCE ,X 1))


(DEFOPTIMIZER ZEROP (&REST ARGS)
                    (CONS '[OPENLAMBDA (X)
                              (COND
                                 ((EQ X 0))
```

```
                                ((FLOATP X)
                                 (\FZEROP X]
                            ARGS))

(PUTPROPS RESETBUFS MACRO [(A . B)
                           ([LAMBDA ($$BUFS)
                              (DECLARE (LOCALVARS $$BUFS))
                              (PROG1 (PROGN A . B)
                                  (AND $$BUFS (BKBUFS $$BUFS)))]
                            (PROGN (LINBUF)
                                   (SYSBUF)
                                   (CLBUFS NIL T READBUF])

(PUTPROPS FLESSP MACRO [LAMBDA (X Y)
                          (FGREATERP Y X])

(PUTPROPS PROG2 MACRO ((X . Y)
                       (PROGN X (PROG1 . Y))))

(PUTPROPS SIGNED MACRO [ARGS (COND
                              ((EQ COMPILE.CONTEXT 'EFFECT)
                               (CAR ARGS))
                              (T (CONS '(OPENLAMBDA (N WIDTH)
                                                    (COND
                                                       [[IGREATERP N (SUB1 (LLSH 1 (SUB1 WIDTH]
                                                                                  (* done this way just so that (SIGNED X |2^16|) doesn't box)
                                                        (SUB1 (IDIFFERENCE N (SUB1 (LLSH 1 WIDTH]
                                                       (T N)))
                                       ARGS])

(PUTPROPS UNSIGNED MACRO [(X WIDTH)
                          (LOGAND X (SUB1 (LLSH 1 WIDTH])
```

;; obsolete Interlisp macro functions

```
(DEFINEQ
```

(EXPANDMACRO

```
  [LAMBDA (EXP QUIETFLG OPTIONS COMPILE.CONTEXT)                (* Pavel "24-Oct-86 23:44")
    (DECLARE (SPECVARS NCF PCF VCF EFF EXP COMPILE.CONTEXT))
    (PROG [ALLFLG MACRODEF NCF PCF (VCF (NEQ COMPILE.CONTEXT 'EFFECT))
                  (EFF (EQ COMPILE.CONTEXT 'EFFECT]
      LP  (COND
             ((NLISTP EXP)
              (GO OUT))
             ((AND (EQ ALLFLG 'CLISP)
                   (GETHASH EXP CLISPARRAY))
              (SETQ EXP (GETHASH EXP CLISPARRAY))
              (GO LP)))
      MLP (SETQ MACRODEF (GETMACROPROP (CAR EXP)
                                       COMPILERMACROPROPS))
          [COND
             ((NEQ EXP (SETQ EXP (MACROEXPANSION EXP MACRODEF)))
              (COND
                 (ALLFLG (GO LP]
      OUT (COND
             (QUIETFLG (RETURN EXP))
             (T (RESETFORM (OUTPUT T)
                       (PRINTDEF EXP NIL T)
                       (TERPRI T])
```

(MACROEXPANSION

```
  [LAMBDA (EXPR MACRODEF COMPFLG COMPILE.CONTEXT)               ; Edited 17-Feb-88 14:10 by amd
    (DECLARE (SPECVARS COMPILE.CONTEXT))
    (COND
       ((NLISTP MACRODEF)
        EXPR)
       (T (SELECTQ (CAR MACRODEF)
                   (NIL (COND
                           ((CDDR MACRODEF)
                            (CONS 'PROGN (CDR MACRODEF)))
                           (T (CADR MACRODEF))))
                   ([LAMBDA NLAMBDA]
                       (CONS MACRODEF (CDR EXPR)))
                   (=                                           (* bytemacro abbreviation)
                      (CONS (CDR MACRODEF)
                            (CDR EXPR)))
                   (OPENLAMBDA (EXPANDOPENLAMBDA MACRODEF (CDR EXPR)))
                   ((APPLY APPLY*)
                       EXPR)
                   (DEFMACRO (EXPAND-DEFMACRO (CDR MACRODEF)
                                              EXPR))
                   (COND
                      [(LISTP (CAR MACRODEF))
                       (SUBPAIR (CAR MACRODEF)
                                (CDR EXPR)
```

```
                        (COND
                           ((CDDR MACRODEF)
                            (CONS 'PROGN (CDR MACRODEF)))
                           (T (CADR MACRODEF]
                    ((LITATOM (CAR MACRODEF))
                     (COND
                        ((FMEMB (CAR MACRODEF)
                                LAMBDASPLST)
                         (CONS MACRODEF (CDR EXPR)))
                        ((OR (EQ [SETQ MACRODEF
                                  (COND
                                     (COMPFLG (APPLY (CONS 'NLAMBDA MACRODEF)
                                                     (CDR EXPR)))
                                     (T (PROG ((EXP EXPR)
                                               (EFF (EQ COMPILE.CONTEXT 'EFFECT))
                                               (VCF (NEQ COMPILE.CONTEXT 'EFFECT))
                                              NCF PCF PREDF)
                                             (DECLARE (SPECVARS NCF PCF VCF EFF EXPR EXP RETF PREDF))
                                                                  (* various variables bound in the Interlisp-D and Interlisp-10
                                                                  compiler)
                                             (RETURN (APPLY (CONS 'NLAMBDA MACRODEF)
                                                            (CDR EXPR]
                                'IGNOREMACRO)
                             (EQ MACRODEF 'COMPILER:PASS))
                         (AND (EQ MACRODEF 'COMPILER:PASS)
                              (CL:WARN "Macroexpansion of ~S produced COMPILER:PASS. This should probably be an
                                    optimizer." (CAR EXPR)))
                         EXPR)
                        (T MACRODEF)))
                    (T EXPR])
```

(EXPAND-DEFMACRO

```
[CL:LAMBDA (DEF FORM &OPTIONAL DEFAULT-VALUE)                    (* lmm "25-May-86 00:15")
        (LET (*MACRO-VARS* *MACRO-VALS* (*MACRO-FORM* FORM)
              (*MACRO-DEFAULT* DEFAULT-VALUE))
             (DECLARE (CL:SPECIAL *MACRO-VARS* *MACRO-VALS* *MACRO-FORM* *MACRO-DEFAULT*))
             (COMPUTE-MACRO-ARGS (CAR DEF)
                    (CDR FORM)
                    NIL)
             (LET [(VAL (CL:PROGV *MACRO-VARS* *MACRO-VALS*
                                  (EVAL (CONS 'PROGN (CDR DEF))))]
                  (if (EQ VAL 'IGNOREMACRO)
                      then FORM
                    else VAL])
```

(COMPUTE-MACRO-ARGS

```
[CL:LAMBDA
 (ARGUMENT-LIST MACRO-CALL-BODY CONTEXT)                         (* lmm "18-Apr-86 12:04")
 (COND
    ((NULL ARGUMENT-LIST)
     NIL)
    ((CL:ATOM ARGUMENT-LIST)
     (SETQ *MACRO-VARS* (CONS ARGUMENT-LIST *MACRO-VARS*))
     (SETQ *MACRO-VALS* (CONS MACRO-CALL-BODY *MACRO-VALS*)))
    (T (SELECTQ (CAR ARGUMENT-LIST)
            ((&REST &BODY)
                 (COMPUTE-MACRO-ARGS (CADR ARGUMENT-LIST)
                        MACRO-CALL-BODY NIL)
                 (COMPUTE-MACRO-ARGS (CDDR ARGUMENT-LIST)
                        MACRO-CALL-BODY
                        'AUX-ONLY))
            (&WHOLE (COMPUTE-MACRO-ARGS (CADR ARGUMENT-LIST)
                           *MACRO-FORM*
                           'ONE)
                    (COMPUTE-MACRO-ARGS (CDDR ARGUMENT-LIST)
                           MACRO-CALL-BODY
                           'AUX-ONLY))
            (&ENVIRONMENT                                        (* dunno exactly what to do about this--
                                                                 there no environments here right now)
                          (COMPUTE-MACRO-ARGS (CADR ARGUMENT-LIST)
                                 NIL
                                 'ONE)
                          (COMPUTE-MACRO-ARGS (CDDR ARGUMENT-LIST)
                                 MACRO-CALL-BODY
                                 'AUX-ONLY))
            (&OPTIONAL (COMPUTE-MACRO-ARGS (CDR ARGUMENT-LIST)
                              MACRO-CALL-BODY
                              'OPTIONAL))
            (&AUX (COMPUTE-MACRO-ARGS (CDR ARGUMENT-LIST)
                         MACRO-CALL-BODY
                         'AUX))
            (&KEY (SETQ ARGUMENT-LIST (CDR ARGUMENT-LIST))
                  [while ARGUMENT-LIST
                     do (SELECTQ (CAR ARGUMENT-LIST)
                                 ((&REST &ALLOW-OTHER-KEYS &AUX)
```

```
                    (RETURN (COMPUTE-MACRO-ARGS ARGUMENT-LIST MACRO-CALL-BODY NIL)))
                (PROGN (LET* [(KEYWORD-VARIABLE (CAR ARGUMENT-LIST))
                              SUPPLIED-P-VARIABLE
                              [DEFAULT-VALUE (COND
                                                 ((CL:CONSP KEYWORD-VARIABLE)
                                                  (PROG1 (CADR KEYWORD-VARIABLE)
                                                      (SETQ SUPPLIED-P-VARIABLE (CADDR
                                                                                    KEYWORD-VARIABLE
                                                                                    ))
                                                      (SETQ KEYWORD-VARIABLE (CAR KEYWORD-VARIABLE)
                                                       )))]
                              [CL:KEYWORD (COND
                                             [(CL:CONSP KEYWORD-VARIABLE)
                                              (PROG1 (CAR KEYWORD-VARIABLE)
                                                  (SETQ KEYWORD-VARIABLE (CADR KEYWORD-VARIABLE)))
                                              ]
                                             (T (MAKE-KEYWORD KEYWORD-VARIABLE]
                              (FOUND-VALUE (for FM on MACRO-CALL-BODY by (CDDR FM)
                                              do (COND
                                                     ((EQ (CAR FM)
                                                          CL:KEYWORD)
                                                      (RETURN (CDR FM]
                             [COND
                                (SUPPLIED-P-VARIABLE (COMPUTE-MACRO-ARGS SUPPLIED-P-VARIABLE
                                                         (COND
                                                            (FOUND-VALUE T)
                                                            (T NIL))
                                                         'ONE]
                             (COMPUTE-MACRO-ARGS KEYWORD-VARIABLE (COND
                                                                     (FOUND-VALUE (CAR
                                                                                    FOUND-VALUE
                                                                                    ))
                                                                     (T (EVAL DEFAULT-VALUE)))
                                                 'ONE))
                            (pop ARGUMENT-LIST])
        (PROGN [COND
                  [(EQ CONTEXT 'OPTIONAL)
                   (COND
                      [(CL:CONSP (CAR ARGUMENT-LIST))       (* an optional of the form (arg init val))
                       (DESTRUCTURING-BIND (ARG INIT SUPPLIEDP)
                              (CAR ARGUMENT-LIST)
                              (COND
                                 ((CL:ATOM MACRO-CALL-BODY)
                                                            (* optional omitted)
                                  (AND SUPPLIEDP (COMPUTE-MACRO-ARGS SUPPLIEDP NIL 'ONE))
                                  (COMPUTE-MACRO-ARGS (CAAR ARGUMENT-LIST)
                                         (EVAL INIT)
                                         'ONE))
                                 (T                         (* optional present)
                                    [COND
                                       (SUPPLIEDP (COMPUTE-MACRO-ARGS SUPPLIEDP T 'ONE]
                                    (COMPUTE-MACRO-ARGS (CAAR ARGUMENT-LIST)
                                           (CAR MACRO-CALL-BODY)
                                           NIL]
                      (T (COND
                            ((CL:ATOM MACRO-CALL-BODY)      (* optional omitted)
                             (COMPUTE-MACRO-ARGS (CAR ARGUMENT-LIST)
                                    *MACRO-DEFAULT*
                                    'ONE))
                            (T                              (* optional present)
                               (COMPUTE-MACRO-ARGS (CAR ARGUMENT-LIST)
                                      (CAR MACRO-CALL-BODY)
                                      NIL]
                  [(EQ CONTEXT 'AUX)
                   (for BINDING in ARGUMENT-LIST do (COND
                                                       ((CL:CONSP BINDING)
                                                        (COMPUTE-MACRO-ARGS (CAR BINDING)
                                                               (EVAL (CADR BINDING))
                                                               'ONE))
                                                       (T (COMPUTE-MACRO-ARGS BINDING NIL 'ONE]
                  (T (COND
                        ((CL:ATOM MACRO-CALL-BODY)
                         (ERROR "macro body missing value for" ARGUMENT-LIST))
                        (T (COMPUTE-MACRO-ARGS (CAR ARGUMENT-LIST)
                                  (CAR MACRO-CALL-BODY)
                                  NIL]
               (COMPUTE-MACRO-ARGS (CDR ARGUMENT-LIST)
                      (CDR MACRO-CALL-BODY)
                      CONTEXT])
```

(MACROS.GETDEF

```
  [LAMBDA (NAME TYPE OPTIONS)                                   (* lmm " 2-Apr-85 17:26")
    (MKPROGN (for X on (GETPROPLIST NAME) by (CDDR X) when (FMEMB (CAR X)
                                                                  MACROPROPS)
                collect (if (AND (EQ (CAR X)
                                     'MACRO)
```

```
                            (EQ (CAADR X)
                                'DEFMACRO))
                       then `(DEFMACRO %, NAME
                                %,@
                                (CDR (CADR X)))
                      else `(PUTPROPS %,
                                      NAME %,
                                      (CAR X) %,
                                      (CADR X))])
```

(GETMACROPROP

```
  [LAMBDA (FN PROPS)                                                    (* lmm "18-APR-82 13:23")
    (for X in PROPS bind VAL do (COND
                                   ((SETQ VAL (GETPROP FN X))
                                    (RETURN VAL])
```

(EXPANDOPENLAMBDA

```
  [LAMBDA (OPENLAM ACTUALS)                                             (* lmm "27-FEB-83 23:26")
    (PROG ((FORMALS (CADR OPENLAM))
           A ARGS VALS SUBSTPAIRS VAL GENARGS TMP)
      LP   (COND
              ((NLISTP FORMALS)
               (GO OUT)))
           (SETQ A (CAR FORMALS))
           (COND
              ((NLISTP ACTUALS)                                         (* Here if ran out of actuals before formals)
               (for A in FORMALS do (push SUBSTPAIRS (LIST A)))
               (GO OUT)))
           (SETQ VAL (CAR ACTUALS))
           (COND
              [(SETQ TMP (CONSTANTEXPRESSIONP VAL))
               (push SUBSTPAIRS (CONS A (KWOTE (CAR TMP]
              (T (push ARGS A)
                 (push VALS VAL)))
           (SETQ FORMALS (CDR FORMALS))
           (SETQ ACTUALS (CDR ACTUALS))
           (GO LP)
      OUT [while (AND VALS (ATOM (CAR VALS))) do (push SUBSTPAIRS (CONS (pop ARGS)
                                                                        (pop VALS]
           (SETQ OPENLAM (CDDR OPENLAM))
           [COND
              (SUBSTPAIRS [COND
                            (ARGS (SETQ OPENLAM (SUBPAIR ARGS [SETQ ARGS (MAPCAR ARGS (FUNCTION (LAMBDA (A)
                                                                                                 (PACK* A
                                                                                                        (GENSYM
                                                                                                         ]
                                                                                  OPENLAM]      (* Replace variables to avoid conflict with names in substituted values)
                        (SETQ OPENLAM (SUBLIS SUBSTPAIRS OPENLAM]   (* Any ACTUALS left are extra but still need to be evaluated)
           (RETURN (COND
                     (ARGS `([LAMBDA %,
                                (SETQ ARGS (REVERSE ARGS))
                                (DECLARE (LOCALVARS ., ARGS))
                                ., OPENLAM]
                           .,
                           (REVERSE VALS)
                           ., ACTUALS))
                     (T (MKPROGN (NCONC ACTUALS OPENLAM])

)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS NOFIXFNSLST BYTECOMPFLG CLISPARRAY BYTEMACROPROP)
)

(PUTPROPS LOADTIMECONSTANT MACRO ((X)
                                  (DEFERREDCONSTANT X)))
```

(CL:DEFUN CSELECT (L)

;;; The macro-expansion function for the optimizer on SELECT.

```
   [LET* [(SELECTOR (CAR L))
          (CLAUSES (CDR L))
          (COND-FORM `(COND
                         ,@(for TAIL on CLAUSES
                              collect (CL:IF (NULL (CDR TAIL))     ; The default clause
                                         `(T ,(CAR TAIL))
                                         [LET* [(CASES (CAR (CAR TAIL)))
                                                (ACTIONS (CDR (CAR TAIL)))
                                                [EQ-FORMS (FOR CASE IN (OR (LISTP CASES)
                                                                           (LIST CASES))
                                                                COLLECT `(EQ .SELEC. ,CASE]
```

```
                                          (TEST (CL:IF (NULL (CDR EQ-FORMS))
                                                    (CAR EQ-FORMS)
                                                    `(OR ,@EQ-FORMS))]
                                         `(,TEST ,@ACTIONS])]
        (CL:IF (NULL CLAUSES)
            SELECTOR
            `([LAMBDA (.SELEC.)
                (DECLARE (LOCALVARS .SELEC.))
                ,COND-FORM]
              ,SELECTOR))])

(DEFINEQ
```

(PRINTCOMSTRAN

```
  [LAMBDA (FORM TAIL MACROS FILEFORM FROMDWIM)                    (* lmm "10-Jan-86 13:55")
```

(* This function computes the translations for PRINTOUT type CLISP forms.
FORM is the form beginning with the CLISPWORD. After it is dwimified, TAIL is applied to obtain the TAIL of printing commands. If FILEFORM~=NIL, it is applied to FORM after dwimification to produce the output file specification.)

```
    (PROG [FORMATLIST (VARS (AND FROMDWIM (APPEND (MAPCAR MACROS (FUNCTION CAR))
                                                  PRINTOUTTOKENS VARS]
          (DECLARE (SPECVARS VARS))
          [for ARG in (CDR FORM) bind (TYPE POINT WIDTH)
             when [AND (LITATOM ARG)
                       (NOT (FASSOC ARG FORMATLIST))
                       (EQ (CHCON1 ARG)
                           (CHARCODE %.))
                       (SELCHARQ (SETQ TYPE (NTHCHARCODE ARG 2))
                            ((I F)
                             T)
                            NIL)
                       (FIXP (SETQ WIDTH (SUBATOM ARG 3 (AND (SETQ POINT (STRPOS '%. ARG 3))
                                                             (SUB1 POINT]
             do (push VARS ARG)                                    (* Suppress spelling-correction of formatcode atoms)
                (push FORMATLIST (CONS ARG ''(%, (COND
                                                    ((EQ TYPE (CHARCODE I))
                                                     'FIX)
                                                    (T 'FLOAT))
                                                  %, WIDTH ., (while POINT
                                                                 collect (SUBATOM ARG (ADD1 POINT)
                                                                                  (AND (SETQ POINT
                                                                                         (STRPOS '%. ARG (ADD1 POINT))
                                                                                         )
                                                                                       (SUB1 POINT]
                                                                 (* Since we did all the work to decode the format, save it for later.)
          (AND FROMDWIM (DWIMIFY0? (CDR FORM)
                                   FORM NIL NIL NIL FAULTFN))
          [COND
             (FILEFORM (SETQ FILEFORM (LIST (COND
                                               ((EQ FILEFORM T)
                                                T)
                                               (T (APPLY* FILEFORM FORM]
          (SETQ TAIL (APPLY* TAIL FORM))
          (RETURN
           (while TAIL bind (ARG TEMP RESETOUT)
              collect [COND
                         ((SETQ TEMP (ASSOC (CAR TAIL)
                                            MACROS))
                          (SETQ RESETOUT T)
```

(* Probably should pass FILEFORM to macrofn, but then would have to explain interface, smashing etc.)

```
                          (SETQ TAIL (APPLY* (CADR TEMP)
                                             TAIL))
                          (pop TAIL))
                         (T (SELECTQ (SETQ ARG (pop TAIL))
                                (.TAB0 `(TAB %, (pop TAIL)
                                             0 ., FILEFORM))
                                (.TAB `(TAB %, (pop TAIL)
                                            NIL %,@ FILEFORM))
                                ((0 T)
                                     `(TERPRI %,@ FILEFORM))
                                (.RESET `(PRIN1 (CONSTANT (CHARACTER (CHARCODE CR)))
                                                %,@ FILEFORM))
                                (%# (SETQ RESETOUT T)
                                    (pop TAIL))
                                (.P2 `(PRIN2 %, (pop TAIL)
                                             %,@ FILEFORM))
                                ((.PPF .PPV .PPFTL .PPVTL)
                                     `(PRINTDEF %, (pop TAIL)
                                              (POSITION %,@ FILEFORM)
                                              %,
                                              (OR (EQ ARG '.PPF)
                                                  (EQ ARG '.PPFTL))
                                              %,
```

```
                                    (OR (EQ ARG '.PPVTL)
                                        (EQ ARG '.PPFTL))
                                    NIL %,@ FILEFORM))
                               (.FONT (SETQ ARG (pop TAIL))
                                      `(CHANGEFONT %, (COND
                                                          ((FIXP ARG)
                                                           (PACK* 'FONT ARG))
                                                          (T ARG))
                                                  %,@ FILEFORM))
                               ((.SUB .SUP .BASE)
                                    `(AND FONTCHANGEFLG (PROGN (CHANGEFONT SUPERSCRIPTFONT %,@ FILEFORM)
                                                               (PRIN3 %, (LIST 'QUOTE
                                                                               (SELECTQ ARG
                                                                                    (.SUB (CONSTANT (CHARACTER
                                                                                                     20)))
                                                                                    (.SUP (CONSTANT (CHARACTER
                                                                                                     8)))
                                                                                    (.BASE (CONSTANT (CHARACTER
                                                                                                      14)))
                                                                                    NIL))
                                                                         %,@ FILEFORM))))
                               (%, `(SPACES 1 %,@ FILEFORM))
                               (%,, `(SPACES 2 %,@ FILEFORM))
                               (%,,, `(SPACES 3 %,@ FILEFORM))
                               (.SP `(SPACES %, (pop TAIL)
                                            %,@ FILEFORM))
                               (.SKIP `(FRPTQ %, (pop TAIL)
                                              (TERPRI %,@ FILEFORM)))
                               (.N `(PRINTNUM %, (pop TAIL)
                                          %,
                                          (pop TAIL)
                                          %,@ FILEFORM))
                               ((.FR .FR2 .CENTER .CENTER2)
                                    `(FLUSHRIGHT %, (pop TAIL)
                                                 %,
                                                 (pop TAIL)
                                                 0 %, (SELECTQ ARG
                                                               ((.FR2 .CENTER2)
                                                                    T)
                                                               NIL)
                                                 %,
                                                 (SELECTQ ARG
                                                      ((.CENTER .CENTER2)
                                                           T)
                                                      NIL)
                                                 %,@ FILEFORM))
                               ((.PARA .PARA2)
                                    `(PRINTPARA %, (pop TAIL)
                                                %,
                                                (pop TAIL)
                                                %,
                                                (pop TAIL)
                                                %,
                                                (EQ ARG '.PARA2)
                                                NIL %,@ FILEFORM))
                               (.PAGE `(PROGN (PRIN3 %, (LIST 'QUOTE (CHARACTER (CHARCODE FORM)))
                                                     %,@ FILEFORM)
                                              (POSITION (PROGN %,@ FILEFORM)
                                                        0)))
                               (COND
                                  ((SETQ TEMP (CDR (FASSOC ARG FORMATLIST)))
                                   `(PRINTNUM %, TEMP %, (pop TAIL)
                                              %,@ FILEFORM))
                                  ((NOT (FIXP ARG))
                                   `(PRIN1 %, ARG %,@ FILEFORM))
                                  ((MINUSP ARG)
                                   `(SPACES %, (IMINUS ARG)
                                            %,@ FILEFORM))
                                  (T `(TAB %, ARG NIL %,@ FILEFORM]
                    finally (RETURN (COND
                               ((AND (CAR FILEFORM)
                                     RESETOUT)
                                `(RESETFORM (OUTPUT %, (PROG1 (CAR FILEFORM)
                                                              (RPLACA FILEFORM NIL)))
                                        %,@ $$VAL))
                               [(LISTP (CAR FILEFORM))
                                `([LAMBDA ($$OUTPUT)
                                    (DECLARE (LOCALVARS $$OUTPUT))
                                    %,@ $$VAL]
                                  %,
                                  (PROG1 (CAR FILEFORM)
                                      (RPLACA FILEFORM '$$OUTPUT))]
                               (T `(PROGN ., $$VAL])

)

(DECLARE%: DOEVAL@COMPILE DONTCOPY
```

```
(GLOBALVARS COMMENTFLG LCASEFLG PRINTOUTMACROS)
)

(ADDTOVAR PRINTOUTMACROS )

(RPAQQ PRINTOUTTOKENS (.RESET .TAB %# %, %,, %,,, .P2 .PPF .PPV .PPFTL .PPVTL .TAB0 .FR .FR2 .CENTER .CENTER2
                           .PARA .PARA2 .PAGE .FONT .SUP .SUB .BASE .SP .SKIP .N))

(PUTPROPS PRINTOUT INFO NOEVAL)

(PUTPROPS printout INFO NOEVAL)

(PUTPROPS PRINTOUT MACRO (DEFMACRO (&WHOLE X) (PRINTCOMSTRAN X (FUNCTION CDDR)
                                                              PRINTOUTMACROS
                                                              (FUNCTION CADR))))

(PUTPROPS printout MACRO (DEFMACRO (&WHOLE X) (PRINTCOMSTRAN X (FUNCTION CDDR)
                                                            PRINTOUTMACROS
                                                            (FUNCTION CADR))))

(ADDTOVAR SYSPROPS ALTOMACRO MACRO BYTEMACRO DMACRO)

(PUTPROPS ALTOMACRO PROPTYPE MACROS)

(PUTPROPS MACRO PROPTYPE MACROS)

(PUTPROPS BYTEMACRO PROPTYPE MACROS)

(PUTPROPS DMACRO PROPTYPE MACROS)

(PUTPROPS GETTOPVAL SETFN SETTOPVAL)

(PUTPROPS MACROS FILETYPE CL:COMPILE-FILE)

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

(ADDTOVAR NLAMA )

(ADDTOVAR NLAML )

(ADDTOVAR LAMA )
)

(PUTPROPS MACROS COPYRIGHT ("Venue & Xerox Corporation" 1984 1985 1986 1987 1988 1990))
```

FUNCTION INDEX

OPTIMIZER INDEX

MACRO INDEX

PROPERTY INDEX

VARIABLE INDEX