File created:	11-Jun-90 15:57:59 **{DSK}<usr>local>lde>lispcore>library>FX-80DRIVER.;2**
changes to:	(VARS FX-80DRIVERCOMS)
previous date:	23-Sep-88 10:26:48 {DSK}<usr>local>lde>lispcore>library>FX-80DRIVER.;1
Read Table:	INTERLISP
Package:	INTERLISP
Format:	XCCS

```
(RPAQQ FX-80DRIVERCOMS
       (
;;;; FX-80 driver

       (COMS * FX-80.FAST-DRIVERCOMS                           ; the fast driver
             )
       (COMS * FX-80.HIGH-QUALITY-DRIVERCOMS                   ; the  higher quality driver
             )
       (COMS * FX80-PRINTCOMS                                  ; FXPrinter emulation
             )
       (COMS                                                   ; common routines
             (FUNCTIONS                                        ; abort window stuff
                    WITH-ABORT-WINDOW \FX80.CREATE-SEND-ABORT-WINDOW)
             (FUNCTIONS                                        ; font profile hacking
                    \ADD-TO-FONTPROFILE \GET-FROM-FONTPROFILE))


;;;; initialization

             [COMS (DECLARE%: DONTEVAL@LOAD DOCOPY (P (\HQFX80.INIT)
                                                     (\FASTFX80.INIT]
             (PROP FILETYPE FX-80DRIVER)))


;;;; FX-80 driver

(RPAQQ FX-80.FAST-DRIVERCOMS
       [;; Fast driver
        ;;
        (STRUCTURES FASTFX80DATA)
        (FNS \FASTFX80.INIT)
        ;; Imagestream methods

        (COMS ;; opening/closing imagestream

              (COMS (FNS OPENFASTFX80STREAM)
                    (FUNCTIONS \FASTFX80.PREAMBLE \FASTFX80.RESET-PRINTER \FASTFX80.OUTPUT-SIGNATURE))
              (FNS \FASTFX80.CLOSE))
        (COMS ;; methods that hack fonts

              (FNS \FASTFX80.CHANGEFONT \FASTFX80.FONTCREATE \FASTFX80.CREATECHARSET)
              (FUNCTIONS \FASTFX80.INIT-FONT-PROFILE))
        (COMS ;; methods for measuring

              (FNS \FASTFX80.STRINGWIDTH \FASTFX80.CHARWIDTH \FASTFX80.SUBCHARWIDTH)
              (FUNCTIONS \FASTFX80.SPACEFACTOR))
        (COMS ;; methods that affect the current position/size of drawing surface

              (FNS \FASTFX80.CLIPPINGREGION \FASTFX80.MOVETO \FASTFX80.XPOSITION \FASTFX80.YPOSITION
                   \FASTFX80.BACKUP.PAPER \FASTFX80.ADVANCE.PAPER \FASTFX80.NEWPAGE \FASTFX80.OUTCHAR
                   \FASTFX80.NEWLINE \FASTFX80.LINEFEED \FASTFX80.DRAWLINE)
              (FUNCTIONS \FASTFX80.STARTPAGE \FASTFX80.SMART-XPOSITION \FASTFX80.TOPMARGIN
                   \FASTFX80.BOTTOMMARGIN \FASTFX80.LEFTMARGIN \FASTFX80.RIGHTMARGIN
                   \FASTFX80.CUR-POS-VISIBLE? \FASTFX80.HORIZONTAL))
        (COMS ;; printer code

              (FUNCTIONS \FASTFX80.SEND MAKE-FASTFX80 FASTFX80FILEP \FASTFX80.CANNOT-PRINT-BITMAPS)
              (FNS \FASTFX80.CONVERT-TEDIT))
        (COMS ;; Character transmission method

              (FNS \FASTFX80.BOUT))
        ;; Miscellany

        (FUNCTIONS \FASTFX80.TRANSLATE-CHAR WITH-FASTFX80-DATA)
        (CONSTANTS (\FASTFX80.DOTSPERINCH 72)
                   (\FASTFX80.LINESPERINCH 6)
                   (\FASTFX80.LINEHEIGHT 12)
                                                              ; in dots
                   (\FASTFX80.FILE-SIGNATURE "FastFX-80/Xerox/1.0 "))
```

```
        (INITVARS (FASTFX80-DEFAULT-DESTINATION "{TTY}")
                (\FASTFX80.INCHES-PER-PAGE 11)
                (\FASTFX80.INCHES-PER-LINE 8.5))
        (COMS  ;; need to load these exports

                (DECLARE%: EVAL@LOAD EVAL@COMPILE DONTCOPY (FILES (LOADCOMP)
                                                                  ADISPLAY])
```

;; Fast driver
;;

```
(CL:DEFSTRUCT FASTFX80DATA

    ;; the imagedata vector for a fastfx80 imagestream

    (VIRTUAL-XPOS 0)
    (VIRTUAL-YPOS 0)
    (REAL-XPOS 0)
    (REAL-YPOS 0)
    CLIPPINGREGION BACKINGSTREAM (LEFTMARGIN 72)
    RIGHTMARGIN TOPMARGIN (BOTTOMMARGIN 0)
    FONT PAPER-WIDTH PAPER-HEIGHT (SPACEFACTOR 1.0))

(DEFINEQ

(\FASTFX80.INIT
  [LAMBDA NIL                                                        ; Edited 16-Dec-86 12:03 by hdj
    (DECLARE (GLOBALVARS DEFAULTPRINTINGHOST IMAGESTREAMTYPES PRINTERTYPES PRINTFILETYPES \FASTFX80.IMAGEOPS
                    \FASTFX80.FDEV))
    (SETQ \FASTFX80.FDEV (create FDEV
                                DEVICENAME _ (LIST 'FASTFX80 'PRINTER)
                                CLOSEFILE _ (FUNCTION NILL)
                                BOUT _ (FUNCTION \FASTFX80.BOUT)))
    (SETQ \FASTFX80.IMAGEOPS (create IMAGEOPS
                                IMAGETYPE _ 'FASTFX80
                                IMFONT _ (FUNCTION \FASTFX80.CHANGEFONT)
                                IMLEFTMARGIN _ (FUNCTION \FASTFX80.LEFTMARGIN)
                                IMRIGHTMARGIN _ (FUNCTION \FASTFX80.RIGHTMARGIN)
                                IMTOPMARGIN _ (FUNCTION \FASTFX80.TOPMARGIN)
                                IMBOTTOMMARGIN _ (FUNCTION \FASTFX80.BOTTOMMARGIN)
                                IMLINEFEED _ (FUNCTION NILL)
                                IMTERPRI _ (FUNCTION \FASTFX80.NEWLINE)
                                IMXPOSITION _ (FUNCTION \FASTFX80.XPOSITION)
                                IMYPOSITION _ (FUNCTION \FASTFX80.YPOSITION)
                                IMCLOSEFN _ (FUNCTION \FASTFX80.CLOSE)
                                IMMOVETO _ (FUNCTION \FASTFX80.MOVETO)
                                IMDRAWCURVE _ (FUNCTION NILL)
                                IMFILLCIRCLE _ (FUNCTION NILL)
                                IMDRAWLINE _ (FUNCTION NILL)
                                IMDRAWELLIPSE _ (FUNCTION NILL)
                                IMDRAWCIRCLE _ (FUNCTION NILL)
                                IMBITBLT _ (FUNCTION NILL)
                                IMBLTSHADE _ (FUNCTION NILL)
                                IMNEWPAGE _ (FUNCTION \FASTFX80.NEWPAGE)
                                IMSCALE _ [FUNCTION (LAMBDA NIL 1]
                                IMSPACEFACTOR _ (FUNCTION NILL)
                                IMFONTCREATE _ 'FASTFX80
                                IMCOLOR _ (FUNCTION NILL)
                                IMBACKCOLOR _ (FUNCTION NILL)
                                IMOPERATION _ (FUNCTION NILL)
                                IMSTRINGWIDTH _ (FUNCTION \FASTFX80.STRINGWIDTH)
                                IMCHARWIDTH _ (FUNCTION \FASTFX80.CHARWIDTH)
                                IMCLIPPINGREGION _ (FUNCTION \FASTFX80.CLIPPINGREGION)
                                IMRESET _ (FUNCTION NILL)
                                IMDRAWPOLYGON _ (FUNCTION NILL)
                                IMFILLPOLYGON _ (FUNCTION NILL)
                                IMSCALEDBITBLT _ (FUNCTION NILL)))
    [push IMAGESTREAMTYPES (COPYALL '(FASTFX80 (OPENSTREAM OPENFASTFX80STREAM)
                                            (FONTCREATE \FASTFX80.FONTCREATE)
                                            (FONTSAVAILABLE \SEARCHDISPLAYFONTFILES)
                                            (CREATECHARSET \FASTFX80.CREATECHARSET]
    [push PRINTERTYPES (COPYALL '((FASTFX80)
                                        (CANPRINT (FASTFX80))
                                        (STATUS TRUE)
                                        (SEND \FASTFX80.SEND)
                                        (BITMAPSCALE NIL)
                                        (BITMAPFILE (\FASTFX80.CANNOT-PRINT-BITMAPS FILE BITMAP SCALEFACTOR REGION
                                                        ROTATION TITLE))
                                        (PROPERTIES NIL]
    [push PRINTFILETYPES (COPYALL '(FASTFX80 (TEST FASTFX80FILEP)
                                        (EXTENSION (FASTFX80))
                                        (CONVERSION (TEXT MAKE-FASTFX80 TEDIT \FASTFX80.CONVERT-TEDIT]
    (push DEFAULTPRINTINGHOST (LIST 'FASTFX80 'FASTFX80))
    (PUTPROP 'FASTFX80 'PRINTERTYPE 'FASTFX80)
    (\FASTFX80.INIT-FONT-PROFILE)
    T])
```

)

;; Imagestream methods
;; opening/closing imagestream

```
(DEFINEQ
```

(**OPENFASTFX80STREAM**
  [LAMBDA (FILENAME OPTIONS)   ; Edited 20-Jan-88 11:22 by jds

      ;; open a fastfx80 imagestream

      (LET* [[BACKING (OPENSTREAM FILENAME 'OUTPUT NIL '((SEQUENTIAL T)
                                                         (TYPE FASTFX80]
              (PAPER-WIDTH (FIX (TIMES \FASTFX80.INCHES-PER-LINE \FASTFX80.DOTSPERINCH)))
              (PAPER-HEIGHT (FIX (TIMES \FASTFX80.INCHES-PER-PAGE \FASTFX80.DOTSPERINCH)))
              (FASTFX80STREAM (**create** STREAM
                                      FULLFILENAME _ (FULLNAME BACKING)
                                      DEVICE _ \FASTFX80.FDEV
                                      ACCESS _ 'OUTPUT
                                      OUTCHARFN _ (FUNCTION \FASTFX80.OUTCHAR)
                                      IMAGEOPS _ \FASTFX80.IMAGEOPS
                                      IMAGEDATA _ (MAKE-FASTFX80DATA :BACKINGSTREAM BACKING :CLIPPINGREGION
                                                              (CREATEREGION 0 0 PAPER-WIDTH PAPER-HEIGHT)
                                                              :RIGHTMARGIN
                                                              (- PAPER-WIDTH \FASTFX80.DOTSPERINCH)
                                                              :TOPMARGIN
                                                              (- PAPER-HEIGHT (TIMES 3 \FASTFX80.LINEHEIGHT))
                                                              :BOTTOMMARGIN
                                                              (TIMES 3 \FASTFX80.LINEHEIGHT)
                                                              :PAPER-WIDTH PAPER-WIDTH :PAPER-HEIGHT PAPER-HEIGHT]
              (**replace** (STREAM USERVISIBLE) of BACKING **with** NIL)
              (**FASTFX80.PREAMBLE** FASTFX80STREAM)
              FASTFX80STREAM])
)

(CL:DEFUN **FASTFX80.PREAMBLE** (FASTFX80STREAM)
    ;; start a FASTFX80 master
```

;;; must change FASTFX80FILEP when this changes

```
      (**DECLARE** (GLOBALVARS \FASTFX80.INCHES-PER-PAGE))
      (\**FASTFX80.RESET-PRINTER** FASTFX80STREAM \FASTFX80.INCHES-PER-PAGE)
      (\**FASTFX80.OUTPUT-SIGNATURE** FASTFX80STREAM)
      (\**FASTFX80.CHANGEFONT** FASTFX80STREAM (DEFAULTFONT 'FASTFX80))
      (\**FASTFX80.STARTPAGE** FASTFX80STREAM))


(CL:DEFUN \**FASTFX80.RESET-PRINTER** (FASTFX80STREAM INCHES-PER-PAGE)
    ;; send a reset sequence to the fx-80

    (**IF** (AND (<= 1 INCHES-PER-PAGE)
            (<= INCHES-PER-PAGE 21))
        **THEN** ;; send a reset sequence to the fx-80...

            (\**FASTFX80.BOUT** FASTFX80STREAM (CHARCODE ESC))
            (\**FASTFX80.BOUT** FASTFX80STREAM (CHARCODE @))

            ;; ...and set the form length

            (\**FASTFX80.BOUT** FASTFX80STREAM (CHARCODE ESC))
            (\**FASTFX80.BOUT** FASTFX80STREAM (CHARCODE C))
            (\**FASTFX80.BOUT** FASTFX80STREAM (FIXR (TIMES INCHES-PER-PAGE \FASTFX80.LINESPERINCH)))
        **ELSE** (ERROR "Illegal page length value" INCHES-PER-PAGE)))


(CL:DEFUN \**FASTFX80.OUTPUT-SIGNATURE** (FASTFX80STREAM)
    ;; start the file with an identifying signature.  Ensure it is not printed by following it with an equal number of ASCII 127's.
    ;; This will not work if SIGNATURE contains line-ending characters.

    (LET ((DEL-BYTE 127))
        (**FOR** BYTE INSTRING \FASTFX80.FILE-SIGNATURE **DO** (\**FASTFX80.BOUT** FASTFX80STREAM BYTE))
        (**FOR** BYTE INSTRING \FASTFX80.FILE-SIGNATURE **DO** (\**FASTFX80.BOUT** FASTFX80STREAM DEL-BYTE))))

(DEFINEQ
```

(**FASTFX80.CLOSE**
  [LAMBDA (FASTFX80STREAM)   ; Edited 2-Jun-87 19:11 by Snow

      ;; close a fast fx80 stream

      (**WITH-FASTFX80-DATA** (DATA FASTFX80STREAM)
          (**FASTFX80.OUTCHAR** FASTFX80STREAM (CHARCODE CR))

          ;; do a bout here because an outchar will cause a new-page which then adds 4 lines to the output.

```
              (\FASTFX80.BOUT FASTFX80STREAM (CHARCODE FF))
              (\CLOSEFILE (FASTFX80DATA-BACKINGSTREAM DATA))
              (fetch (STREAM FULLFILENAME) of FASTFX80STREAM])
)
```

;; methods that hack fonts

```
(DEFINEQ
```

(**FASTFX80.CHANGEFONT**
```
  [LAMBDA (STREAM FONT)                                               ; Edited 14-Aug-87 14:40 by Snow

    ;; font-change method for the fast fx-80 device

    (WITH-FASTFX80-DATA (DATA STREAM)
        (PROG1 (FASTFX80DATA-FONT DATA)
              [COND
                  (FONT (SETQ FONT (FONTCREATE FONT NIL NIL NIL 'FASTFX80))
                        (COND
                           ((NEQ FONT (FASTFX80DATA-FONT DATA))
                            [LET [[ITALICP (FMEMB 'ITALIC (FONTPROP FONT 'FACE]
                                  [BOLDP (FMEMB 'BOLD (FONTPROP FONT 'FACE]
                                  (UNDERLINE-NESS 128)
                                  (ITALIC-NESS 64)
                                  (EXPANDED-NESS 32)
                                  (DOUBLE-STRIKE-NESS 16)
                                  (EMPHASIZED-NESS 8)
                                  (COMPRESSED-NESS 4)
                                  (ELITE-NESS 1)
                                  (PICA-NESS 0)
                                  (SIZE (FONTPROP FONT 'SIZE]

                             ;; Send master select code and inform printer of boldness, italicism, and new font size: pica for regular sized
                             ;; fonts, emphasized pica for large fonts.

                                (\FASTFX80.BOUT STREAM (CHARCODE ESC))
                                (\FASTFX80.BOUT STREAM (CHARCODE !))
                                [\FASTFX80.BOUT STREAM (LOGOR (COND
                                                                    (ITALICP ITALIC-NESS)
                                                                    (T PICA-NESS))
                                                             (COND
                                                                 (BOLDP EMPHASIZED-NESS)
                                                                 (T PICA-NESS))
                                                             (COND
                                                                 ((> SIZE 12)
                                                                  EXPANDED-NESS)
                                                                 ((<= SIZE 8)
                                                                  COMPRESSED-NESS)
                                                                 (T PICA-NESS]

                             ;; Set italicness, since FX-80 doesn't support the ITALIC bit in master reset.

                                (\FASTFX80.BOUT STREAM (CHARCODE ESC))
                                (COND
                                   (ITALICP                          ; turn it on
                                         (\FASTFX80.BOUT STREAM (CHARCODE 4)))
                                   (T                                ; turn it off
                                      (\FASTFX80.BOUT STREAM (CHARCODE 5]
                           (CL:SETF (FASTFX80DATA-FONT DATA)
                                  FONT])])
```

(**FASTFX80.FONTCREATE**
```
  [LAMBDA (FAMILY SIZE FACE ROTATION DEVICE CHARSET)            (* hdj "27-Oct-86 14:59")

    ;; create and return a fontdescriptor for a fastfx80 font

    (LET ((FONTDESC (create FONTDESCRIPTOR
                            FONTDEVICE _ 'FASTFX80
                            FONTFAMILY _ FAMILY
                            FONTSIZE _ SIZE
                            FONTFACE _ FACE
                            ROTATION _ ROTATION
                            FONTSCALE _ 1
                            \SFHeight _ 9
                            \SFAscent _ 7
                            \SFDescent _ 2)))
         (if (\GETCHARSETINFO CHARSET FONTDESC T)
             then FONTDESC
           else NIL])
```

(**FASTFX80.CREATECHARSET**
```
  [LAMBDA (FAMILY SIZE FACE ROTATION DEVICE CHARSET FONTDESC)     (* hdj "27-Oct-86 14:57")

    ;; Create a character set for the fast fx-80.  Really only works for char set 0; returns the same info for all sets.

         (* * (if (NEQ 0 CHARSET) then (ERROR "FX-80 does not support NS characters.")))

    (LET ((WIDTHS (\CREATECSINFOELEMENT)))
         (for C from 32 to 254 do (\FSETWIDTH WIDTHS C (\FASTFX80.SUBCHARWIDTH C SIZE)))
```

```
        (create CHARSETINFO
                WIDTHS _ WIDTHS
                IMAGEWIDTHS _ WIDTHS
                YWIDTHS _ (\CREATECSINFOELEMENT)
                CHARSETASCENT _ (ffetch \SFAscent of FONTDESC)
                CHARSETDESCENT _ (ffetch \SFDescent of FONTDESC])

)


(CL:DEFUN \FASTFX80.INIT-FONT-PROFILE ()
    ;; set up the fonts for the FASTFX80, based on the DISPLAY font profile entries

    [FOR FONT-CLASS IN '(DEFAULTFONT ITALICFONT BOLDFONT LITTLEFONT TINYFONT BIGFONT COMMENTFONT TEXTFONT)
       DO (\ADD-TO-FONTPROFILE FONTPROFILE FONT-CLASS 'FASTFX80 (\GET-FROM-FONTPROFILE FONTPROFILE FONT-CLASS
                                                                              'DISPLAY]
    (FONTPROFILE FONTPROFILE)
    T)
```

;; methods for measuring

```
(DEFINEQ
```

(**FASTFX80.STRINGWIDTH**
```
  [LAMBDA (FASTFX80STREAM STRING RDTBL)                              (* hdj " 6-Nov-86 15:15")

    ;;  returns STRING's width, relative to STREAM's current font and the readtable RDTBL

    (if RDTBL
        then (bind (FIRSTFLG _ T)
                   (SA _ (fetch READSA of RDTBL))
                   (ESCAPE-CHAR-WIDTH _ (\FASTFX80.CHARWIDTH FASTFX80STREAM (fetch (READTABLEP ESCAPECHAR)
                                                                                    of RDTBL)))
                   (SYN _ NIL) for CHARCODE instring STRING
              sum (PROG1 (+ (\FASTFX80.CHARWIDTH FASTFX80STREAM CHARCODE)
                            (IF (AND (fetch (READCODE ESCQUOTE) of (SETQ SYN (\SYNCODE SA CHARCODE)))
                                     (OR FIRSTFLG (fetch (READCODE INNERESCQUOTE) of SYN)))
                                THEN ESCAPE-CHAR-WIDTH
                              ELSE 0))
                         (SETQ FIRSTFLG NIL)))
        else (for CHAR instring STRING sum (\FASTFX80.CHARWIDTH FASTFX80STREAM CHAR])
```

(**FASTFX80.CHARWIDTH**
```
  [LAMBDA (STREAM CHARCODE)                                          ; Edited  4-Feb-87 15:52 by hdj

    ;;  returns the width of CHARCODE, relative to STREAM's current font

    (WITH-FASTFX80-DATA (DATA STREAM)
        (LET [(WIDTH (\FASTFX80.SUBCHARWIDTH (\FASTFX80.TRANSLATE-CHAR CHARCODE)
                                (FONTPROP (FASTFX80DATA-FONT DATA)
                                          'SIZE]
             (IF (EQ CHARCODE (CHARCODE SPACE))
                 THEN (FIXR (TIMES WIDTH (FASTFX80DATA-SPACEFACTOR DATA)))
               ELSE WIDTH])
```

(**FASTFX80.SUBCHARWIDTH**
```
  [LAMBDA (CHARCODE SIZE)                                            ; Edited 21-Jan-88 12:10 by jds

    ;; Computes the size for a single character in Fast-FX80 mode.

    (COND
       ((IGEQ CHARCODE 31)                                          ; Only non-control characters have real widths
        (COND
           ((GREATERP SIZE 12)                                      ; Fonts bigger than 12 are printed EXPANDED.
            14)
           ((<= SIZE 8)                                             ; Sizes 8 & under are printed compressed, 17.16 pitch, or 4.19
                                                                    ; dots per
            4)
           (T                                                       ; Should really be 7.2 dots, but this is close.
             7)))
       (T 0])

)


(CL:DEFUN \FASTFX80.SPACEFACTOR (FASTFX80STREAM FACTOR)
    ;; returns/sets the width of the space character (32 ASCII) for FASTFX80STREAM

    [WITH-FASTFX80-DATA (DATA FASTFX80STREAM)
        (PROG1 (FASTFX80DATA-SPACEFACTOR DATA)
               (AND FACTOR (IF (NUMBERP FACTOR)
                               THEN (CL:SETF (FASTFX80DATA-SPACEFACTOR DATA)
                                             FACTOR)
                             ELSE (\ILLEGAL.ARG FACTOR)))))])
```

;; methods that affect the current position/size of drawing surface

```
(DEFINEQ
```

(\FASTFX80.CLIPPINGREGION
```
  [LAMBDA (STREAM REGION)                                              ; Edited  8-Dec-86 15:16 by hdj
```
   ;; Returns old clipping region and sets new one. will never set onelarger than the paper size.

```
      (DECLARE (GLOBALVARS \FASTFX80.PAGESIZE))
      (WITH-FASTFX80-DATA (DATA STREAM)
            (PROG1 (COPY (FASTFX80DATA-CLIPPINGREGION DATA))
                [AND REGION (CL:SETF (FASTFX80DATA-CLIPPINGREGION DATA)
                                     (INTERSECTREGIONS REGION (CREATEREGION 0 0 (FASTFX80DATA-PAPER-WIDTH DATA)
                                                                 (FASTFX80DATA-PAPER-HEIGHT DATA]))])
```

(\FASTFX80.MOVETO
```
  [LAMBDA (STREAM X Y)                                                 (* hdj "27-Oct-86 11:40")
```
   ;; move to (X,Y) on STREAM's drawing surface

```
      (\FASTFX80.XPOSITION STREAM X)
      (\FASTFX80.YPOSITION STREAM Y])
```

(\FASTFX80.XPOSITION
```
  [LAMBDA (FASTFX80STREAM XPOS)                                        (* hdj "20-Nov-86 17:50")
```
   ;; Return old x-position, optionally move to new one. If new position would lie outside the clipping region, set the virtual x position, but don't change
   ;; the real x position or move the printer's print head.

```
      (WITH-FASTFX80-DATA (DATA FASTFX80STREAM)
            (LET ((OLD-REAL-XPOS (FASTFX80DATA-REAL-XPOS DATA))
                  (OLD-VIRTUAL-XPOS (FASTFX80DATA-VIRTUAL-XPOS DATA))
                  (CLIPPINGREGION (FASTFX80DATA-CLIPPINGREGION DATA)))
                 (PROG1 OLD-VIRTUAL-XPOS
                     (if XPOS
                         then  ;; Space or backspace till new x-pos approximates desired position

                             (LET ((LEFT-BORDER (fetch (REGION LEFT) of CLIPPINGREGION))
                                   (RIGHT-BORDER (fetch (REGION RIGHT) of CLIPPINGREGION)))
                                 (if (AND (LEQ LEFT-BORDER XPOS)
                                          (LEQ XPOS RIGHT-BORDER))
                                     then (if (AND (EQP (FASTFX80DATA-LEFTMARGIN DATA)
                                                        0)
                                                   (EQP XPOS 0))
                                              then                  ; if we can, just send a CR
                                                  (\FASTFX80.BOUT FASTFX80STREAM (CHARCODE CR))
                                              else                  ; otherwise do the full schmeer
                                                  (\FASTFX80.SMART-XPOSITION OLD-REAL-XPOS XPOS FASTFX80STREAM))
                                          (CL:SETF (FASTFX80DATA-REAL-XPOS DATA)
                                                   XPOS)))
                             (CL:SETF (FASTFX80DATA-VIRTUAL-XPOS DATA)
                                      XPOS)))])
```

(\FASTFX80.YPOSITION
```
  [LAMBDA (STREAM YPOS)                                                ; Edited  9-Dec-86 22:43 by hdj
```
   ;; Return old y position, optionally move to new one. If new position would lie outside the clipping region, set the virtual Y position, but don't change
   ;; the real Y position or move the printer's print head.

```
      (WITH-FASTFX80-DATA
       (DATA STREAM)
       (LET ((OLD-REAL-YPOS (FASTFX80DATA-REAL-YPOS DATA))
             (OLD-VIRTUAL-YPOS (FASTFX80DATA-VIRTUAL-YPOS DATA))
             (CLIPPINGREGION (FASTFX80DATA-CLIPPINGREGION DATA)))
            (PROG1 OLD-VIRTUAL-YPOS
                (if YPOS
                    then [if (NOT (EQP YPOS OLD-REAL-YPOS))
                             then (LET ((TOP-BORDER (fetch (REGION TOP) of CLIPPINGREGION))
                                        (BOTTOM-BORDER (fetch (REGION BOTTOM) of CLIPPINGREGION)))
                                       (if (NOT (EQP YPOS OLD-REAL-YPOS))
                                           then (LET [(DOTS-TO-MOVE (FIX (- YPOS OLD-REAL-YPOS]
                                                     (if (MINUSP DOTS-TO-MOVE)
                                                         then (\FASTFX80.ADVANCE.PAPER STREAM DOTS-TO-MOVE)
                                                         else (\FASTFX80.BACKUP.PAPER STREAM DOTS-TO-MOVE)))
                                                (CL:SETF (FASTFX80DATA-REAL-YPOS DATA)
                                                         YPOS]
                             (CL:SETF (FASTFX80DATA-VIRTUAL-YPOS DATA)
                                      YPOS)))])
```

(\FASTFX80.BACKUP.PAPER
```
  [LAMBDA (STREAM DOTS)                                                (* hdj "28-Oct-86 12:59")
```
   ;; backup the page DOTS raster lines

```
      (SETQ DOTS (TIMES 3 (ABS DOTS)))
      (while (GREATERP DOTS 0) do (\FASTFX80.BOUT STREAM (CHARCODE ESC))
                                  (\FASTFX80.BOUT STREAM (CHARCODE j))
                                  (\FASTFX80.BOUT STREAM (LET ((MAXBACKUP (MIN DOTS 255)))
                                                              (add DOTS (MINUS MAXBACKUP))
```

                                                                            MAXBACKUP])

FASTFX80.ADVANCE.PAPER
```
  [LAMBDA (STREAM DOTS)                                          (* hdj "28-Oct-86 12:58")
```
  ;; advance the page DOTS raster lines
```
    (SETQ DOTS (TIMES 3 (ABS DOTS)))
    (while (GREATERP DOTS 0) do (\FASTFX80.BOUT STREAM (CHARCODE ESC))
                                (\FASTFX80.BOUT STREAM (CHARCODE J))
                                (\FASTFX80.BOUT STREAM (LET ((MAXADVANCE (MIN DOTS 255)))
                                                         (add DOTS (MINUS MAXADVANCE))
                                                         MAXADVANCE])
```

FASTFX80.NEWPAGE
```
  [LAMBDA (FASTFX80STREAM)                                       ; Edited 17-Dec-86 10:32 by hdj
```
  ;; End the old page, start a new one
  ;; Just send a form-feed
```
    (\FASTFX80.BOUT FASTFX80STREAM (CHARCODE FF))
    (\FASTFX80.STARTPAGE FASTFX80STREAM])
```

FASTFX80.OUTCHAR
```
  [LAMBDA (FASTFX80STREAM CHARCODE)                              ; Edited 12-Feb-87 09:08 by jds
```
  ;; outcharfn for fastfx80 imagestreams
```
    (LET ((TRANSLATED-CHAR (\FASTFX80.TRANSLATE-CHAR CHARCODE)))
         (WITH-FASTFX80-DATA (DATA FASTFX80STREAM)
              (SELCHARQ CHARCODE
                  (^L (\FASTFX80.NEWPAGE FASTFX80STREAM))
                  ((CR EOL)
                       (\FASTFX80.NEWLINE FASTFX80STREAM))
                  (LF (\FASTFX80.LINEFEED FASTFX80STREAM))
                  (SPACE [\FASTFX80.XPOSITION FASTFX80STREAM (+ (\FASTFX80.XPOSITION FASTFX80STREAM)
                                                               (\FASTFX80.CHARWIDTH FASTFX80STREAM
                                                                      (CHARCODE SPACE]
                        (COND
                           ((> (FASTFX80DATA-VIRTUAL-XPOS DATA)
                               (FASTFX80DATA-RIGHTMARGIN DATA))
                            (\FASTFX80.NEWLINE FASTFX80STREAM))))
                  (COND
                     ((GREATERP CHARCODE 32)
```
                       ;; only print graphic characters
```
                       (LET ((CHARWIDTH (\FASTFX80.CHARWIDTH FASTFX80STREAM TRANSLATED-CHAR)))
```
                            ;; if character will be visible, output it
```
                            (COND
                               ( (\FASTFX80.CUR-POS-VISIBLE? DATA)
                                 (\FASTFX80.BOUT FASTFX80STREAM TRANSLATED-CHAR)))
                            (CL:INCF (FASTFX80DATA-REAL-XPOS DATA)
                                   CHARWIDTH)
                            (CL:INCF (FASTFX80DATA-VIRTUAL-XPOS DATA)
                                   CHARWIDTH)
```
                            ;; if we've passed the margin, DING!, do a newline
```
                            (COND
                               ((> (FASTFX80DATA-VIRTUAL-XPOS DATA)
                                   (FASTFX80DATA-RIGHTMARGIN DATA))
                                (\FASTFX80.NEWLINE FASTFX80STREAM])
```

FASTFX80.NEWLINE
```
  [LAMBDA (FASTFX80STREAM)                                       (* hdj "11-Nov-86 14:02")
```
  ;; perform a newline on a fastfx80 imagestream. if we go below the bottom margin, start a new page.
```
    (DECLARE (GLOBALVARS \FASTFX80.LINEHEIGHT))
    (WITH-FASTFX80-DATA (DATA FASTFX80STREAM)
         (LET ((NEW-Y (- (FASTFX80DATA-VIRTUAL-YPOS DATA)
                         \FASTFX80.LINEHEIGHT)))
              (if (< NEW-Y (FASTFX80DATA-BOTTOMMARGIN DATA))
                  then (\FASTFX80.NEWPAGE FASTFX80STREAM)
                  else                                          ; move to the left margin
                       (\FASTFX80.XPOSITION FASTFX80STREAM (FASTFX80DATA-LEFTMARGIN DATA))
                       (FREPLACE (STREAM CHARPOSITION) OF FASTFX80STREAM WITH 0)
                                                                ; then move down or newpage
                       (\FASTFX80.YPOSITION FASTFX80STREAM NEW-Y])
```

FASTFX80.LINEFEED
```
  [LAMBDA (FASTFX80STREAM)                                       (* hdj " 6-Nov-86 15:38")
```
  ;; move down 1 line, leaving the x-position alone
```
    (DECLARE (GLOBALVARS \FASTFX80.LINEHEIGHT))
    (WITH-FASTFX80-DATA (DATA FASTFX80STREAM)
```

```
              (LET ((NEW-YPOS (- (FASTFX80DATA-VIRTUAL-YPOS DATA)
                                 \FASTFX80.LINEHEIGHT))
                    (OLD-XPOS (FASTFX80DATA-VIRTUAL-XPOS DATA)))
                   (if (< NEW-YPOS (FASTFX80DATA-BOTTOMMARGIN DATA))
                       then                                             ; move to a new page
                            (\FASTFX80.NEWPAGE FASTFX80STREAM)          ; restore the old x position
                            (\FASTFX80.XPOSITION FASTFX80STREAM OLD-XPOS)
                       else                                            ; move down
                            (\FASTFX80.YPOSITION FASTFX80STREAM NEW-YPOS])
```

(**FASTFX80.DRAWLINE**

```
   [LAMBDA (STREAM X1 Y1 X2 Y2 WIDTH OPERATION COLOR DASHING)        (* hdj "31-Oct-86 14:09")

   ;; dummy drawline for the fast fx80 device

     (MOVETO X2 Y2 STREAM])

)
```

```
(CL:DEFUN \FASTFX80.STARTPAGE (FASTFX80STREAM)
    (WITH-FASTFX80-DATA (DATA FASTFX80STREAM)
        (LET [(ASCENT (FONTPROP (DSPFONT NIL FASTFX80STREAM)
                               'ASCENT]

            ;; set the "actual" position of printhead on paper after a newpage, then let the driver figure out how to get to (leftmargin, topmargin).

            (CL:SETF (FASTFX80DATA-VIRTUAL-XPOS DATA)
                    0)
            (CL:SETF (FASTFX80DATA-REAL-XPOS DATA)
                    0)
            (CL:SETF (FASTFX80DATA-VIRTUAL-YPOS DATA)
                    (FASTFX80DATA-PAPER-HEIGHT DATA))
            (CL:SETF (FASTFX80DATA-REAL-YPOS DATA)
                    (FASTFX80DATA-PAPER-HEIGHT DATA))

            ;; move the paper

            (MOVETO (FASTFX80DATA-LEFTMARGIN DATA)
                    (- (FASTFX80DATA-TOPMARGIN DATA)
                       ASCENT)
                    FASTFX80STREAM)
            FASTFX80STREAM)))
```

```
(CL:DEFUN \FASTFX80.SMART-XPOSITION (CURRENT-XPOS DESIRED-XPOS FASTFX80STREAM)

   ;; if it would create less output to space from the left margin, rather than to backspace from the current position, do so

    (LET* ((SPACEWIDTH (\FASTFX80.CHARWIDTH FASTFX80STREAM (CHARCODE SP)))
           (CURRENT-XPOS-IN-SPACES (IQUOTIENT CURRENT-XPOS SPACEWIDTH))
           (DESIRED-XPOS-IN-SPACES (IQUOTIENT DESIRED-XPOS SPACEWIDTH))
           (NUM-BACKSPACES-NEEDED (- CURRENT-XPOS-IN-SPACES DESIRED-XPOS-IN-SPACES)))
         (IF (< NUM-BACKSPACES-NEEDED DESIRED-XPOS-IN-SPACES)
             THEN ;; if backspacing's cheaper, backspace away

                  (\FASTFX80.HORIZONTAL (- NUM-BACKSPACES-NEEDED)
                         FASTFX80STREAM)
             ELSE ;; otherwise, go to the left margin...

                  (\FASTFX80.BOUT FASTFX80STREAM (CHARCODE CR))

                  ;; ... and then space to the right spot

                  (\FASTFX80.HORIZONTAL DESIRED-XPOS-IN-SPACES FASTFX80STREAM))))
```

```
(CL:DEFUN \FASTFX80.TOPMARGIN (STREAM &OPTIONAL YPOSITION)
    [WITH-FASTFX80-DATA (DATA STREAM)
        (PROG1 (FASTFX80DATA-TOPMARGIN DATA)
            (AND YPOSITION (IF (SMALLP YPOSITION)
                               THEN (CL:SETF (FASTFX80DATA-TOPMARGIN DATA)
                                            YPOSITION)
                               ELSE (\ILLEGAL.ARG YPOSITION))))])
```

```
(CL:DEFUN \FASTFX80.BOTTOMMARGIN (STREAM &OPTIONAL YPOSITION)
    [WITH-FASTFX80-DATA (DATA STREAM)
        (PROG1 (FASTFX80DATA-BOTTOMMARGIN DATA)
            (AND YPOSITION (IF (SMALLP YPOSITION)
                               THEN (CL:SETF (FASTFX80DATA-BOTTOMMARGIN DATA)
                                            YPOSITION)
                               ELSE (\ILLEGAL.ARG YPOSITION))))])
```

```
(CL:DEFUN \FASTFX80.LEFTMARGIN (STREAM &OPTIONAL XPOSITION)
    [WITH-FASTFX80-DATA (DATA STREAM)
        (PROG1 (FASTFX80DATA-LEFTMARGIN DATA)
            (AND XPOSITION (IF (SMALLP XPOSITION)
                               THEN (CL:SETF (FASTFX80DATA-LEFTMARGIN DATA)
                                            XPOSITION)
```

```
                                     ELSE (\ILLEGAL.ARG XPOSITION)))))])


(CL:DEFUN \FASTFX80.RIGHTMARGIN (STREAM &OPTIONAL XPOSITION)
   [WITH-FASTFX80-DATA (DATA STREAM)
          (PROG1 (FASTFX80DATA-RIGHTMARGIN DATA)
             (AND XPOSITION (IF (SMALLP XPOSITION)
                                THEN (CL:SETF (FASTFX80DATA-RIGHTMARGIN DATA)
                                              XPOSITION)
                                ELSE (\ILLEGAL.ARG XPOSITION)))))])


(DEFMACRO \FASTFX80.CUR-POS-VISIBLE? (FASTFX80DATA)
   `(INSIDEP (FASTFX80DATA-CLIPPINGREGION ,FASTFX80DATA)
            (FASTFX80DATA-REAL-XPOS ,FASTFX80DATA)
            (FASTFX80DATA-REAL-YPOS ,FASTFX80DATA)))


(CL:DEFUN \FASTFX80.HORIZONTAL (SPACES FASTFX80STREAM)

   ;; print SPACES space characters if SPACES > 0, print SPACES backspaces if < 0, and do nothing if SPACES=0.

   [if (MINUSP SPACES)
       then (for SPACE from 1 to (ABS SPACES) by 1 do (\FASTFX80.BOUT FASTFX80STREAM (CHARCODE BS)))
       else (for SPACE from 1 to SPACES by 1 do (\FASTFX80.BOUT FASTFX80STREAM (CHARCODE SP])

;; printer code


(CL:DEFUN \FASTFX80.SEND (PRINTER FILENAME &OPTIONAL OPTIONS)

   ;; send the file designated by FILENAME to PRINTER, obeying OPTIONS.  Since we only have one fx-80 per machine, ignore PRINTER and send
   ;; to FASTFX80-DEFAULT-DESTINATION

   (DECLARE (GLOBALVARS FASTFX80-DEFAULT-DESTINATION))
   [LET [(COPIES (LISTGET OPTIONS '%#COPIES]
        (FOR COPY FROM 1 TO COPIES DO  ;; allow the user to abort it while running

                                   (WITH-ABORT-WINDOW ((THIS.PROCESS)
                                                        FILENAME PRINTER COPY)
                                      (COPYFILE FILENAME FASTFX80-DEFAULT-DESTINATION
                                                '((TYPE FASTFX80])


(CL:DEFUN MAKE-FASTFX80 (FILE FASTFX80FILE &OPTIONAL FONTS HEADING TABS OPTIONS)

   ;; turn FILE into a FASTFX80 master

   (TEXTTOIMAGEFILE FILE FASTFX80FILE 'FASTFX80 FONTS HEADING TABS OPTIONS))


(CL:DEFUN FASTFX80FILEP (FASTFX80FILE?)

   ;; is FILE (a filename or stream) a fastfx80 file?

   [LET [(FILE-TYPE (GETFILEINFO FASTFX80FILE? 'TYPE]
        (IF (EQ FILE-TYPE 'FASTFX80)
            THEN                                                ; if file has a type, and type=FASTFX80, we win
              T
            ELSE                                                ; no filetype or filetype not FASTFX80, so read the file
              (LET [(STREAM (OPENSTREAM (INTERLISP-NAMESTRING FASTFX80FILE?)
                                        'INPUT
                                        'OLD
                                        '(SEQUENTIAL]
                   ;; file looks like ESC@ESCCn<signature>...

                   (PROG1 [AND (> (GETFILEINFO STREAM 'LENGTH)
                                  (+ 5 (NCHARS \FASTFX80.FILE-SIGNATURE)))

                               ;; yuck...

                               (EQ (CHARCODE ESC)
                                   (BIN STREAM))
                               (EQ (CHARCODE @)
                                   (BIN STREAM))
                               (EQ (CHARCODE ESC)
                                   (BIN STREAM))
                               (EQ (CHARCODE C)
                                   (BIN STREAM))
                               (BIN STREAM)
                               (FOR CH INSTRING \FASTFX80.FILE-SIGNATURE ALWAYS (EQ CH (BIN STREAM]
                          (CLOSEF STREAM])


(CL:DEFUN \FASTFX80.CANNOT-PRINT-BITMAPS (&OPTIONAL FILE BITMAP SCALEFACTOR REGION ROTATION TITLE)
   (PRINTOUT PROMPTWINDOW "Sorry, FASTFX80 cannot render graphics." T "Use HQFX80 instead."))

(DEFINEQ

(\FASTFX80.CONVERT-TEDIT
  [LAMBDA (TEDIT-FILE IMAGESTREAM)                                      ; Edited 11-Dec-86 17:29 by hdj
```

```
    ;; Send the text to the printer.

    (SETQ TEDIT-FILE (OPENTEXTSTREAM TEDIT-FILE))
    (TEDIT.FORMAT.HARDCOPY TEDIT-FILE IMAGESTREAM T NIL NIL NIL 'FASTFX80)
    (CLOSEF? IMAGESTREAM)
    IMAGESTREAM])
)
```

;; Character transmission method

```
(DEFINEQ
```

(\FASTFX80.BOUT
```
  [LAMBDA (FASTFX80STREAM BYTE)                                      (* hdj "27-Oct-86 11:51")

    ;; send a byte to the fx80

    (WITH-FASTFX80-DATA (DATA FASTFX80STREAM)
          (BOUT (FASTFX80DATA-BACKINGSTREAM DATA)
                BYTE])
)
```

;; Miscellany

```
(DEFMACRO \FASTFX80.TRANSLATE-CHAR (CHARCODE)
    '(SELCHARQ ,CHARCODE
          (357,146                                                  ; bullet
                  (CHARCODE *))
          (357,45                                                   ; em-dash
                  95)
          (357,44                                                   ; en-dash
                  45)
          (\CHAR8CODE ,CHARCODE)))


(DEFMACRO WITH-FASTFX80-DATA ((VAR-NAME STREAM)
                                      &BODY
                                      (BODY DECLS ENV))
    '(LET [(,VAR-NAME (FETCH (STREAM IMAGEDATA) OF ,STREAM]
        ,@DECLS
        ,@BODY))

(DECLARE%: EVAL@COMPILE

(RPAQQ \FASTFX80.DOTSPERINCH 72)

(RPAQQ \FASTFX80.LINESPERINCH 6)

(RPAQQ \FASTFX80.LINEHEIGHT 12)

(RPAQ \FASTFX80.FILE-SIGNATURE "FastFX-80/Xerox/1.0 ")

(CONSTANTS (\FASTFX80.DOTSPERINCH 72)
        (\FASTFX80.LINESPERINCH 6)
        (\FASTFX80.LINEHEIGHT 12)
        (\FASTFX80.FILE-SIGNATURE "FastFX-80/Xerox/1.0 "))
)

(RPAQ? FASTFX80-DEFAULT-DESTINATION "{TTY}")

(RPAQ? \FASTFX80.INCHES-PER-PAGE 11)

(RPAQ? \FASTFX80.INCHES-PER-LINE 8.5)
```

;; need to load these exports

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

(FILESLOAD (LOADCOMP)
        ADISPLAY)
)

(RPAQQ FX-80.HIGH-QUALITY-DRIVERCOMS
            [;; High-quality driver
            ;;
            (STRUCTURES HQFX80DATA)
            (FNS \HQFX80.INIT)
            (COMS ;; imagestream methods

                  (COMS ;; opening/closing imagestream

                        (COMS (FNS OPENHQFX80STREAM)
                              (FUNCTIONS \HQFX80.PREAMBLE \HQFX80.RESET-PRINTER \HQFX80.OUTPUT-SIGNATURE))
```

```
                              (FNS \HQFX80.CLOSE))
              (COMS ;; methods that hack fonts

                              (FNS \HQFX80.FONTCREATE \HQFX80.CHANGEFONT \HQFX80.CREATECHARSET \HQFX80.CHANGE-CHARSET
                                   \HQFX80.READ-FONT-FILE \HQFX80.SEARCH-FONTS)
                              (FUNCTIONS \HQFX80.INIT-FONT-PROFILE))
              (COMS ;; methods for measuring

                              (FNS \HQFX80.CHARWIDTH \HQFX80.STRINGWIDTH)
                              (FUNCTIONS \HQFX80.SPACEFACTOR))
              (COMS ;; methods that affect the current position/size of drawing surface

                              (FNS \HQFX80.CLIPPINGREGION \HQFX80.LEFTMARGIN \HQFX80.RIGHTMARGIN \HQFX80.TOPMARGIN
                                   \HQFX80.BOTTOMMARGIN \HQFX80.XPOSITION \HQFX80.YPOSITION \HQFX80.NEWLINE
                                   \HQFX80.NEWPAGE \HQFX80.LINEFEED \HQFX80.RESET \HQFX80.STARTPAGE)
                              (FUNCTIONS \HQFX80.CUR-POS-VISIBLE?))
              (COMS ;; graphical operations

                              (RESOURCES \HQFX80.BRUSHBBT)
                              (FNS \HQFX80.BITBLT \HQFX80.BLTSHADE \HQFX80.DRAWELLIPSE \HQFX80.OPERATION \HQFX80.DRAWPOINT
                                   )
                              (FNS \HQFX80.DRAWLINE \HQFX80.CLIP-AND-DRAW-LINE \HQFX80.CLIP-AND-DRAW-LINE1)
                              (COMS (FNS \HQFX80.DRAWCIRCLE \HQFX80.CREATE-BRUSH-BBT)
                                    (FUNCTIONS \HQFX80.DRAW-4-CIRCLE-POINTS))
                              (COMS (FNS \HQFX80.FILLCIRCLE \HQFX80.DRAWARC)
                                    (FUNCTIONS \HQFX80.FILL-CIRCLE-BLT))
                              (COMS ;; curve-drawing

                                    (FNS \HQFX80.DRAWCURVE \HQFX80.DRAWCURVE2 \HQFX80.DRAWCURVE3 \HQFX80.LINEWITHBRUSH)
                                    (FNS \HQFX80.BBTCURVEPT)
                                    (MACROS \HQFX80.CURVEPT)
                                    (FUNCTIONS \HQFX80.SMOOTH-CURVE .SETUP.FOR.\HQFX80.BBTCURVEPT.)))
              (COMS ;; character printing methods

                              (FNS \HQFX80.OUTCHAR \HQFX80.BLT-CHAR))
              (COMS ;; printer code

                              (FNS \HQFX80.DUMP-PAGE-BUFFER \HQFX80.ADVANCE-8-LINES)
                              (FUNCTIONS \HQFX80.EIGHT-LINES-BLANK? \HQFX80.BITMAP-LDB \HQFX80.CLEAR-SCANLINE
                                   \HQFX80.CLEAR-WORD-BOX)
                              (FUNCTIONS \HQFX80.SEND MAKE-HQFX80 HQFX80FILEP))
              (COMS ;; window hardcopy

                              (FNS \HQFX80.BITMAP-FILE \HQFX80.CONVERT-TEDIT))
              (COMS ;; character transmission method

                              (FNS \HQFX80.BOUT))
              (COMS ;; handling font-information caching

                              (FNS \HQFX80.FIX-LINE-LENGTH \HQFX80.FIX-FONT \HQFX80.FIX-Y)
                              (FUNCTIONS \HQFX80.INVALIDATE-CACHE \HQFX80.INVALIDATE-FONT-CACHE
                                   \HQFX80.GET-CACHED-CHAR-WIDTH \HQFX80.GET-CHARACTER-OFFSET))
              (COMS ;; auxiliary functions

                              (FUNCTIONS \HQFX80.GRAPHICS-MODE)
                              (FNS \HQFX80.PRINTER-MODE)
                              (FUNCTIONS WITH-HQFX80-DATA))
              ;; and miscellany

              (CONSTANTS (\HQFX80.FILE-SIGNATURE "HQFX-80/Xerox/1.0 ")
                         (\HQFX80.1-TO-1-MODE-DPI 72)
                         (\HQFX80.LOW-SPEED-DOUBLE-MODE-DPI 120))
              (INITVARS (\HQFX80.INCHES-PER-PAGE 11)
                        (\HQFX80.INCHES-PER-LINE 8.5)
                        (HQFX80-DEFAULT-DESTINATION "{TTY}")
                        (HQFX80-FONT-EXTENSIONS DISPLAYFONTEXTENSIONS)
                        (HQFX80-FONT-DIRECTORIES DISPLAYFONTDIRECTORIES)
                        (HQFX80-FONT-COERCIONS DISPLAYFONTCOERCIONS)
                        (HQFX80-MISSING-FONT-COERCIONS MISSINGDISPLAYFONTCOERCIONS])
```

;; High-quality driver
;;

```
(CL:DEFSTRUCT HQFX80DATA

    ;; the imagedata vector for an HQFX80 imagestream

    BACKINGBITMAP
    BACKINGSTREAM
    (XPOS 0)
    (YPOS 0)
    (CLIPPINGREGION (create REGION))
    LINEFEED RIGHTMARGIN (LEFTMARGIN 0)
    TOPMARGIN
    (BOTTOMMARGIN 0)
    OPERATION SOURCETYPE (PILOTBBT (create PILOTBBT
```

```
                                                  PBTDISJOINT _ T))
    (TEXTURE WHITESHADE)
    FONT
    (CHARSET-ASCENT-CACHE MAX.SMALLP)
    WIDTHS-CACHE OFFSETS-CACHE IMAGE-WIDTHS-CACHE (CHARSET-CACHE MAX.SMALLP)
    CHARSET-DESCENT-CACHE CHARHEIGHTDELTA (SPACEWIDTH 1.0)          ; a misnomer -- this is actually the space factor, not its width
    [SERIALIZING-BOX (fetch (ARRAYP BASE) of (ARRAY 1 'BYTE]
    SERIALIZING-PILOTBBT SCRATCH-SCANLINE SCRATCH-SCANLINE-PILOTBBT [EIGHT-LINES-BLANK (fetch (ARRAYP BASE)
                                                                             of (ARRAY 1
                                                                                   'WORD]
    EIGHT-LINES-BLANK-PILOTBBT COMPRESSED?)
```

(DEFINEQ

(\HQFX80.INIT
```
  [LAMBDA NIL                                                         ; Edited  3-Feb-87 17:23 by hdj

    ;; Initializes global variables for the FX80

    (DECLARE (GLOBALVARS DEFAULTPRINTINGHOST IMAGESTREAMTYPES PRINTERTYPES PRINTFILETYPES \HQFX80.IMAGEOPS
                    \HQFX80.FDEV))
    (SETQ \HQFX80.FDEV (create FDEV
                                  DEVICENAME _ (LIST 'HQFX80 'PRINTER)
                                  CLOSEFILE _ (FUNCTION NILL)
                                  BOUT _ (FUNCTION \HQFX80.OUTCHAR)))
    (SETQ \HQFX80.IMAGEOPS (create IMAGEOPS
                                  IMAGETYPE _ 'HQFX80
                                  IMFONT _ (FUNCTION \HQFX80.CHANGEFONT)
                                  IMLEFTMARGIN _ (FUNCTION \HQFX80.LEFTMARGIN)
                                  IMRIGHTMARGIN _ (FUNCTION \HQFX80.RIGHTMARGIN)
                                  IMTOPMARGIN _ (FUNCTION \HQFX80.TOPMARGIN)
                                  IMBOTTOMMARGIN _ (FUNCTION \HQFX80.BOTTOMMARGIN)
                                  IMLINEFEED _ (FUNCTION \HQFX80.LINEFEED)
                                  IMXPOSITION _ (FUNCTION \HQFX80.XPOSITION)
                                  IMYPOSITION _ (FUNCTION \HQFX80.YPOSITION)
                                  IMCLOSEFN _ (FUNCTION \HQFX80.CLOSE)
                                  IMDRAWCURVE _ (FUNCTION \HQFX80.DRAWCURVE)
                                  IMFILLCIRCLE _ (FUNCTION \HQFX80.FILLCIRCLE)
                                  IMDRAWLINE _ (FUNCTION \HQFX80.DRAWLINE)
                                  IMDRAWELLIPSE _ (FUNCTION \HQFX80.DRAWELLIPSE)
                                  IMDRAWCIRCLE _ (FUNCTION \HQFX80.DRAWCIRCLE)
                                  IMBITBLT _ (FUNCTION \HQFX80.BITBLT)
                                  IMBLTSHADE _ (FUNCTION \HQFX80.BLTSHADE)
                                  IMNEWPAGE _ (FUNCTION \HQFX80.NEWPAGE)
                                  IMSCALE _ [FUNCTION (LAMBDA NIL 1]
                                  IMSPACEFACTOR _ (FUNCTION \HQFX80.SPACEFACTOR)
                                  IMFONTCREATE _ 'HQFX80
                                  IMCOLOR _ (FUNCTION NILL)
                                  IMBACKCOLOR _ (FUNCTION NILL)
                                  IMOPERATION _ (FUNCTION \HQFX80.OPERATION)
                                  IMSTRINGWIDTH _ (FUNCTION \HQFX80.STRINGWIDTH)
                                  IMCHARWIDTH _ (FUNCTION \HQFX80.CHARWIDTH)
                                  IMCLIPPINGREGION _ (FUNCTION \HQFX80.CLIPPINGREGION)
                                  IMRESET _ (FUNCTION \HQFX80.RESET)
                                  IMDRAWPOINT _ (FUNCTION \HQFX80.DRAWPOINT)
                                  IMDRAWARC _ (FUNCTION \HQFX80.DRAWARC)
                                  IMFILLPOLYGON _ (FUNCTION POLYSHADE.BLT)))
    [push IMAGESTREAMTYPES (COPYALL '(HQFX80 (OPENSTREAM OPENHQFX80STREAM)
                                       (FONTCREATE \HQFX80.FONTCREATE)
                                       (CREATECHARSET \HQFX80.CREATECHARSET)
                                       (FONTSAVAILABLE \HQFX80.SEARCH-FONTS]
    [push PRINTERTYPES (COPYALL '((HQFX80)
                                       (CANPRINT (HQFX80))
                                       (STATUS TRUE)
                                       (PROPERTIES NILL)
                                       (SEND \HQFX80.SEND)
                                       (BITMAPSCALE NIL)
                                       (BITMAPFILE (\HQFX80.BITMAP-FILE FILE BITMAP SCALEFACTOR REGION ROTATION TITLE]
    [push PRINTFILETYPES (COPYALL '(HQFX80 (TEST HQFX80FILEP)
                                       (EXTENSION (HQFX80))
                                       (CONVERSION (TEXT MAKE-HQFX80 TEDIT \HQFX80.CONVERT-TEDIT]
    (push DEFAULTPRINTINGHOST (LIST 'HQFX80 'HQFX80))
    (PUTPROP 'HQFX80 'PRINTERTYPE 'HQFX80)
    (\HQFX80.INIT-FONT-PROFILE)
    T])
```

)

;; imagestream methods
;; opening/closing imagestream

(DEFINEQ

(OPENHQFX80STREAM
```
  [LAMBDA (FILENAME OPTIONS)                                         ; Edited 29-May-87 19:30 by Snow

    ;; Opens an imagestream on a high-quality FX80
```

```
      (LET* ([BACKING (OPENSTREAM FILENAME 'OUTPUT NIL '((SEQUENTIAL T)
                                                         (TYPE HQFX80]
             (COMPRESSED? (LISTGET OPTIONS 'COMPRESSED))
             [DOTS-PER-LINE (FIX (TIMES \HQFX80.INCHES-PER-LINE (if COMPRESSED?
                                                                   then \HQFX80.LOW-SPEED-DOUBLE-MODE-DPI
                                                                   else \HQFX80.1-TO-1-MODE-DPI]
             (DOTS-PER-PAGE (ITIMES 8 (CL:CEILING (FIX (TIMES \HQFX80.INCHES-PER-PAGE
                                                         (if COMPRESSED?
                                                           then \HQFX80.LOW-SPEED-DOUBLE-MODE-DPI
                                                           else \HQFX80.1-TO-1-MODE-DPI)))
                                        8)))
             (BACKING-BITMAP (BITMAPCREATE DOTS-PER-LINE DOTS-PER-PAGE))
             (BACKING-BITMAP-WORD-WIDTH (fetch (BITMAP BITMAPRASTERWIDTH) of BACKING-BITMAP))
             (DATA (MAKE-HQFX80DATA :BACKINGSTREAM BACKING :CLIPPINGREGION (CREATEREGION 0 0 DOTS-PER-LINE
                                                                             DOTS-PER-PAGE)
                          :BACKINGBITMAP BACKING-BITMAP :RIGHTMARGIN DOTS-PER-LINE :TOPMARGIN (- DOTS-PER-PAGE 15)
                          :BOTTOMMARGIN 30 :PILOTBBT (create PILOTBBT
                                                             PBTDISJOINT _ T
                                                             PBTDESTBPL _ (UNFOLD BACKING-BITMAP-WORD-WIDTH
                                                                                  BITSPERWORD))
                          :SCRATCH-SCANLINE
                          (fetch (BITMAP BITMAPBASE) of (BITMAPCREATE DOTS-PER-LINE 1))
                          :OPERATION
                          'REPLACE :SOURCETYPE 'INPUT :COMPRESSED? COMPRESSED?))
             (HQFX80STREAM (create STREAM
                                   FULLFILENAME _ (FULLNAME BACKING)
                                   DEVICE _ \HQFX80.FDEV
                                   ACCESS _ 'OUTPUT
                                   OUTCHARFN _ (FUNCTION \HQFX80.OUTCHAR)
                                   STRMBOUTFN _ (FUNCTION \HQFX80.OUTCHAR)
                                   IMAGEOPS _ \HQFX80.IMAGEOPS
                                   USERCLOSEABLE _ T
                                   USERVISIBLE _ T
                                   IMAGEDATA _ DATA)))
```

;; set up the BitBLT table that transforms 8-bit columns of bitmap data into single BOUT-able bytes

```
        (CL:SETF (HQFX80DATA-SERIALIZING-PILOTBBT DATA)
                 (create PILOTBBT
                         PBTDISJOINT _ T
                         PBTDEST _ (HQFX80DATA-SERIALIZING-BOX DATA)
                         PBTWIDTH _ 1
                         PBTHEIGHT _ 8
                         PBTSOURCEBPL _ (UNFOLD BACKING-BITMAP-WORD-WIDTH BITSPERWORD)
                         PBTDESTBPL _ 1))
```

;; set up the BitBLT table that ORs together eight sequential scanlines (for blank-line group detection) into one scanline

```
        (CL:SETF (HQFX80DATA-SCRATCH-SCANLINE-PILOTBBT DATA)
                 (create PILOTBBT
                         PBTDISJOINT _ T
                         PBTDEST _ (HQFX80DATA-SCRATCH-SCANLINE DATA)
                         PBTWIDTH _ (UNFOLD BACKING-BITMAP-WORD-WIDTH BITSPERWORD)
                         PBTHEIGHT _ 8
                         PBTSOURCEBPL _ (UNFOLD BACKING-BITMAP-WORD-WIDTH BITSPERWORD)
                         PBTDESTBPL _ 0
                         PBTSOURCEBIT _ 0
                         PBTDESTBIT _ 0
                         PBTOPERATION _ 2))
```

;; set up the BitBLT table that ORs one scanline into one 16-bit word

```
        (CL:SETF (HQFX80DATA-EIGHT-LINES-BLANK-PILOTBBT DATA)
                 (create PILOTBBT
                         PBTDISJOINT _ T
                         PBTSOURCE _ (HQFX80DATA-SCRATCH-SCANLINE DATA)
                         PBTDEST _ (HQFX80DATA-EIGHT-LINES-BLANK DATA)
                         PBTWIDTH _ BITSPERWORD
                         PBTHEIGHT _ (FOLDHI DOTS-PER-LINE BITSPERWORD)
                         PBTSOURCEBPL _ BITSPERWORD
                         PBTDESTBPL _ 0
                         PBTSOURCEBIT _ 0
                         PBTDESTBIT _ 0
                         PBTOPERATION _ 2))
```

;; make the backing file invisible

(**replace** (STREAM USERVISIBLE) **of** BACKING **with** NIL)

;; put the preamble on the master

```
        (\HQFX80.PREAMBLE HQFX80STREAM)
        HQFX80STREAM])
```

)


```
(CL:DEFUN \HQFX80.PREAMBLE (HQFX80STREAM)
```

    ;; start an HQFX80 master

```
    (DECLARE (GLOBALVARS \HQFX80.INCHES-PER-PAGE))
    (\HQFX80.RESET-PRINTER HQFX80STREAM \HQFX80.INCHES-PER-PAGE)
```

```
    (\HQFX80.OUTPUT-SIGNATURE HQFX80STREAM)
    (DSPFONT (DEFAULTFONT 'HQFX80)
             HQFX80STREAM)
    (\HQFX80.STARTPAGE HQFX80STREAM))
```

(CL:DEFUN **HQFX80.RESET-PRINTER** (HQFX80STREAM INCHES-PER-PAGE)

   ;; send a reset sequence to the fx-80

   (**IF** (AND (<= 1 INCHES-PER-PAGE)
           (<= INCHES-PER-PAGE 22))
      THEN ;; send a reset sequence to the fx-80...

         (**HQFX80.BOUT** HQFX80STREAM (CHARCODE ESC))
         (**HQFX80.BOUT** HQFX80STREAM (CHARCODE @))

         ;; ...and set the form length

         (**HQFX80.BOUT** HQFX80STREAM (CHARCODE ESC))
         (**HQFX80.BOUT** HQFX80STREAM (CHARCODE C))
         (**HQFX80.BOUT** HQFX80STREAM (FIXR (TIMES 6 INCHES-PER-PAGE)))
    ELSE (ERROR "Illegal page length value" INCHES-PER-PAGE)))

(CL:DEFUN **HQFX80.OUTPUT-SIGNATURE** (HQFX80TREAM)

   ;; start the file with an identifying signature. Ensure it is not printed by following it with an equal number of ASCII 127's.

   ;; This will not work if SIGNATURE contains line-ending characters.

   (LET ((DEL-BYTE 127))
      (**FOR** BYTE INSTRING \HQFX80.FILE-SIGNATURE **DO** (**HQFX80.BOUT** HQFX80TREAM BYTE))
      (**FOR** BYTE INSTRING \HQFX80.FILE-SIGNATURE **DO** (**HQFX80.BOUT** HQFX80TREAM DEL-BYTE))))

(DEFINEQ

(**HQFX80.CLOSE**
  [LAMBDA (HQFX80STREAM)   ; Edited 21-Jan-88 12:20 by jds

   ;; do the cleanup prefatory to closing the HQFX80STREAM

   (**WITH-HQFX80-DATA** (DATA HQFX80STREAM)   ;
      (LET ((BACKING-STREAM (HQFX80DATA-BACKINGSTREAM DATA)))
        (**HQFX80.DUMP-PAGE-BUFFER** (HQFX80DATA-BACKINGBITMAP DATA)
           HQFX80STREAM)
        (\BOUT BACKING-STREAM (CHARCODE ESCAPE))
        (\BOUT BACKING-STREAM (CHARCODE !))
        (\BOUT BACKING-STREAM 0)
        (\BOUT BACKING-STREAM (CHARCODE ESCAPE))
        (\BOUT BACKING-STREAM (CHARCODE 5))
        (\CLOSEFILE BACKING-STREAM)
        (**fetch** (STREAM FULLFILENAME) **of** HQFX80STREAM])

)

;; methods that hack fonts

(DEFINEQ

(**HQFX80.FONTCREATE**
  [LAMBDA (FAMILY SIZE FACE ROTATION DEVICE CHARSET)   (* hdj "10-Nov-86 11:30")

   ;; create a font for the hqfx80

   (LET [(FONTDESC (**create** FONTDESCRIPTOR
                     FONTDEVICE _ DEVICE
                     FONTFAMILY _ FAMILY
                     FONTSIZE _ SIZE
                     FONTFACE _ FACE
                     \SFAscent _ 0
                     \SFDescent _ 0
                     \SFHeight _ 0
                     ROTATION _ ROTATION
                     FONTDEVICESPEC _ (LIST FAMILY SIZE FACE ROTATION DEVICE]
     (AND (\GETCHARSETINFO CHARSET FONTDESC T)
       FONTDESC])

(**HQFX80.CHANGEFONT**
  [LAMBDA (HQFX80STREAM FONT)   ; Edited 4-Feb-87 11:48 by hdj

   ;; sets/returns the font of an HQFX80 imagestream

   (**WITH-HQFX80-DATA** (HQFX80DATA HQFX80STREAM)
      (LET ((OLD-FONT (HQFX80DATA-FONT HQFX80DATA)))

        ;; save old value to return, smash new value and update the record.

        (PROG1 OLD-FONT
          [**if** FONT
             then (LET [(NEW-FONT (OR (\COERCEFONTDESC FONT HQFX80STREAM T)
                              (FONTCOPY (HQFX80DATA-FONT HQFX80DATA)
                                   FONT]

```
                                 ;; updating font information is fairly expensive operation.  Don't bother unless font has changed.
                          (OR (EQ OLD-FONT NEW-FONT)
                              (UNINTERRUPTABLY
                                  (CL:SETF (HQFX80DATA-FONT HQFX80DATA)
                                           NEW-FONT)
                                  (CL:SETF (HQFX80DATA-LINEFEED HQFX80DATA)
                                           (IMINUS (fetch (FONTDESCRIPTOR \SFHeight) of NEW-FONT)))
                                  (\HQFX80.FIX-FONT HQFX80STREAM HQFX80DATA)]]])
```

(**HQFX80.CREATECHARSET**
```
  [LAMBDA (FAMILY SIZE FACE ROTATION DEVICE CHARSET FONTDESC NOSLUG?)
```
                                                          ; Edited 1-Jun-87 13:08 by Snow

;;; Tries to build the csinfo required for CHARSET. Does the necessary coercions.

;;; NOSLUG? means don't create an empty (slug) csinfo if the charset is not found, just return NIL.

```
        (DECLARE (GLOBALVARS HQFX80-FONT-COERCIONS HQFX80-MISSING-FONT-COERCIONS))
```
    ;; HQFX80-FONT-COERCIONS is a list of font coercions, in the form ((user-font real-font) (user-font real-font) ...). Each user-font is a list of
    ;; FAMILY, and optionally SIZE and CHARSET, (e.g., (GACHA) or (GACHA 10) or (GACHA 10 143)), and each real-font is a similar list.
```
        (PROG (XCSINFO)
              [SETQ XCSINFO (COND
                              [(PROGN
                                 ;; Just recursively call ourselves to handle entries in HQFX80-FONT-COERCIONS
                                     (for TRANSL in HQFX80-FONT-COERCIONS bind NEWCSINFO USRFONT REALFONT
                                        when (AND (SETQ USRFONT (CAR TRANSL))
                                                  (EQ FAMILY (CAR USRFONT))
                                                  (OR (NOT (CADR USRFONT))
                                                      (EQ SIZE (CADR USRFONT)))
                                                  (OR (NOT (CADDR USRFONT))
                                                      (EQ CHARSET (CADDR USRFONT)))
                                                  (SETQ REALFONT (CADR TRANSL))
                                                  (SETQ NEWCSINFO (\HQFX80.CREATECHARSET (OR (CAR REALFONT)
                                                                                             FAMILY)
                                                                       (OR (CADR REALFONT)
                                                                           SIZE)
                                                                       FACE ROTATION DEVICE (OR (CADDR REALFONT)
                                                                                                CHARSET)
                                                                       FONTDESC NOSLUG?)))
                                        do (RETURN NEWCSINFO]
                              ((AND (EQ ROTATION 0)
                                    ;; If it is available, this will force the appropriate file to be read to fill in the charset entry
                                    (\HQFX80.READ-FONT-FILE FAMILY SIZE FACE ROTATION 'HQFX80 CHARSET)))
                              (T
                                 ;; if we get here, the font is not directly available, either it needs to be rotated, boldified, or italicised 'by hand'
                                (PROG (NEWFONT XFONT XLATEDFAM CSINFO)
                                      (RETURN (COND
                                                [(NEQ ROTATION 0)
                                    ;; to make a rotated font (even if it is bold or whatnot), recursively call fontcreate to get the unrotated
                                    ;; font (maybe bold, etc), then call \SFMAKEROTATEDFONT on the csinfo.
                                                 (OR (MEMB ROTATION '(90 270))
                                                     (ERROR "only implemented rotations are 0, 90 and 270."
                                                            ROTATION))
                                                 (COND
                                                   ((SETQ XFONT (FONTCREATE FAMILY SIZE FACE 0 'HQFX80 T CHARSET
                                                                            ))
                                    ;; actually call FONTCREATE here, rather than a device-specific method, so that the vanilla font that
                                    ;; is built in this process will be cached and not repeated.
                                                    (COND
                                                      ((SETQ CSINFO (\GETCHARSETINFO CHARSET XFONT T))
                                                       (\SFROTATECSINFO CSINFO ROTATION))
                                                      (T NIL]
                                                ((AND (EQ (fetch (FONTFACE WEIGHT) of FACE)
                                                          'BOLD)
                                                      (SETQ XFONT (FONTCREATE FAMILY SIZE
                                                                              (create FONTFACE using FACE WEIGHT _
                                                                                      'MEDIUM)
                                                                       0
                                                                       'HQFX80 T CHARSET)))
                                    ;; if we want a bold font, and the medium weight font is available, build the medium weight
                                    ;; version then call \SFMAKEBOLD on the csinfo
                                                 (COND
                                                   ((SETQ CSINFO (\GETCHARSETINFO CHARSET XFONT T))
                                                    (\SFMAKEBOLD CSINFO))
                                                   (T NIL)))
                                                ((AND (EQ (fetch (FONTFACE SLOPE) of FACE)
                                                          'ITALIC)
                                                      (SETQ XFONT (FONTCREATE FAMILY SIZE
                                                                              (create FONTFACE using FACE SLOPE _
                                                                                      'REGULAR)
```

```
                                                             0
                                                            'HQFX80 T CHARSET)))
                                        (COND
                                           ((SETQ CSINFO (\GETCHARSETINFO CHARSET XFONT T))
                                            (\SFMAKEITALIC CSINFO))
                                           (T NIL)))
                                 ((for TRANSL in HQFX80-MISSING-FONT-COERCIONS
                                     bind NEWCSINFO USRFONT REALFONT
                                     when (AND (SETQ USRFONT (CAR TRANSL))
                                               (EQ FAMILY (CAR USRFONT))
                                               (OR (NOT (CADR USRFONT))
                                                   (EQ SIZE (CADR USRFONT)))
                                               (OR (NOT (CADDR USRFONT))
                                                   (EQ CHARSET (CADDR USRFONT)))
                                               (SETQ REALFONT (CADR TRANSL))
                                               (SETQ NEWCSINFO (\HQFX80.CREATECHARSET
                                                                  (OR (CAR REALFONT)
                                                                      FAMILY)
                                                                  (OR (CADR REALFONT)
                                                                      SIZE)
                                                                  FACE ROTATION DEVICE
                                                                  (OR (CADDR REALFONT)
                                                                      CHARSET)
                                                                  FONTDESC NOSLUG?)))
                                     do (RETURN NEWCSINFO)))
                                 ((NOT NOSLUG?)
                                  (\BUILDSLUGCSINFO (fetch (FONTDESCRIPTOR FONTAVGCHARWIDTH)
                                                        of FONTDESC)
                                                    (FONTPROP FONTDESC 'ASCENT)
                                                    (FONTPROP FONTDESC 'DESCENT)
                                                    (FONTPROP FONTDESC 'DEVICE]
          (RETURN XCSINFO)])
```

HQFX80.CHANGE-CHARSET

```
  [LAMBDA (HQFX80DATA CHARSET)                                    (* hdj "10-Nov-86 16:00")

    ;; Called when the character set information cached in hqfx80 stream doesn't correspond to CHARSET

    (LET* ((PBT (HQFX80DATA-PILOTBBT HQFX80DATA))
           (CSINFO (\GETCHARSETINFO CHARSET (HQFX80DATA-FONT HQFX80DATA)))
           (CHARACTER-BITMAP (ffetch CHARSETBITMAP of CSINFO)))

          ;; Since we called \GETCHARSETINFO without the NOSLUG? flag, we presume we will get back a CSINFO , even if it is a slug csinfo

          (UNINTERRUPTABLY
               (CL:SETF (HQFX80DATA-WIDTHS-CACHE HQFX80DATA)
                     (ffetch (CHARSETINFO WIDTHS) of CSINFO))
               (CL:SETF (HQFX80DATA-OFFSETS-CACHE HQFX80DATA)
                     (ffetch (CHARSETINFO OFFSETS) of CSINFO))
               (CL:SETF (HQFX80DATA-IMAGE-WIDTHS-CACHE HQFX80DATA)
                     (ffetch (CHARSETINFO IMAGEWIDTHS) of CSINFO))
               (CL:SETF (HQFX80DATA-CHARSET-CACHE HQFX80DATA)
                     CHARSET)
               (freplace PBTSOURCEBPL of PBT with (UNFOLD (ffetch (BITMAP BITMAPRASTERWIDTH) of CHARACTER-BITMAP)
                                                          BITSPERWORD))
               [IF (OR (NEQ (HQFX80DATA-CHARSET-ASCENT-CACHE HQFX80DATA)
                            (ffetch CHARSETASCENT of CSINFO))
                       (NEQ (HQFX80DATA-CHARSET-DESCENT-CACHE HQFX80DATA)
                            (ffetch CHARSETDESCENT of CSINFO)))
                   THEN (\HQFX80.FIX-Y HQFX80DATA CSINFO)
                   ELSE (freplace PBTSOURCE of PBT with (\ADDBASE (ffetch (BITMAP BITMAPBASE) of CHARACTER-BITMAP)
                                                                  (ITIMES (ffetch (BITMAP BITMAPRASTERWIDTH) of
                                                                                                    CHARACTER-BITMAP
                                                                            )
                                                                          (HQFX80DATA-CHARHEIGHTDELTA HQFX80DATA])])
```

HQFX80.READ-FONT-FILE

```
  [LAMBDA (FAMILY SIZE FACE ROTATION DEVICE CHARSET)                    ; Edited  6-Jan-87 17:52 by hdj

    ;; Look for new filename convention, then old file name convention, with extensions.  Note we assume \FONTFILENAME calls
    ;; \FONTFILENAME.NEW

    (DECLARE (GLOBALVARS HQFX80-FONT-EXTENSIONS HQFX80-FONT-DIRECTORIES))
    (bind FONTFILE CSINFO STRM for EXT inside HQFX80-FONT-EXTENSIONS
       when (SETQ FONTFILE (\FINDFONTFILE FAMILY SIZE FACE ROTATION DEVICE CHARSET HQFX80-FONT-DIRECTORIES
                                  (LIST EXT)))
       do (SETQ STRM (OPENSTREAM FONTFILE 'INPUT))
          (RESETLST
              [SETQ CSINFO (SELECTQ (FONTFILEFORMAT STRM T)
                                 (STRIKE (RESETSAVE NIL (LIST (FUNCTION CLOSEF)
                                                             STRM))
                                        (\READSTRIKEFONTFILE STRM FAMILY SIZE FACE))
                                 (AC  ;; CLOSEF is guaranteed inside \READACFONTFILE, against the possibility that we have to copy to
                                      ;; make randaccessp

                                     (\READACFONTFILE STRM FAMILY SIZE FACE))
                                 (PROG1 (CLOSEF STRM)
                                        (SHOULDNT)                         ; This would get done by RESETSAVE if AC's were read
                                                                           ; sequentially and we could factor the RESETSAVE
```

```
                                    )])
```

;; If not a recognizable format, I guess we should keep looking for another possible extension, altho it would also be nice to tell the user
;; that he has a bogus file.

```
          (RETURN CSINFO])
```

(**HQFX80.SEARCH-FONTS**
```
  [LAMBDA (FAMILY SIZE FACE ROTATION)                                    (* hdj "10-Nov-86 12:09")
```

;;; returns a list of the fonts that can be read in for the hqfx80 device. (This is the same as all fonts for the dissplay device.) Rotation is ignored because
;;; it is assumed that all devices support 0 90 and 270

```
      (DECLARE (GLOBALVARS HQFX80-FONT-EXTENSIONS HQFX80-FONT-DIRECTORIES))
      (for E FILENAMEPATTERN FONTSFOUND THISFONT THISFACE inside HQFX80-FONT-EXTENSIONS
         do (SETQ FILENAMEPATTERN (\FONTFILENAME FAMILY SIZE FACE E))
            [for DIR inside HQFX80-FONT-DIRECTORIES
               do (for FONTFILE in (DIRECTORY (PACKFILENAME 'DIRECTORY DIR 'BODY FILENAMEPATTERN))
                    do (OR (MEMBER (SETQ THISFONT (\FONTINFOFROMFILENAME FONTFILE 'DISPLAY))
                                   FONTSFOUND)
                           (COND
                              ((AND [OR (EQ FACE '*)
                                        (EQUAL FACE (SETQ THISFACE (CADDR THISFONT)))
                                        (AND (OR (EQ (CAR FACE)
                                                     '*)
                                                 (EQ (CAR FACE)
                                                     (CAR THISFACE)))
                                             (OR (EQ (CADR FACE)
                                                     '*)
                                                 (EQ (CADR FACE)
                                                     (CADR THISFACE)))
                                             (OR (EQ (CADR FACE)
                                                     '*)
                                                 (EQ (CADR FACE)
                                                     (CADR THISFACE]
                                    (OR (EQ FAMILY '*)
                                        (EQ FAMILY (CAR THISFONT))
                                        (STRPOS "*" FAMILY)))
```

;; make sure the face, size, and family really match. Family name match allows anything if the family has a * in
;; it. This is wrong but better than what was there before which let in anything with the right beginning.

```
                               (SETQ FONTSFOUND (CONS THISFONT FONTSFOUND]
         finally (RETURN FONTSFOUND])
)
```

```
(CL:DEFUN \HQFX80.INIT-FONT-PROFILE ()
```
   ;; set up the fonts for the HQFX80, based on the DISPLAY font profile entries

```
   [FOR FONT-CLASS IN '(DEFAULTFONT ITALICFONT BOLDFONT LITTLEFONT TINYFONT BIGFONT COMMENTFONT TEXTFONT)
      DO (\ADD-TO-FONTPROFILE FONTPROFILE FONT-CLASS 'HQFX80 (\GET-FROM-FONTPROFILE FONTPROFILE FONT-CLASS
                                                                                    'DISPLAY]
   (FONTPROFILE FONTPROFILE)
   T)
```

;; methods for measuring

```
(DEFINEQ
```

(**HQFX80.CHARWIDTH**
```
  [LAMBDA (HQFX80STREAM CHARCODE)                                         ; Edited  4-Feb-87 13:20 by hdj
```
   ;; gets the width of the rendering of charcode on an hqfx80 image stream. We treat space specially.

```
   (WITH-HQFX80-DATA (DATA HQFX80STREAM)
          (LET ((WIDTH (\FGETCHARWIDTH (HQFX80DATA-FONT DATA)
                                       CHARCODE)))
               (if (EQ CHARCODE (CHARCODE SPACE))
                   then (FIXR (TIMES WIDTH (HQFX80DATA-SPACEWIDTH DATA)))
                 else WIDTH])
```

(**HQFX80.STRINGWIDTH**
```
  [LAMBDA (HQFX80STREAM STRING RDTBL)                                     ; Edited  3-Feb-87 17:36 by hdj
```
   ;; returns STRING's width, relative to HQFX80STREAM's current font and the readtable RDTBL

```
   (IF RDTBL
       THEN (BIND (FIRSTFLG _ T)
                  (SA _ (FETCH READSA OF RDTBL))
                  (ESCAPE-CHAR-WIDTH _ (\HQFX80.CHARWIDTH HQFX80STREAM (FETCH (READTABLEP ESCAPECHAR)
                                                                             OF RDTBL)))
                  (SYN _ NIL) FOR CHARCODE INSTRING STRING
               SUM (PROG1 (IPLUS (COND
                                    ((AND (FETCH (READCODE ESCQUOTE) OF (SETQ SYN (\SYNCODE SA CHARCODE)))
                                          (OR FIRSTFLG (FETCH (READCODE INNERESCQUOTE) OF SYN)))
                                     ESCAPE-CHAR-WIDTH)
```

```
                                    (T 0))
                          (\FASTFX80.CHARWIDTH HQFX80STREAM CHARCODE))
                    (SETQ FIRSTFLG NIL)))
       ELSE (FOR CHAR INSTRING STRING SUM (\HQFX80.CHARWIDTH HQFX80STREAM CHAR])
)
```

```
(CL:DEFUN \HQFX80.SPACEFACTOR (HQFX80STREAM FACTOR)
```
   ;; returns/sets the width of the space character (32 ASCII) for HQFX80STREAM

```
   [WITH-HQFX80-DATA (DATA HQFX80STREAM)
         (PROG1 (HQFX80DATA-SPACEWIDTH DATA)
              (AND FACTOR (IF (NUMBERP FACTOR)
                              THEN (CL:SETF (HQFX80DATA-SPACEWIDTH DATA)
                                        FACTOR)
                              ELSE (\ILLEGAL.ARG FACTOR))))])
```

;; methods that affect the current position/size of drawing surface

```
(DEFINEQ
```

\HQFX80.CLIPPINGREGION
```
  [LAMBDA (HQFX80STREAM REGION)                                           ; Edited  8-Dec-86 14:04 by hdj
```
   ;; sets the clipping region of an HQFX80 image stream. do not allow it to exceed the confines of the bitmap.

```
      (WITH-HQFX80-DATA (HQFX80DATA HQFX80STREAM)
            (PROG1 (COPY (HQFX80DATA-CLIPPINGREGION HQFX80DATA))   ; copy so it can't be side-effected later
                [LET ((BACKING (HQFX80DATA-BACKINGBITMAP HQFX80DATA)))
                     (AND REGION (OR (type? REGION REGION)
                                     (ERROR REGION " is not a REGION."))
                          (UNINTERRUPTABLY
                               (CL:SETF (HQFX80DATA-CLIPPINGREGION HQFX80DATA)
                                     (INTERSECTREGIONS (CREATEREGION 0 0 (BITMAPWIDTH BACKING)
                                                                        (BITMAPHEIGHT BACKING))
                                         REGION))
                          (\HQFX80.INVALIDATE-FONT-CACHE HQFX80DATA))])])
```

\HQFX80.LEFTMARGIN
```
  [LAMBDA (HQFX80STREAM XPOSITION)                                        ; Edited  3-Feb-87 17:11 by hdj
```
   ;; sets/returns the position that a carriage return returns to for an hqfx80stream

```
      (WITH-HQFX80-DATA (DATA HQFX80STREAM)
            (PROG1 (HQFX80DATA-LEFTMARGIN DATA)
                 (AND XPOSITION (if (SMALLP XPOSITION)
                                    then (UNINTERRUPTABLY
                                              (CL:SETF (HQFX80DATA-LEFTMARGIN DATA)
                                                    XPOSITION)
                                          (\HQFX80.FIX-LINE-LENGTH HQFX80STREAM))
                                    else (\ILLEGAL.ARG XPOSITION))))])
```

\HQFX80.RIGHTMARGIN
```
  [LAMBDA (HQFX80STREAM XPOSITION)                                        ; Edited 10-Dec-86 18:17 by hdj
```
   ;; Sets the right margin of an HQFX80STREAM

```
      (WITH-HQFX80-DATA (DATA HQFX80STREAM)
            (PROG1 (HQFX80DATA-RIGHTMARGIN DATA)
                 (AND XPOSITION (IF (SMALLP XPOSITION)
                                    THEN (UNINTERRUPTABLY
                                              (CL:SETF (HQFX80DATA-RIGHTMARGIN DATA)
                                                    XPOSITION)
                                          (\HQFX80.FIX-LINE-LENGTH HQFX80STREAM))
                                    ELSE (\ILLEGAL.ARG XPOSITION))))])
```

\HQFX80.TOPMARGIN
```
  [LAMBDA (HQFX80STREAM YPOSITION)                                        ; Edited 10-Dec-86 18:16 by hdj
```
   ;; Sets the top margin of an hqfx80stream

```
      (WITH-HQFX80-DATA (DATA HQFX80STREAM)
            (PROG1 (HQFX80DATA-TOPMARGIN DATA)
                 (AND YPOSITION (IF (SMALLP YPOSITION)
                                    THEN (CL:SETF (HQFX80DATA-TOPMARGIN DATA)
                                              YPOSITION)
                                    ELSE (\ILLEGAL.ARG YPOSITION))))])
```

\HQFX80.BOTTOMMARGIN
```
  [LAMBDA (HQFX80STREAM YPOSITION)                                        ; Edited 10-Dec-86 18:17 by hdj
```
   ;; Sets the bottom margin of an HQFX80STREAM

```
      (WITH-HQFX80-DATA (DATA HQFX80STREAM)
            (PROG1 (HQFX80DATA-BOTTOMMARGIN DATA)
                 (AND YPOSITION (IF (SMALLP YPOSITION)
                                    THEN (CL:SETF (HQFX80DATA-BOTTOMMARGIN DATA)
```

```
                                        YPOSITION)
                    ELSE (\ILLEGAL.ARG YPOSITION)))))])
```

(**HQFX80.XPOSITION**
```
  [LAMBDA (HQFX80STREAM XPOSITION)                                (* hdj " 3-Nov-86 15:14")
    (WITH-HQFX80-DATA (DATA HQFX80STREAM)
        (PROG1 (HQFX80DATA-XPOS DATA)
            (AND XPOSITION (IF (NUMBERP XPOSITION)
                            THEN (CL:SETF (HQFX80DATA-XPOS DATA)
                                        XPOSITION)
                            ELSE (\ILLEGAL.ARG XPOSITION)))))])
```

(**HQFX80.YPOSITION**
```
  [LAMBDA (HQFX80STREAM YPOSITION)                                ; Edited  5-Jan-87 17:25 by hdj
```
    ;; set the y-pos of an HQFX80STREAM
```
    (WITH-HQFX80-DATA (DATA HQFX80STREAM)
        (PROG1 (HQFX80DATA-YPOS DATA)
            (AND YPOSITION (if (NUMBERP YPOSITION)
                            then (UNINTERRUPTABLY
                                    (CL:SETF (HQFX80DATA-YPOS DATA)
                                            YPOSITION)
                                    (\HQFX80.INVALIDATE-CACHE DATA))
                            else (\ILLEGAL.ARG YPOSITION)))))])
```

(**HQFX80.NEWLINE**
```
  [LAMBDA (CHARCODE HQFX80STREAM)                                 (* hdj "14-Nov-86 17:44")
```
    ;; CODE is EOL, CR, or LF. Performs the appropriate printing operation on hqfx80stream.
```
    (WITH-HQFX80-DATA (DATA HQFX80STREAM)
        (LET [(NEW-Y (+ (HQFX80DATA-YPOS DATA)
                        (HQFX80DATA-LINEFEED DATA]
            (if (< NEW-Y (HQFX80DATA-BOTTOMMARGIN DATA))
```
                then ;; we're below the bottom margin, so eject the page. If this was a LF, restore the old x-position
```
                        (LET ((OLD-X (HQFX80DATA-XPOS DATA)))
                            (\HQFX80.NEWPAGE HQFX80STREAM)
                            (if (EQ CHARCODE (CHARCODE LF))
                                then (\HQFX80.XPOSITION HQFX80STREAM OLD-X)))
```
                else ;; just decrement the y coord
```
                        (\HQFX80.YPOSITION HQFX80STREAM NEW-Y)
```
                ;; if this was a CR or EOL, set the x-position too.
```
                        (if (NEQ CHARCODE (CHARCODE LF))
                            then (\HQFX80.XPOSITION HQFX80STREAM (HQFX80DATA-LEFTMARGIN DATA))
                                (freplace (STREAM CHARPOSITION) of HQFX80STREAM with 0])
```

(**HQFX80.NEWPAGE**
```
  [LAMBDA (HQFX80STREAM)                                          ; Edited  8-Dec-86 15:18 by hdj
```
    ;; end an HQFX80 page
```
    (WITH-HQFX80-DATA (DATA HQFX80STREAM)
        (\HQFX80.DUMP-PAGE-BUFFER (HQFX80DATA-BACKINGBITMAP DATA)
            HQFX80STREAM)
```
        ;; start a new page
```
        (\HQFX80.STARTPAGE HQFX80STREAM])
```

(**HQFX80.LINEFEED**
```
  [LAMBDA (HQFX80STREAM DELTAY)                                   (* hdj " 3-Nov-86 14:58")
```
    ;; Sets the linefeed distance for an HQFX80 stream
```
    (WITH-HQFX80-DATA (DATA HQFX80STREAM)
        (PROG1 (HQFX80DATA-LINEFEED DATA)
            (AND DELTAY (IF (NUMBERP DELTAY)
                        THEN (CL:SETF (HQFX80DATA-LINEFEED DATA)
                                    DELTAY)
                        ELSE (\ILLEGAL.ARG DELTAY)))))])
```

(**HQFX80.RESET**
```
  [LAMBDA (HQFX80STREAM)                                          (* hdj " 4-Nov-86 15:35")
```
    ;; resets an hqfx80 image stream to a virgin state
```
    (\HQFX80.STARTPAGE HQFX80STREAM])
```

(**HQFX80.STARTPAGE**
```
  [LAMBDA (HQFX80STREAM)                                          ; Edited 18-Dec-86 15:25 by hdj
```
    ;; start a new page for an HQFX80 imagestream

```
     (WITH-HQFX80-DATA (DATA HQFX80STREAM)
            (LET* ((CLIPPINGREGION (HQFX80DATA-CLIPPINGREGION DATA))
                   (FONT (HQFX80DATA-FONT DATA))
                   (FONT-ASCENT (FONTASCENT FONT)))
```

;; first clear the backing bitmap...

```
                (BLTSHADE (HQFX80DATA-TEXTURE DATA)
                       (HQFX80DATA-BACKINGBITMAP DATA)
                     NIL NIL NIL NIL 'REPLACE)
```

;; ... and then reset the current position

```
                   (\HQFX80.XPOSITION HQFX80STREAM (HQFX80DATA-LEFTMARGIN DATA))
                   (\HQFX80.YPOSITION HQFX80STREAM (ADD1 (- (HQFX80DATA-TOPMARGIN DATA)
                                                          FONT-ASCENT])
```

)


```
(DEFMACRO \HQFX80.CUR-POS-VISIBLE? (HQFX80DATA)
   '(INSIDEP (HQFX80DATA-CLIPPINGREGION ,HQFX80DATA)
          (HQFX80DATA-XPOS ,HQFX80DATA)
          (HQFX80DATA-YPOS ,HQFX80DATA)))
```

;; graphical operations

```
(DECLARE%: EVAL@COMPILE

[PUTDEF '\HQFX80.BRUSHBBT 'RESOURCES '(NEW (CREATE PILOTBBT]
)

(DEFINEQ
```

(**HQFX80.BITBLT**
```
  [LAMBDA (SOURCEBITMAP SOURCELEFT SOURCEBOTTOM HQFX80STREAM DESTINATIONLEFT DESTINATIONBOTTOM WIDTH HEIGHT
              SOURCETYPE OPERATION TEXTURE CLIPPINGREGION CLIPPEDSOURCELEFT CLIPPEDSOURCEBOTTOM)
                                                            ; Edited  1-Jun-87 13:07 by Snow
```

;;; BITBLT onto the HQFX80 page

;;;

```
     (DECLARE (LOCALVARS . T))
     (COND
        ((NEQ 1 (fetch (BITMAP BITMAPBITSPERPIXEL) of SOURCEBITMAP))

       ;; going from color bitmap into black and white bitmap.

        (ERROR "Cannot BitBLT a color bitmap onto the FX-80 page")))
     (WITH-HQFX80-DATA (DATA HQFX80STREAM)
         (PROG (SOURCE-TO-DEST-X SOURCE-TO-DEST-Y LEFT TOP BOTTOM RIGHT DESTBITMAP)
               (SETQ DESTBITMAP (HQFX80DATA-BACKINGBITMAP DATA))
               [LET ((FXCLIPPINGREGION (HQFX80DATA-CLIPPINGREGION DATA)))

                  ;; compute limits based on clipping regions.

                  (SETQ LEFT (fetch (REGION LEFT) of FXCLIPPINGREGION))
                  (SETQ BOTTOM (fetch (REGION BOTTOM) of FXCLIPPINGREGION))
                  (SETQ RIGHT (fetch (REGION RIGHT) of FXCLIPPINGREGION))
                  (SETQ TOP (fetch (REGION TOP) of FXCLIPPINGREGION))
                  (COND
                     (CLIPPINGREGION

                        ;; hard case, two destination clipping regions: do calculations to merge them.

                        (PROG (CRLEFT CRBOTTOM)
                              [SETQ LEFT (IMAX LEFT (SETQ CRLEFT (fetch (REGION LEFT) of CLIPPINGREGION]
                              [SETQ BOTTOM (IMAX BOTTOM (SETQ CRBOTTOM (fetch (REGION BOTTOM)
                                                                            of CLIPPINGREGION]
                              [SETQ RIGHT (IMIN RIGHT (IPLUS CRLEFT (fetch (REGION WIDTH) of
                                                                            CLIPPINGREGION
                                                                            ]
                              (SETQ TOP (IMIN TOP (IPLUS CRBOTTOM (fetch (REGION HEIGHT) of
                                                                            CLIPPINGREGION
                                                                            ]
```
;; left, right top and bottom are the limits in destination taking into account Clipping Regions. Clip to region in the arguments of this call.
```
                     (PROGN (SETQ LEFT (IMAX DESTINATIONLEFT LEFT))
                            (SETQ BOTTOM (IMAX DESTINATIONBOTTOM BOTTOM))
                            (AND WIDTH (SETQ RIGHT (IMIN (IPLUS DESTINATIONLEFT WIDTH)
                                                     RIGHT)))     ; WIDTH is optional
                            (AND HEIGHT (SETQ TOP (IMIN (IPLUS DESTINATIONBOTTOM HEIGHT)
                                                     TOP)))        ; HEIGHT is optional

                     )                                             ; Clip and translate coordinates.
                  (SETQ SOURCE-TO-DEST-X (IDIFFERENCE DESTINATIONLEFT SOURCELEFT))
                  (SETQ SOURCE-TO-DEST-Y (IDIFFERENCE DESTINATIONBOTTOM SOURCEBOTTOM))
```
;; compute the source dimensions (left right bottom top) by intersecting the source bit map, the source area to be moved with the limits
;; of the region to be moved in the destination coordinates.
```
                  [PROGN                                          ; compute left margin
```

```
                          (SETQ LEFT (IMAX CLIPPEDSOURCELEFT (IDIFFERENCE LEFT SOURCE-TO-DEST-X)
                                         0))                              ; compute bottom margin
                          (SETQ BOTTOM (IMAX CLIPPEDSOURCEBOTTOM (IDIFFERENCE BOTTOM SOURCE-TO-DEST-Y)
                                           0))
                          [PROGN                                          ; compute right margin
                              (SETQ RIGHT (IMIN (ffetch (BITMAP BITMAPWIDTH) of SOURCEBITMAP)
                                                (IDIFFERENCE RIGHT SOURCE-TO-DEST-X)
                                                (IPLUS CLIPPEDSOURCELEFT WIDTH]
                          (PROGN                                          ; compute top margin
                              (SETQ TOP (IMIN (ffetch (BITMAP BITMAPHEIGHT) of SOURCEBITMAP)
                                              (IDIFFERENCE TOP SOURCE-TO-DEST-Y)
                                              (IPLUS CLIPPEDSOURCEBOTTOM HEIGHT]
                 (COND
                     ((OR (ILEQ RIGHT LEFT)
                          (ILEQ TOP BOTTOM))                              ; there is nothing to move.
                      (RETURN)))
                 (OR OPERATION (SETQ OPERATION (HQFX80DATA-OPERATION DATA)))
                 (SELECTQ SOURCETYPE
                     (MERGE                                              ; Need to use complement of TEXTURE
                          [SETQ TEXTURE (COND
                                          ((NULL TEXTURE)
                                           BLACKSHADE)
                                          ((FIXP TEXTURE)
                                           (LOGXOR (LOGAND TEXTURE BLACKSHADE)
                                                   BLACKSHADE))
                                          [(type? BITMAP TEXTURE)
                                           (INVERT.TEXTURE.BITMAP TEXTURE (OR \BBSCRATCHTEXTURE
                                                                                (SETQ \BBSCRATCHTEXTURE
                                                                                  (BITMAPCREATE 16 16]
                                          (T (\ILLEGAL.ARG TEXTURE]
                          NIL)
                     (UNINTERRUPTABLY
                         [PROG ([PILOTBBT (COND
                                            ((type? PILOTBBT \SYSPILOTBBT)
                                             \SYSPILOTBBT)
                                            (T (SETQ \SYSPILOTBBT (create PILOTBBT]
                                (HEIGHT (IDIFFERENCE TOP BOTTOM))
                                (WIDTH (IDIFFERENCE RIGHT LEFT))
                                (DTY (\SFInvert DESTBITMAP (IPLUS TOP SOURCE-TO-DEST-Y)))
                                (DLX (IPLUS LEFT SOURCE-TO-DEST-X))
                                (STY (\SFInvert SOURCEBITMAP TOP))
                                (SLX LEFT))
                               (replace PBTWIDTH of PILOTBBT with WIDTH)
                               (replace PBTHEIGHT of PILOTBBT with HEIGHT)
                               (COND
                                   ((EQ SOURCETYPE 'MERGE)
                                    (\BITBLT.MERGE PILOTBBT SOURCEBITMAP SLX STY DESTBITMAP DLX DTY WIDTH HEIGHT
                                           OPERATION TEXTURE))
                                   (T (\BITBLTSUB PILOTBBT SOURCEBITMAP SLX STY DESTBITMAP DLX DTY HEIGHT SOURCETYPE
                                             OPERATION TEXTURE])
                         (RETURN T]))
```

HQFX80.BLTSHADE

```
  [LAMBDA (TEXTURE HQFX80STREAM DESTINATIONLEFT DESTINATIONBOTTOM WIDTH HEIGHT OPERATION CLIPPINGREGION)
                                                                   ; Edited  1-Jun-87 13:05 by Snow

    ;; BLTSHADE to an HQFX80 imagestream

    (DECLARE (LOCALVARS . T))
    (WITH-HQFX80-DATA (DATA HQFX80STREAM)
        (PROG (LEFT TOP BOTTOM RIGHT DESTINATIONBITMAP)

          ;; compute limits based on clipping regions.

            (LET ((FXCLIPPINGREGION (HQFX80DATA-CLIPPINGREGION DATA)))
                 (SETQ LEFT (fetch (REGION LEFT) of FXCLIPPINGREGION))
                 (SETQ BOTTOM (fetch (REGION BOTTOM) of FXCLIPPINGREGION))
                 (SETQ RIGHT (fetch (REGION RIGHT) of FXCLIPPINGREGION))
                 (SETQ TOP (fetch (REGION TOP) of FXCLIPPINGREGION)))
            [COND
                (CLIPPINGREGION

                      ;; hard case, two destination clipping regions: do calculations to merge them.

                      (PROG (CRLEFT CRBOTTOM)
                            [SETQ LEFT (IMAX LEFT (SETQ CRLEFT (fetch (REGION LEFT) of CLIPPINGREGION]
                            [SETQ BOTTOM (IMAX BOTTOM (SETQ CRBOTTOM (fetch (REGION BOTTOM) of
                                                                              CLIPPINGREGION
                                                                              ]
                            [SETQ RIGHT (IMIN RIGHT (IPLUS CRLEFT (fetch (REGION WIDTH) of CLIPPINGREGION]
                            [SETQ TOP (IMIN TOP (IPLUS CRBOTTOM (fetch (REGION HEIGHT) of CLIPPINGREGION]
                 (SETQ DESTINATIONBITMAP (HQFX80DATA-BACKINGBITMAP DATA)))
          ;; left, right top and bottom are the limits in destination taking into account Clipping Regions.  Clip to region in the arguments of this call.

            (SETQ LEFT (IMAX DESTINATIONLEFT LEFT))
            (SETQ BOTTOM (IMAX DESTINATIONBOTTOM BOTTOM))
            (AND WIDTH (SETQ RIGHT (IMIN (IPLUS DESTINATIONLEFT WIDTH)
                                         RIGHT)))                  ; WIDTH is optional
            (AND HEIGHT (SETQ TOP (IMIN (IPLUS DESTINATIONBOTTOM HEIGHT)
```

```
                                        TOP)))                      ; HEIGHT is optional
                (COND
                   ((AND (IGREATERP RIGHT LEFT)
                         (IGREATERP TOP BOTTOM)))
                   (T                                               ; there is nothing to move.
                      (RETURN NIL)))
                (CL:ETYPECASE TEXTURE
                    (LITATOM                                        ; includes NIL case
                        (COND
                           ((NULL TEXTURE)                          ; default texture to background texture.
                             (SETQ TEXTURE (HQFX80DATA-TEXTURE DATA)))
                           (T (\ILLEGAL.ARG TEXTURE))))
                    (SMALLP (SETQ TEXTURE (LOGAND TEXTURE BLACKSHADE)))
                    (FIXP (SETQ TEXTURE (LOGAND TEXTURE BLACKSHADE)))
                    (BITMAP NIL))
                (UNINTERRUPTABLY
                    (LET ([PILOTBBT (IF (type? PILOTBBT \SYSPILOTBBT)
                                        THEN \SYSPILOTBBT
                                        ELSE (SETQ \SYSPILOTBBT (create PILOTBBT]
                          (HEIGHT (IDIFFERENCE TOP BOTTOM)))
                         (replace PBTWIDTH of PILOTBBT with (IDIFFERENCE RIGHT LEFT))
                         (replace PBTHEIGHT of PILOTBBT with HEIGHT)
                         (\BITBLTSUB PILOTBBT NIL LEFT NIL DESTINATIONBITMAP LEFT (\SFInvert DESTINATIONBITMAP
                                                                                             TOP)
                                     HEIGHT
                                     'TEXTURE
                                     (OR OPERATION (HQFX80DATA-OPERATION DATA))
                                     TEXTURE)))
                (RETURN T])
```

(**HQFX80.DRAWELLIPSE**

```
  [LAMBDA (HQFX80STREAM CENTERX CENTERY SEMIMINORRADIUS SEMIMAJORRADIUS ORIENTATION BRUSH DASHING)
                                                    ; Edited 12-Feb-87 14:37 by jds
     (DECLARE (LOCALVARS . T))

  ;; Draws an ellipse.  At ORIENTATION 0, the semimajor axis is horizontal, the semiminor axis vertical.  Orientation is positive in the
  ;; counterclockwise direction.  The current location in the stream is left at the center of the ellipse.

     (PROG ((CENTERX (FIXR CENTERX))
            (CENTERY (FIXR CENTERY))
            (SEMIMINORRADIUS (FIXR SEMIMINORRADIUS))
            (SEMIMAJORRADIUS (FIXR SEMIMAJORRADIUS)))
           (COND
              ((OR (EQ 0 SEMIMINORRADIUS)
                   (EQ 0 SEMIMAJORRADIUS))
               (MOVETO CENTERX CENTERY HQFX80STREAM)
               (RETURN)))
           (COND
              ((ILESSP SEMIMINORRADIUS 1)
               (\ILLEGAL.ARG SEMIMINORRADIUS))
              ((ILESSP SEMIMAJORRADIUS 1)
               (\ILLEGAL.ARG SEMIMAJORRADIUS))
              ((OR (NULL ORIENTATION)
                   (EQ SEMIMINORRADIUS SEMIMAJORRADIUS))
               (SETQ ORIENTATION 0))
              ((NULL (NUMBERP ORIENTATION))
               (\ILLEGAL.ARG ORIENTATION)))
  ;; This function is the implementation of the algorithm given in 'Algorithm for drawing ellipses or hyperbolae with a digital plotter' by Pitteway
  ;; appearing in Computer Journal 10: (3) Nov 1967. The input parameters are used to determine the ellipse equation (1/8) Ayy+ (1/8) Bxx+ (1/4)
  ;; Gxy+ (1/4) Ux+ (1/4) Vy= (1/4) K which specifies a translated version of the desired ellipse.  This ellipse passes through the mesh point (0,0),
  ;; the initial point of the algorithm.  The power of 2 factors reflect an implementation convenience.
           (WITH-HQFX80-DATA (HQFX80DATA HQFX80STREAM)
               (GLOBALRESOURCE \HQFX80.BRUSHBBT
                   (PROG (DESTINATION-BITMAP LEFT RIGHTPLUS1 BOTTOM TOP BOTTOMMINUSBRUSH TOPMINUSBRUSH
                          LEFTMINUSBRUSH DESTINATIONBASE BRUSHBASE BRUSHHEIGHT BRUSHWIDTH RASTERWIDTH
                          BRUSHRASTERWIDTH BRUSHBM OPERATION HEIGHTMINUS1 (BBT \HQFX80.BRUSHBBT)
                          (COS-ORIENTATION (COS ORIENTATION))
                          (SIN-ORIENTATION (SIN ORIENTATION))
                          (SEMIMINORRADIUSSQUARED (ITIMES SEMIMINORRADIUS SEMIMINORRADIUS))
                          (SEMIMAJORRADIUSSQUARED (ITIMES SEMIMAJORRADIUS SEMIMAJORRADIUS))
                          (x 0)
                          (y 0)
                          (x2 1)
                          x1 y1 y2 k1 k2 k3 a b d w A B G U V K CX CY yOffset CYPlusOffset
                          CYMinusOffset (USERFN (AND (LITATOM BRUSH)
                                                     BRUSH)))
                         [COND
                            (USERFN                                ; if calling user fn, don't bother with set up
                                 (SETQ CX CENTERX)
                                 (SETQ CY CENTERY))
                            (T (.SETUP.FOR.\HQFX80.BBTCURVEPT. HQFX80DATA)
                                                               ; take into account the brush thickness.
                                 (SETQ CX (- CENTERX (FOLDLO BRUSHWIDTH 2)))
                                 (SETQ CY (- CENTERY (FOLDLO BRUSHHEIGHT 2]
                         (SETQ A (FPLUS (FTIMES SEMIMAJORRADIUSSQUARED COS-ORIENTATION COS-ORIENTATION)
                                        (FTIMES SEMIMINORRADIUSSQUARED SIN-ORIENTATION SIN-ORIENTATION)))
```

```
                        (SETQ B (LSH (FIXR (FPLUS (FTIMES SEMIMINORRADIUSSQUARED COS−ORIENTATION
                                                         COS−ORIENTATION)
                                                  (FTIMES SEMIMAJORRADIUSSQUARED SIN−ORIENTATION
                                                         SIN−ORIENTATION)))
                                     3))
                        (SETQ G (FTIMES COS−ORIENTATION SIN−ORIENTATION (LSH (− SEMIMINORRADIUSSQUARED
                                                                               SEMIMAJORRADIUSSQUARED)
                                                                            1)))
                        [SETQ yOffset (FIXR (FQUOTIENT (ITIMES SEMIMINORRADIUS SEMIMAJORRADIUS)
                                                      (SQRT A]
                        (SETQ CYPlusOffset (+ CY yOffset))
                        (SETQ CYMinusOffset (− CY yOffset))
                        (SETQ U (LSH (FIXR (FTIMES A (LSH yOffset 1)))
                                     2))
                        (SETQ V (LSH (FIXR (FTIMES G yOffset))
                                     2))
                        (SETQ K (LSH [FIXR (FDIFFERENCE (ITIMES SEMIMINORRADIUSSQUARED
                                                                SEMIMAJORRADIUSSQUARED)
                                                       (FTIMES A (ITIMES yOffset yOffset]
                                     2))
                        (SETQ A (LSH (FIXR A)
                                     3))
                        (SETQ G (LSH (FIXR G)
                                     2))
```

;; The algorithm is incremental and iterates through the octants of a cartesian plane. The octants are labeled from 1
;; through 8 beginning above the positive X axis and proceeding counterclockwise. Decisions in making the
;; incremental steps are determined according to the error term d which is updated according to the curvature terms a
;; and b. k1, k2, and k3 are used to correct the error and curvature terms at octant boundaries. The initial values of
;; these terms depends on the octant in which drawing begins. The initial move steps (x1,y1) and (x2,y2) also depend
;; on the starting octant.

```
                        [COND
                           [(ILESSP (ABS U)
                                    (ABS V))
                            (SETQ x1 0)
                            (COND
                               [(MINUSP V)                       ; start in octant 2
                                (SETQ y1 1)
                                (SETQ y2 1)
                                (SETQ k1 (IMINUS A))
                                (SETQ k2 (− k1 G))
                                (SETQ k3 (− k2 (+ B G)))
                                (SETQ b (+ U (RSH (+ A G)
                                                  1)))
                                (SETQ a (IMINUS (+ b V)))
                                (SETQ d (+ b (RSH B 3)
                                            (RSH V 1)
                                            (IMINUS K]
                               (T                               ; start in octant 7
                                  (SETQ y1 −1)
                                  (SETQ y2 −1)
                                  (SETQ k1 A)
                                  (SETQ k2 (− k1 G))
                                  (SETQ k3 (+ k2 B (IMINUS G)))
                                  (SETQ b (+ U (RSH (− G A)
                                                    1)))
                                  (SETQ a (− V b))
                                  (SETQ d (+ b K (IMINUS (+ (RSH V 1)
                                                            (RSH B 3]
                           (T (SETQ x1 1)
                              (SETQ y1 0)
                              (COND
                                 [(MINUSP V)                     ; start in octant 1
                                  (SETQ y2 1)
                                  (SETQ k1 B)
                                  (SETQ k2 (+ k1 G))
                                  (SETQ k3 (+ k2 A G))
                                  [SETQ b (IMINUS (+ V (RSH (+ B G)
                                                            1]
                                  (SETQ a (− U b))
                                  (SETQ d (+ b K (IMINUS (+ (RSH A 3)
                                                            (RSH U 1]
                                 (T                             ; start in octant 8
                                    (SETQ y2 −1)
                                    (SETQ k1 (IMINUS B))
                                    (SETQ k2 (+ k1 G))
                                    (SETQ k3 (+ k2 G (IMINUS A)))
                                    (SETQ b (+ V (RSH (− B G)
                                                      1)))
                                    (SETQ a (− U b))
                                    (SETQ d (+ b (RSH A 3)
                                                (IMINUS (+ K (RSH U 1]
```

;; The ellipse equation describes an ellipse of the desired size and ORIENTATION centered at (0,0) and then dropped
;; yOffset mesh points so that it will pass through (0,0). Thus, the intended starting point is (CX, CY+yOffset) where
;; (CX, CY) is the center of the desired ellipse. Drawing is accomplished with point relative steps. In each octant, the
;; error term d is used to choose between move 1 (an axis move) and move 2 (a diagonal move).

```
                        MOVE
                            [COND
                                ((MINUSP d)                              ; move 1
                                 (SETQ x (+ x x1))
                                 (SETQ y (+ y y1))
                                 (SETQ b (− b k1))
                                 (SETQ a (+ a k2))
                                 (SETQ d (+ b d)))
                                (T                                       ; move 2
                                   (SETQ x (+ x x2))
                                   (SETQ y (+ y y2))
                                   (SETQ b (− b k2))
                                   (SETQ a (+ a k3))
                                   (SETQ d (− d a]
                            (COND
                                ((MINUSP x)
                                 (MOVETO CENTERX CENTERY HQFX80STREAM)
                                 (RETURN NIL)))
                            [COND
                                (USERFN (APPLY* USERFN (+ CX x)
                                                (+ CYPlusOffset y)
                                                HQFX80STREAM)
                                        (APPLY* USERFN (− CX x)
                                                (− CYMinusOffset y)
                                                HQFX80STREAM))
                                (T (\HQFX80.CURVEPT (+ CX x)
                                           (+ CYPlusOffset y))
                                   (\HQFX80.CURVEPT (− CX x)
                                           (− CYMinusOffset y]
                            (AND (MINUSP b)
                                 (GO SQUARE))
                        DIAGONAL
                            (OR (MINUSP a)
                                (GO MOVE))                               ; diagonal octant change
                            (SETQ x1 (− x2 x1))
                            (SETQ y1 (− y2 y1))
                            (SETQ w (− (LSH k2 1)
                                       k3))
                            (SETQ k1 (− w k1))
                            (SETQ k2 (− k2 k3))
                            (SETQ k3 (IMINUS k3))
                            [SETQ b (+ b a (IMINUS (RSH (ADD1 k2)
                                                        1]
                            [SETQ d (+ b (RSH (+ k3 4)
                                              3)
                                         (IMINUS d)
                                         (IMINUS (RSH (ADD1 a)
                                                      1]
                            (SETQ a (− (RSH (ADD1 w)
                                            1)
                                       a))
                            (OR (MINUSP b)
                                (GO MOVE))
                        SQUARE
                                                                        ; square octant change
                            [COND
                                ((EQ 0 x1)
                                 (SETQ x2 (IMINUS x2)))
                                (T (SETQ y2 (IMINUS y2]
                            (SETQ w (− k2 k1))
                            (SETQ k1 (IMINUS k1))
                            (SETQ k2 (+ w k1))
                            (SETQ k3 (− (LSH w 2)
                                        k3))
                            (SETQ b (− (IMINUS b)
                                       w))
                            (SETQ d (− (− b a)
                                       d))
                            (SETQ a (− (− a w)
                                       (LSH b 1)))
                            (GO DIAGONAL])
```

(**HQFX80.OPERATION**

```
  [LAMBDA (HQFX80STREAM OPERATION)                                      (* hdj " 4-Nov-86 17:25")

    ;; sets the operation field of an hqfx80 stream

    (WITH-HQFX80-DATA (DATA HQFX80STREAM)
          (PROG1 (HQFX80DATA−OPERATION DATA)
              (AND OPERATION (OR (FMEMB OPERATION '(PAINT REPLACE INVERT ERASE))
                                 (\ILLEGAL.ARG OPERATION))
                  (UNINTERRUPTABLY
                      (CL:SETF (HQFX80DATA−OPERATION DATA)
                               OPERATION)                               ; update other fields that depend on operation.
                      (\SETPBTFUNCTION (HQFX80DATA−PILOTBBT DATA)
                           (HQFX80DATA−SOURCETYPE DATA)
                           OPERATION))))])
```

(**HQFX80.DRAWPOINT**
```
  [LAMBDA (HQFX80STREAM X Y BRUSH OPERATION)                              (* hdj "19-Nov-86 15:21")
```
    ;; draws a brush point at position X Y on an HQFX80STREAM

```
    (LET ((BRUSHBM (\GETBRUSH BRUSH)))                              ; SUB1 is to put extra bit of even brush on the top or left.
        (BITBLT BRUSHBM 0 0 HQFX80STREAM [IDIFFERENCE X (HALF (SUB1 (BITMAPWIDTH BRUSHBM]
            [IDIFFERENCE Y (HALF (SUB1 (BITMAPHEIGHT BRUSHBM]
            NIL NIL NIL OPERATION])
```
)

(DEFINEQ

(**HQFX80.DRAWLINE**
```
  [LAMBDA (HQFX80STREAM X1 Y1 X2 Y2 WIDTH OPERATION COLOR DASHING)
                                                                 ; Edited 5-Jan-87 18:10 by hdj
```
    ;; Draws a line from (x1,y1) to (x2,y2) on an hqfx80 imagestream, leaving the position at (x2,y2).

```
    (WITH-HQFX80-DATA (DATA HQFX80STREAM)
        (LET ((CLIPPINGREGION (HQFX80DATA-CLIPPINGREGION DATA)))
            ;; draw the line ...
            (if DASHING
                then (GLOBALRESOURCE \HQFX80.BRUSHBBT (LET ((BBT \HQFX80.BRUSHBBT)
                                                            (BRUSH (LIST 'ROUND WIDTH COLOR)))
                                                        (\HQFX80.LINEWITHBRUSH (OR (FIXP X1)
                                                                                   (FIXR X1))
                                                            (OR (FIXP Y1)
                                                                (FIXR Y1))
                                                            (OR (FIXP X2)
                                                                (FIXR X2))
                                                            (OR (FIXP Y2)
                                                                (FIXR Y2))
                                                            BRUSH
                                                            (\GOOD.DASHLST DASHING BRUSH)
                                                            HQFX80STREAM BBT)))
                else (\HQFX80.CLIP-AND-DRAW-LINE (OR (FIXP X1)
                                                     (FIXR X1))
                        (OR (FIXP Y1)
                            (FIXR Y1))
                        (OR (FIXP X2)
                            (FIXR X2))
                        (OR (FIXP Y2)
                            (FIXR Y2))
                        [COND
                            ((NULL WIDTH)
                             1)
                            ((OR (FIXP WIDTH)
                                 (FIXR WIDTH]
                        (SELECTQ OPERATION
                            (NIL (HQFX80DATA-OPERATION DATA))
                            ((REPLACE PAINT INVERT ERASE)
                                OPERATION)
                            (\ILLEGAL.ARG OPERATION))
                        (HQFX80DATA-BACKINGBITMAP DATA)
                        (ffetch (REGION LEFT) of CLIPPINGREGION)
                        (SUB1 (ffetch (REGION RIGHT) of CLIPPINGREGION))
                        (ffetch (REGION BOTTOM) of CLIPPINGREGION)
                        (SUB1 (ffetch (REGION TOP) of CLIPPINGREGION))
                        HQFX80STREAM))
            ;; ... then move to (x2,y2)
            (\HQFX80.XPOSITION HQFX80STREAM X2)
            (\HQFX80.YPOSITION HQFX80STREAM Y2])
```

(**HQFX80.CLIP-AND-DRAW-LINE**
```
  [LAMBDA (X1 Y1 X2 Y2 WIDTH OPERATION BITMAP LEFT RIGHT BOTTOM TOP HQFX80STREAM)
                                                                 ; Edited 5-Jan-87 17:59 by hdj
```
    ;; draws a line from (X1,Y1) to (X2,Y2) clipped to region specified by LEFT RIGHT BOTTOM and TOP. This code is a transliterated version of the
    ;; BCPL routine that was in chat.
    ;;
    ;; assumes that the width is at least 1

```
    (PROG NIL
        (COND
            [(EQP X1 X2)                                           ; special case of vertical line.
             [COND
                ((IGREATERP WIDTH 2)
                 (COND
                    ((EQP Y1 Y2)
                        ;; special case.  Since we don't know whether the guy is headed horizontally or vertically, put out a round brush.
                        (RETURN (\HQFX80.DRAWPOINT HQFX80STREAM X1 Y1 (LIST 'ROUND WIDTH)
```

```
                                                  OPERATION)))
                            (T (SETQ X1 (SETQ X2 (IDIFFERENCE X1 (LRSH (SUB1 WIDTH)
                                                                        1]
                    (PROG (MIN MAX)
                          (RETURN (COND
                                    ([OR (IGREATERP X1 RIGHT)
                                         (IGEQ LEFT (SETQ X2 (IPLUS X1 WIDTH)))
                                         (IGREATERP (SETQ MIN (IMIN Y1 Y2))
                                                    TOP)
                                         (IGREATERP BOTTOM (SETQ MAX (IMAX Y1 Y2]
                                                                                   ; outside clippingregion.
                                     NIL)
                                    (T (BLTSHADE BLACKSHADE BITMAP (SETQ X1 (IMAX X1 LEFT))
                                                 (SETQ MIN (IMAX MIN BOTTOM))
                                                 (IDIFFERENCE (IMIN X2 (ADD1 RIGHT))
                                                              X1)
                                                 (ADD1 (IDIFFERENCE (IMIN MAX TOP)
                                                                    MIN))
                                                 OPERATION]
                  [(EQP Y1 Y2)                                              ; special case of horizontal line.
                    [COND
                       ((IGREATERP WIDTH 2)
                        (SETQ Y1 (SETQ Y2 (IDIFFERENCE Y1 (LRSH (SUB1 WIDTH)
                                                                1]
                    (PROG (MIN MAX)
                          (RETURN (COND
                                    ([OR (IGREATERP Y1 TOP)
                                         (IGEQ BOTTOM (SETQ Y2 (IPLUS Y1 WIDTH)))
                                         (IGREATERP (SETQ MIN (IMIN X1 X2))
                                                    RIGHT)
                                         (IGREATERP LEFT (SETQ MAX (IMAX X1 X2]
                                                                               ; outside clippingregion.
                                     NIL)
                                    (T (BLTSHADE BLACKSHADE BITMAP (SETQ MIN (IMAX MIN LEFT))
                                                 (SETQ Y1 (IMAX Y1 BOTTOM))
                                                 (ADD1 (IDIFFERENCE (IMIN MAX RIGHT)
                                                                    MIN))
                                                 (IDIFFERENCE (IMIN Y2 (ADD1 TOP))
                                                              Y1)
                                                 OPERATION]
                  ((EQP WIDTH 1)                                            ; special case of width 1
                    (\HQFX80.CLIP-AND-DRAW-LINE1 X1 Y1 X2 Y2 OPERATION BITMAP LEFT RIGHT BOTTOM TOP HQFX80STREAM))
                  ((IGREATERP (IABS (IDIFFERENCE X1 X2))
                              (IABS (IDIFFERENCE Y1 Y2)))                    ; slope is more horizontal, so make line grow in the positive y
                                                                            ; direction.
                    [COND
                       ((IGREATERP WIDTH 2)
                        (PROG (HALFWIDTH)
                              (SETQ HALFWIDTH (LRSH (SUB1 WIDTH)
                                                    1))
                              (SETQ Y1 (IDIFFERENCE Y1 HALFWIDTH))
                              (SETQ Y2 (IDIFFERENCE Y2 HALFWIDTH]
                    (for I from Y1 to (SUB1 (IPLUS Y1 WIDTH)) as J from Y2
                       do (\HQFX80.CLIP-AND-DRAW-LINE1 X1 I X2 J OPERATION BITMAP LEFT RIGHT BOTTOM TOP)))
                  (T                                                        ; slope is more vertical, so make line grow in the positive x
                                                                            ; direction.
                    [COND
                       ((IGREATERP WIDTH 2)
                        (PROG (HALFWIDTH)
                              (SETQ HALFWIDTH (LRSH (SUB1 WIDTH)
                                                    1))
                              (SETQ X1 (IDIFFERENCE X1 HALFWIDTH))
                              (SETQ X2 (IDIFFERENCE X2 HALFWIDTH]
                    (for I from X1 to (SUB1 (IPLUS X1 WIDTH)) as J from X2
                       do (\HQFX80.CLIP-AND-DRAW-LINE1 I Y1 J Y2 OPERATION BITMAP LEFT RIGHT BOTTOM TOP]))
```

(**HQFX80.CLIP-AND-DRAW-LINE1**

```
  [LAMBDA (X1 Y1 X2 Y2 OPERATION BITMAP LEFT RIGHT BOTTOM TOP)     (* hdj " 6-Nov-86 14:30")

    ;; LEFT, RIGHT, BOTTOM, TOP are set to the boundaries of the clipping region

    (PROG (DX DY YMOVEUP HALFDX HALFDY (BMRASTERWIDTH (fetch (BITMAP BITMAPRASTERWIDTH) of BITMAP)))
          (COND
            ((IGREATERP X1 X2)                                             ; switch points so DX is always positive.
             (SETQ HALFDX X1)
             (SETQ X1 X2)
             (SETQ X2 HALFDX)
             (SETQ HALFDX Y1)
             (SETQ Y1 Y2)
             (SETQ Y2 HALFDX)))                                           ; calculate differences and sign of Y movement.
          (SETQ HALFDX (LRSH (SETQ DX (IDIFFERENCE X2 X1))
                             1))
          (SETQ HALFDY (LRSH [SETQ DY (COND
                                        ((IGREATERP Y2 Y1)
                                         (SETQ YMOVEUP T)
                                         (IDIFFERENCE Y2 Y1))
                                        (T (IDIFFERENCE Y1 Y2]
```

```
                            1))
              (COND
                 ((AND (IGEQ X1 LEFT)
                       (IGEQ RIGHT X2)
                       [COND
                          (YMOVEUP (AND (IGEQ Y1 BOTTOM)
                                        (IGEQ TOP Y2)))
                          (T (AND (IGEQ Y2 BOTTOM)
                                  (IGEQ TOP Y1]
                       (EQ (fetch (BITMAP BITMAPBITSPERPIXEL) of BITMAP)
                           1))                                          ; line is completely visible, fast case.
                  (\DRAWLINE1 X1 (SUB1 (\SFInvert BITMAP Y1))
                       DX DY DX DY (COND
                                      ((IGREATERP DX DY)               ; X is the fastest mover.
                                       HALFDX)
                                      (T                               ; y is the fastest mover.
                                          HALFDY))
                       (COND
                          (YMOVEUP                                     ; y is moving in positive direction but bits are stored inversely
                             (IMINUS BMRASTERWIDTH))
                          (T BMRASTERWIDTH))
                       OPERATION
                       (fetch BITMAPBASE of BITMAP)
                       BMRASTERWIDTH))
                 (T (PROG ((CX1 X1)
                           (CY1 Y1)
                           (CX2 X2)
                           (CY2 Y2)
                           (CA1 (\CLIPCODE X1 Y1 LEFT RIGHT TOP BOTTOM))
                           (CA2 (\CLIPCODE X2 Y2 LEFT RIGHT TOP BOTTOM)))
                                                                       ; save the original points for the clipping computation.
                                                                       ; determine the sectors in which the points fall.
                      CLIPLP
                           [COND
                              ((NOT (EQ 0 (LOGAND CA1 CA2)))           ; line is entirely out of clipping region
                               (RETURN NIL))
                              ((EQ 0 (IPLUS CA1 CA2))                  ; line is completely visible
                              ;; \SFInvert has an off by one bug that everybody else in LLDISPLAY uses to save computation so SUB1 from what
                              ;; you would expect.                     ; reuse the variable CA1
                               (RETURN (\DRAWLINE1 CX1 (SUB1 (\SFInvert BITMAP CY1))
                                            (IDIFFERENCE CX2 CX1)
                                            (COND
                                               (YMOVEUP (IDIFFERENCE CY2 CY1))
                                               (T (IDIFFERENCE CY1 CY2)))
                                            DX DY (COND
                                                     ((IGREATERP DX DY)
                                                       ; X is the fastest mover.
                                                      (IREMAINDER (IPLUS (ITIMES DY (IDIFFERENCE CX1 X1))
                                                                         HALFDX)
                                                           DX))
                                                     (T                ; y is the fastest mover.
                                                        (IREMAINDER (IPLUS [ITIMES DX (COND
                                                                                         (YMOVEUP (IDIFFERENCE
                                                                                                     CY1 Y1))
                                                                                         (T (IDIFFERENCE Y1 CY1]
                                                                           HALFDY)
                                                              DY)))
                                            (COND
                                               (YMOVEUP                ; y is moving in positive direction but bits are stored inversely
                                                  (IMINUS BMRASTERWIDTH))
                                               (T BMRASTERWIDTH))
                                            OPERATION
                                            (fetch BITMAPBASE of BITMAP)
                                            BMRASTERWIDTH]
                           [COND
                              ((NEQ CA1 0)
                              ;; now move point CX1 CY1 so that one of the coordinates is on one of the boundaries.  Which boundary is done first
                              ;; was copied from BCPL.
                               (COND
                                  ((IGREATERP CA1 7)                   ; y1 less than bottom
                                                                       ; calculate the least X for which Y will be at bottom.
                                   [SETQ CX1 (IPLUS X1 (\LEASTPTAT DX DY (IDIFFERENCE BOTTOM Y1]
                                   (SETQ CY1 BOTTOM))
                                  ((IGREATERP CA1 3)                   ; y1 is greater than top
                                   [SETQ CX1 (IPLUS X1 (\LEASTPTAT DX DY (IDIFFERENCE Y1 TOP]
                                   (SETQ CY1 TOP))
                                  (T                                   ; x1 is less than left
                                     [SETQ CY1 (COND
                                                  [YMOVEUP (IPLUS Y1 (\LEASTPTAT DY DX (IDIFFERENCE LEFT X1]
                                                  (T (IDIFFERENCE Y1 (\LEASTPTAT DY DX (IDIFFERENCE LEFT X1]
                                     (SETQ CX1 LEFT)))
                               (SETQ CA1 (\CLIPCODE CX1 CY1 LEFT RIGHT TOP BOTTOM)))
                              (T                                       ; now move point CX2 CY2 so that one of the coordinates is on
                                                                       ; one of the boundaries
                                 (COND
                                    ((IGREATERP CA2 7)                 ; y2 less than bottom
```

```
                                          [SETQ CX2 (IPLUS X1 (\GREATESTPTAT DX DY (IDIFFERENCE Y1 BOTTOM]
                                          (SETQ CY2 BOTTOM))
                                       ((IGREATERP CA2 3)                            ; y2 is greater than top
                                        [SETQ CX2 (IPLUS X1 (\GREATESTPTAT DX DY (IDIFFERENCE TOP Y1]
                                        (SETQ CY2 TOP))
                                       (T                                            ; x2 is greater than right
                                          [SETQ CY2 (COND
                                                      [YMOVEUP (IPLUS Y1 (\GREATESTPTAT DY DX (IDIFFERENCE RIGHT X1]
                                                      (T (IDIFFERENCE Y1 (\GREATESTPTAT DY DX (IDIFFERENCE RIGHT X1]
                                          (SETQ CX2 RIGHT)))
                                      (SETQ CA2 (\CLIPCODE CX2 CY2 LEFT RIGHT TOP BOTTOM]
                                 (GO CLIPLP])
)

(DEFINEQ
```

(**HQFX80.DRAWCIRCLE**
```
  [LAMBDA (HQFX80STREAM CENTERX CENTERY RADIUS BRUSH DASHING)        (* hdj "21-Nov-86 17:11")
```

   ;; draw a circle on a hqfx80 stream
```
    (DECLARE (LOCALVARS . T))
    (COND
       ((OR (NOT (NUMBERP RADIUS))
            (ILESSP (SETQ RADIUS (FIXR RADIUS))
                    0))
        (\ILLEGAL.ARG RADIUS))
       ((EQ RADIUS 0)                                                ; don't draw anything.
        NIL)
       (T (GLOBALRESOURCE \HQFX80.BRUSHBBT
                   (WITH-HQFX80-DATA (HQFX80DATA HQFX80STREAM)
                        (PROG ((X 0)
                               (Y RADIUS)
                               (D (ITIMES 2 (- 1 RADIUS)))
                              LEFT RIGHTPLUS1 TOP BOTTOM DESTINATION-BITMAP BRUSHWIDTH BRUSHHEIGHT
                              LEFTMINUSBRUSH BOTTOMMINUSBRUSH TOPMINUSBRUSH BRUSHBM DESTINATIONBASE BRUSHBASE
                              RASTERWIDTH BRUSHRASTERWIDTH OPERATION HEIGHTMINUS1 CX CY (BBT \HQFX80.BRUSHBBT)
                              (USERFN (AND (LITATOM BRUSH)
                                           BRUSH)))
```

                             ;; many of these variables are used by the macro for \HQFX80.CURVEPT that passes them to \BBTCURVEPT and
                             ;; .SETUP.FOR.\HQFX80.BBTCURVEPT. sets them up.
```
                             [COND
                                (USERFN                             ; if calling user fn, don't bother with set up and leave points in
                                                                    ; stream coordinates.
                                      (SETQ CX CENTERX)
                                      (SETQ CY CENTERY))
                                (T (.SETUP.FOR.\HQFX80.BBTCURVEPT. HQFX80DATA)
                                   (SETQ CX (- CENTERX (FOLDLO BRUSHWIDTH 2)))
                                                                    ; take into account the brush thickness.
                                   (SETQ CY (- CENTERY (FOLDLO BRUSHHEIGHT 2]
                             [COND
                                ((EQ RADIUS 1)                      ; put a single brush down.
                                                                    ; draw the top and bottom most points.
                                 (COND
                                    (USERFN (APPLY* USERFN CX CY HQFX80STREAM))
                                    (T (\HQFX80.CURVEPT CX CY)))
                                 (RETURN))
                                (T                                  ; draw the top and bottom most points.
                                   (COND
                                      (USERFN (APPLY* USERFN CX (+ CY RADIUS)
                                                      HQFX80STREAM)
                                              (APPLY* USERFN CX (- CY RADIUS)
                                                      HQFX80STREAM))
                                      (T (\HQFX80.CURVEPT CX (+ CY RADIUS))
                                         (\HQFX80.CURVEPT CX (- CY RADIUS]
                             LP                                     ; (UNFOLD x 2) is used instead of (ITIMES x 2)
                             [COND
                                [(IGREATERP 0 D)
                                 (SETQ X (ADD1 X))
                                 (COND
                                    ((IGREATERP (UNFOLD (+ D Y)
                                                        2)
                                                1)
                                     (SETQ D (+ D (UNFOLD (- X Y)
                                                          2)
                                                 4))
                                     (SETQ Y (SUB1 Y)))
                                    (T (SETQ D (+ D (UNFOLD X 2)
                                                    1]
                                ((OR (EQ 0 D)
                                     (IGREATERP X D))
                                 (SETQ X (ADD1 X))
                                 (SETQ D (+ D (UNFOLD (- X Y)
                                                      2)
                                             4))
                                 (SETQ Y (SUB1 Y)))
```

```
                                    (T (SETQ D (+ (- D (UNFOLD Y 2))
                                                  3))
                                       (SETQ Y (SUB1 Y]
                             (COND
                                [(EQ Y 0)

                                  ;; left most and right most points are drawn specially so that they are not duplicated which leaves a hole in
                                  ;; XOR mode.

                                  (COND
                                     (USERFN (APPLY* USERFN (+ CX X)
                                                      CY HQFX80STREAM)
                                             (APPLY* USERFN (- CX X)
                                                      CY HQFX80STREAM))
                                     (T (\HQFX80.CURVEPT (+ CX X)
                                                 CY)
                                        (\HQFX80.CURVEPT (- CX X)
                                                 CY]
                                (T (COND
                                     (USERFN (APPLY* USERFN (+ CX X)
                                                      (+ CY Y)
                                                      HQFX80STREAM)
                                             (APPLY* USERFN (- CX X)
                                                      (+ CY Y)
                                                      HQFX80STREAM)
                                             (APPLY* USERFN (+ CX X)
                                                      (- CY Y)
                                                      HQFX80STREAM)
                                             (APPLY* USERFN (- CX X)
                                                      (- CY Y)
                                                      HQFX80STREAM))
                                     (T (\HQFX80.DRAW-4-CIRCLE-POINTS CX CY X Y)))
                             (GO LP)))
                       (MOVETO CENTERX CENTERY HQFX80STREAM)
                       (RETURN NIL)])
```

HQFX80.CREATE-BRUSH-BBT
```
  [LAMBDA (BRUSHBM HQFX80DATA BITBLT-TABLE)                    (* hdj "18-Nov-86 17:33")

    ;; Initializes BITBLT-TABLE for the BRUSHBM and an HQFX80 stream and returns BITBLT-TABLE, unless the BRUSHBM is a 1-point brush, in
    ;; which case it returns NIL.

    (COND
       ((AND (EQ (fetch (BITMAP BITMAPHEIGHT) of BRUSHBM)
                 1)
             (EQ (fetch (BITMAP BITMAPWIDTH) of BRUSHBM)
                 1)
             (EQ (BITMAPBIT BRUSHBM 0 0)
                 1))                                            ; special case of single point brush shape.
         NIL)
       (T                                                      ; update as many fields in the brush bitblt table as possible from
                                                               ; HQFX80DATA.
          (replace (PILOTBBT PBTDESTBPL) of BITBLT-TABLE with (UNFOLD (fetch (BITMAP BITMAPRASTERWIDTH)
                                                                         of (HQFX80DATA-BACKINGBITMAP HQFX80DATA))
                                                                 BITSPERWORD))
          (replace (PILOTBBT PBTSOURCEBPL) of BITBLT-TABLE with (UNFOLD (fetch (BITMAP BITMAPRASTERWIDTH)
                                                                         of BRUSHBM)
                                                                 BITSPERWORD))
          (replace (PILOTBBT PBTFLAGS) of BITBLT-TABLE with 0)
          (replace (PILOTBBT PBTDISJOINT) of BITBLT-TABLE with T)
          (\SETPBTFUNCTION BITBLT-TABLE (HQFX80DATA-SOURCETYPE HQFX80DATA)
                 (SELECTQ (HQFX80DATA-OPERATION HQFX80DATA)
                     ((PAINT REPLACE)
                         'PAINT)
                     ((INVERT ERASE)
                         'ERASE)
                     (SHOULDNT)))
          BITBLT-TABLE])
  )
```

```
(DEFMACRO \HQFX80.DRAW-4-CIRCLE-POINTS (CENTER-X CENTER-Y EDGE-X EDGE-Y)

    ;; draw four points 90 degress apart on the circumference of a circle

    `[PROGN (\HQFX80.CURVEPT (+ ,CENTER-X ,EDGE-X)
                    (+ ,CENTER-Y ,EDGE-Y))
            (\HQFX80.CURVEPT (- ,CENTER-X ,EDGE-X)
                    (+ ,CENTER-Y ,EDGE-Y))
            (\HQFX80.CURVEPT (+ ,CENTER-X ,EDGE-X)
                    (- ,CENTER-Y ,EDGE-Y))
            (\HQFX80.CURVEPT (- ,CENTER-X ,EDGE-X)
                    (- ,CENTER-Y ,EDGE-Y])

(DEFINEQ
```

HQFX80.FILLCIRCLE
```
  [LAMBDA (HQFX80STREAM CENTERX CENTERY RADIUS TEXTURE)        (* hdj " 6-Nov-86 15:45")
```

```
(COND
   ((OR (NOT (NUMBERP RADIUS))
        (ILESSP (SETQ RADIUS (FIXR RADIUS))
                0))
    (\ILLEGAL.ARG RADIUS))
   (T (WITH-HQFX80-DATA (HQFX80DATA HQFX80STREAM)
           (GLOBALRESOURCE \HQFX80.BRUSHBBT
                  (LET* [(CLIPPINGREGION (HQFX80DATA-CLIPPINGREGION HQFX80DATA))
                         (TOP (SUB1 (fetch (REGION TOP) of CLIPPINGREGION)))
                         (BOTTOM (fetch (REGION BOTTOM) of CLIPPINGREGION))
                         (LEFT (fetch (REGION LEFT) of CLIPPINGREGION))
                         (RIGHT (SUB1 (fetch (REGION RIGHT) of HQFX80DATA]
                      (PROG (TOP BOTTOM RIGHT LEFT OPERATION DESTINATION-BITMAP (X 0)
                             (Y RADIUS)
                             (D (ITIMES 2 (- 1 RADIUS))))
                             DESTINATIONBASE RASTERWIDTH CX CY TEXTUREBM GRAYHEIGHT GRAYWIDTH
                             GRAYBASE (FCBBT \HQFX80.BRUSHBBT))
                        (SETQ OPERATION (HQFX80DATA-OPERATION HQFX80DATA))
                        (SETQ DESTINATION-BITMAP (HQFX80DATA-BACKINGBITMAP HQFX80DATA))
                        [SETQ TEXTUREBM (COND
                                          ((BITMAPP TEXTURE))
                                          [(AND (NULL TEXTURE)
                                                (BITMAPP (HQFX80DATA-TEXTURE HQFX80DATA]
                                          ([OR (FIXP TEXTURE)
                                               (AND (NULL TEXTURE)
                                                    (SETQ TEXTURE (HQFX80DATA-TEXTURE HQFX80DATA]
                                             ; create bitmap for the texture.  Could reuse a bitmap but for now
                                             ; this is good enough.
                                           (SETQ TEXTUREBM (BITMAPCREATE 16 4))
                                           (SETQ GRAYBASE (fetch (BITMAP BITMAPBASE) of TEXTUREBM))
                                           (\PUTBASE GRAYBASE 0 (\SFReplicate (LOGAND (LRSH TEXTURE
                                                                                               12)
                                                                                       15)))
                                           (\PUTBASE GRAYBASE 1 (\SFReplicate (LOGAND (LRSH TEXTURE
                                                                                               8)
                                                                                       15)))
                                           (\PUTBASE GRAYBASE 2 (\SFReplicate (LOGAND (LRSH TEXTURE
                                                                                               4)
                                                                                       15)))
                                           (\PUTBASE GRAYBASE 3 (\SFReplicate (LOGAND TEXTURE 15)))
                                           TEXTUREBM)
                                          (T (\ILLEGAL.ARG TEXTURE]
                        (SETQ GRAYBASE (fetch (BITMAP BITMAPBASE) of TEXTUREBM))
                        (SETQ DESTINATIONBASE (fetch (BITMAP BITMAPBASE) of DESTINATION-BITMAP))
                        (SETQ RASTERWIDTH (fetch (BITMAP BITMAPRASTERWIDTH) of DESTINATION-BITMAP))
                   ;; update as many fields in the brush bitblt table as possible from the stream.
                        (replace PBTFLAGS of FCBBT with 0)
                        (replace PBTDESTBPL of FCBBT with (UNFOLD RASTERWIDTH BITSPERWORD))
                   ;; clear gray information.  PBTSOURCEBPL is used for gray information too.
                        (replace PBTSOURCEBPL of FCBBT with 0)
                        (replace PBTUSEGRAY of FCBBT with T)
                        [replace PBTGRAYWIDTHLESSONE of FCBBT
                            with (SUB1 (SETQ GRAYWIDTH (IMIN (fetch (BITMAP BITMAPWIDTH) of TEXTUREBM)
                                                             16]
                        [replace PBTGRAYHEIGHTLESSONE of FCBBT
                            with (SUB1 (SETQ GRAYHEIGHT (IMIN (fetch (BITMAP BITMAPHEIGHT) of TEXTUREBM
                                                                     )
                                                              16]
                        (replace PBTDISJOINT of FCBBT with T)
                        (\SETPBTFUNCTION FCBBT 'TEXTURE OPERATION)
                        (replace PBTHEIGHT of FCBBT with 1)
                                             ; take into account the brush thickness.
                        (SETQ CX (\DSPTRANSFORMX CENTERX HQFX80DATA))
                        (SETQ CY (\DSPTRANSFORMY CENTERY HQFX80DATA))
                                             ; change Y TOP and BOTTOM to be in bitmap coordinates
                        (SETQ CY (\SFInvert DESTINATION-BITMAP CY))
                        [SETQ BOTTOM (PROG1 (SUB1 (\SFInvert DESTINATION-BITMAP TOP))
                                           (SETQ TOP (SUB1 (\SFInvert DESTINATION-BITMAP BOTTOM))))]
                        (COND
                           ((EQ RADIUS 0)                 ; put a single point down.  Use \LINEBLT to get proper texture.
                                             ; NIL
                            (\LINEBLT FCBBT CX CY CX DESTINATIONBASE RASTERWIDTH LEFT RIGHT BOTTOM
                                   TOP GRAYWIDTH GRAYHEIGHT GRAYBASE 1)
                            (RETURN)))
                   LOOP
                                             ; (UNFOLD x 2) is used instead of (ITIMES x 2)
                        [COND
                           [(IGREATERP 0 D)
                            (SETQ X (ADD1 X))
                            (COND
                               ((IGREATERP (UNFOLD (+ D Y)
                                                   2)
                                           1)
                                (SETQ D (+ D (UNFOLD (- X Y)
                                                     2)
```

```
                                              4)))
                          (T (SETQ D (+ D (UNFOLD X 2)
                                          1))        ; don't draw unless Y changes.
                                (GO LOOP]
                ((OR (EQ 0 D)
                     (IGREATERP X D))
                 (SETQ X (ADD1 X))
                 (SETQ D (+ D (UNFOLD (- X Y)
                                        2)
                                4)))
                (T (SETQ D (+ (- D (UNFOLD Y 2))
                                3]
            (COND
                ((EQ Y 0)                          ; draw the middle line differently to avoid duplication.
                 (\LINEBLT FCBBT (- CX X)
                        CY
                        (+ CX X)
                        DESTINATIONBASE RASTERWIDTH LEFT RIGHT BOTTOM TOP GRAYWIDTH
                        GRAYHEIGHT GRAYBASE 1))
                (T (\HQFX80.FILL-CIRCLE-BLT CX CY X Y)
                   (SETQ Y (SUB1 Y))
                   (GO LOOP)))
            (MOVETO CENTERX CENTERY HQFX80STREAM)
            (RETURN NIL])
```

(**HQFX80.DRAWARC**

```
  [LAMBDA (HQFX80STREAM CENTERX CENTERY RADIUS STARTANGLE NDEGREES BRUSH DASHING)
                                                        (* hdj "20-Nov-86 14:27")
```

;; draws an arc on an hqfx80stream

```
    (\DRAWARC.GENERIC HQFX80STREAM CENTERX CENTERY RADIUS STARTANGLE NDEGREES BRUSH DASHING])
```

)

(DEFMACRO **HQFX80.FILL-CIRCLE-BLT** (CENTER-X CENTER-Y X Y)

;; calls bitblt twice to fill in one line of the circle.

```
    '(PROGN (\LINEBLT FCBBT (- ,CENTER-X ,X)
                    (+ ,CENTER-Y ,Y)
                    (+ ,CENTER-X ,X)
                    DESTINATIONBASE RASTERWIDTH LEFT RIGHT BOTTOM TOP GRAYWIDTH GRAYHEIGHT GRAYBASE 1)
            (\LINEBLT FCBBT (- ,CENTER-X ,X)
                    (- ,CENTER-Y ,Y)
                    (+ ,CENTER-X ,X)
                    DESTINATIONBASE RASTERWIDTH LEFT RIGHT BOTTOM TOP GRAYWIDTH GRAYHEIGHT GRAYBASE 1)))
```

;; curve-drawing

(DEFINEQ

(**HQFX80.DRAWCURVE**

```
  [LAMBDA (HQFX80STREAM KNOTS CLOSED BRUSH DASHING)                 (* hdj "19-Nov-86 14:42")
```

;; draws a spline curve with a given brush on HQFX80STREAM

```
    (GLOBALRESOURCE \HQFX80.BRUSHBBT (LET ([DASHLST (AND DASHING (OR (AND (LISTP DASHING)
                                                                        (EVERY DASHING (FUNCTION FIXP))
                                                                   DASHING)
                                                            (\ILLEGAL.ARG DASHING]
                                    (BBT \HQFX80.BRUSHBBT))
                                    (SELECTQ (LENGTH KNOTS)
                                        (0  ;; No knots => empty curve rather than error

                                            NIL)
                                        (1  ;; only one knot, put down a brush shape

                                            (OR (type? POSITION (CAR KNOTS))
                                                (ERROR "bad knot" (CAR KNOTS)))
                                            (DRAWPOINT (fetch XCOORD of (CAR KNOTS))
                                                    (fetch YCOORD of (CAR KNOTS))
                                                BRUSH HQFX80STREAM))
                                        (2 (OR (type? POSITION (CAR KNOTS))
                                               (ERROR "bad knot" (CAR KNOTS)))
                                           (OR (type? POSITION (CADR KNOTS))
                                               (ERROR "bad knot" (CADR KNOTS)))
                                           (\HQFX80.LINEWITHBRUSH (fetch XCOORD of (CAR KNOTS))
                                                    (fetch YCOORD of (CAR KNOTS))
                                                    (fetch XCOORD of (CADR KNOTS))
                                                    (fetch YCOORD of (CADR KNOTS))
                                                    BRUSH DASHLST HQFX80STREAM BBT))
                                        (\HQFX80.DRAWCURVE2 (PARAMETRICSPLINE KNOTS CLOSED)
                                                BRUSH DASHLST BBT HQFX80STREAM))
                                    HQFX80STREAM])
```

(**HQFX80.DRAWCURVE2**

```
  [LAMBDA (SPLINE BRUSH DASHLST BBT HQFX80STREAM)              (* hdj "19-Nov-86 11:58")
```

;;; Given a spline curve, represented as a set of derivatives for each segment, draw it on HQFX80STREAM using the brush BRUSH, and dashing it
;;; according to DASHLST. For speed, use the bitblt table BBT.

```
      (DECLARE (SPECVARS . T))
```

;; Should declare most of these variables local but currently have the \CURVE function between here and \CURVEBBT so can't

```
      (WITH-HQFX80-DATA (HQFX80DATA HQFX80STREAM)
              (PROG (BRUSHBM DESTINATION-BITMAP OPERATION BRUSHWIDTH BRUSHHEIGHT BRUSHBASE BRUSHRASTERWIDTH LEFT
                      RIGHTPLUS1 TOP BOTTOM DESTINATIONBASE LEFTMINUSBRUSH BOTTOMMINUSBRUSH TOPMINUSBRUSH
                      RASTERWIDTH NBITSRIGHTPLUS1 HEIGHTMINUS1 \CURX \CURY \OLDX \OLDY \OLDERX \OLDERY LKNOT
                      (DASHON T)
                      (DASHTAIL DASHLST)
                      (DASHCNT (CAR DASHLST))
                      NPOINTS NSEGS POINTSPERSEG DX D2X D3X DY D2Y D3Y D1 D2 D3 X0 Y0 X1 Y1 DX DDX DDDX DY DDY
                      DDDY (XPOLY (create POLYNOMIAL))
                      (X/PRIME/POLY (create POLYNOMIAL))
                      (YPOLY (create POLYNOMIAL))
                      (Y/PRIME/POLY (create POLYNOMIAL))
                      (USERFN (AND (LITATOM BRUSH)
                                   BRUSH)))
```

;; many of these variables are used by the macro for \CURVEPT that passes them to \BBTCURVEPT and
;; .SETUP.FOR.\HQFX80.BBTCURVEPT. sets them up.

```
                    [COND
                      (USERFN                                       ; if calling user fn, don't bother with set up and leave points in
                                                                    ; window coordinates.
                          (\CURVESTART (ELT (fetch (SPLINE SPLINEX) of SPLINE)
                                             1)
                                       (ELT (fetch (SPLINE SPLINEY) of SPLINE)
                                             1)))
                      (T (.SETUP.FOR.\HQFX80.BBTCURVEPT. HQFX80DATA)
                                                                    ; curve pts will be kept in screen coordinates, start smoothing
                                                                    ; values there.
                          (\CURVESTART (IDIFFERENCE (ELT (fetch (SPLINE SPLINEX) of SPLINE)
                                                          1)
                                                    (LRSH (SUB1 BRUSHWIDTH)
                                                          1))
                                       (IDIFFERENCE (ELT (fetch (SPLINE SPLINEY) of SPLINE)
                                                          1)
                                                    (LRSH (SUB1 BRUSHHEIGHT)
                                                          1]
                    [bind PERSEG for KNOT from 1 to (SUB1 (fetch %#KNOTS of SPLINE))
                        when (PROGN
```

;;; Loop thru the segments of the spline curve, drawing each in turn.

```
                                (SETQ X0 (ELT (fetch (SPLINE SPLINEX) of SPLINE)
                                              KNOT))              ; Set up X0,Y0 -- the starting point of this segment
                                (SETQ Y0 (ELT (fetch (SPLINE SPLINEY) of SPLINE)
                                              KNOT))
                                (SETQ X1 (ELT (fetch (SPLINE SPLINEX) of SPLINE)
                                              (ADD1 KNOT)))       ; And X1,Y1 -- the ending point
                                (SETQ Y1 (ELT (fetch (SPLINE SPLINEY) of SPLINE)
                                              (ADD1 KNOT)))
                                (SETQ DX (ELT (fetch (SPLINE SPLINEDX) of SPLINE)
                                              KNOT))              ; And the initial derivatives -- first
                                (SETQ DY (ELT (fetch (SPLINE SPLINEDY) of SPLINE)
                                              KNOT))
                                (SETQ DDX (ELT (fetch SPLINEDDX of SPLINE)
                                              KNOT))              ; Second
                                (SETQ DDY (ELT (fetch SPLINEDDY of SPLINE)
                                              KNOT))
                                (SETQ DDDX (ELT (fetch SPLINEDDDX of SPLINE)
                                              KNOT))              ; And third.
                                (SETQ DDDY (ELT (fetch SPLINEDDDY of SPLINE)
                                              KNOT))
                                (SETQ NPOINTS (FOLDLO (ITIMES (IMAX (IABS (IDIFFERENCE X1 X0))
                                                                    (IABS (IDIFFERENCE Y1 Y0)))
                                                              3)
                                                      2))
```

;; Establish an upper bound on the number of points we'll draw while painting this segment. We know that
;; 3/2 the maximum DX or DY is the right amount.

```
                                (NOT (ZEROP NPOINTS)))
                        do
```
;; NPOINTS can be zero if a knot is duplicated in the spline curve to produce a discontinuity. Skip over zero-length
;; segments to avoid divide-by-zero trouble

;; To prevent round-off errors from accumulating, we'll draw this segment as runs of no more than 64 points each --
;; recomputing completely at the start of each run. This is a trade off of speed and accuracy.

```
                          [COND
                            ((ILEQ NPOINTS 64)                    ; Fewer than 64 points to draw.  Do it in one run.
                             (SETQ NSEGS 1)
                             (SETQ POINTSPERSEG NPOINTS))
                            (T                                    ; Figure out how many runs to do it in.
                                (SETQ NSEGS (FOLDLO NPOINTS 64))
```

```
                                   (SETQ POINTSPERSEG 64)
                                   (SETQ NPOINTS (UNFOLD NSEGS 64]
                           (SETQ D1 (FQUOTIENT 1.0 NPOINTS))            ; Set up Δt, Δt**2 and Δt**3, for computing the next point.
                           (SETQ D2 (FTIMES D1 D1))
                           (SETQ D3 (FTIMES D2 D1))
                           (SETQ D3X (FTIMES D3 DDDX))
                           (SETQ D3Y (FTIMES D3 DDDY))
                           (COND
                              [(EQ NSEGS 1)                             ; Just one segment to draw.
                               [SETQ DX (FPLUS (FTIMES D1 DX)
                                               (FTIMES DDX D2 0.5)
                                               (FTIMES DDDX D3 (CONSTANT (FQUOTIENT 1.0 6.0]
                               (SETQ D2X (FPLUS (FTIMES D2 DDX)
                                                (FTIMES D3 DDDX)))
                               [SETQ DY (FPLUS (FTIMES D1 DY)
                                               (FTIMES D2 DDY 0.5)
                                               (FTIMES D3 DDDY (CONSTANT (FQUOTIENT 1.0 6.0]
                               (SETQ D2Y (FPLUS (FTIMES D2 DDY)
                                                (FTIMES D3 DDDY)))
                               (COND
                                  (USERFN                              ; Draw this run of points, using the user's supplied function.
                                      (\HQFX80.DRAWCURVE3 X0 Y0 X1 Y1 DX DY D2X D2Y D3X D3Y NPOINTS BRUSHBM
                                             HQFX80DATA BBT NIL USERFN HQFX80STREAM))
                                  (T                                   ; Draw this run of points, using the brush.
                                     (\HQFX80.DRAWCURVE3 X0 Y0 X1 Y1 DX DY D2X D2Y D3X D3Y NPOINTS BRUSHBM
                                             HQFX80DATA BBT NIL NIL HQFX80STREAM]
                              (T                                       ; Have to do this segment in several runs.
                                 (SETQ PERSEG (FQUOTIENT 1.0 NSEGS))
                                 (LOADPOLY XPOLY X/PRIME/POLY DDDX DDX DX X0)
                                 (LOADPOLY YPOLY Y/PRIME/POLY DDDY DDY DY Y0)
                                 (bind (TT _ 0.0)
                                       (DDDX/PER/SEG _ (FTIMES DDDX PERSEG))
                                       (DDDY/PER/SEG _ (FTIMES DDDY PERSEG))
                                       [D3XFACTOR _ (FTIMES D3 DDDX (CONSTANT (FQUOTIENT 1.0 6.0]
                                       [D3YFACTOR _ (FTIMES D3 DDDY (CONSTANT (FQUOTIENT 1.0 6.0] for I from 0
                                    to (SUB1 NSEGS)
                                    do
```

;;; TT is the parameter, and runs from 0 to 1 as the curve segment runs from beginning to end.

```
                                       (SETQ TT (FPLUS TT PERSEG))
                                       (SETQ X1 (POLYEVAL TT XPOLY 3))
                                       (SETQ Y1 (POLYEVAL TT YPOLY 3))
                                       (SETQ DX (FPLUS (FTIMES D1 DX)
                                                       (FTIMES D2 DDX 0.5)
                                                       D3XFACTOR))
                                       (SETQ D2X (FPLUS (FTIMES D2 DDX)
                                                        (FTIMES D3 DDDX)))
                                       (SETQ DY (FPLUS (FTIMES D1 DY)
                                                       (FTIMES D2 DDY 0.5)
                                                       D3YFACTOR))
                                       (SETQ D2Y (FPLUS (FTIMES D2 DDY)
                                                        (FTIMES D3 DDDY)))
                                       (COND
                                          (USERFN (\HQFX80.DRAWCURVE3 X0 Y0 X1 Y1 DX DY D2X D2Y D3X D3Y 64 BRUSHBM
                                                       HQFX80DATA BBT NIL USERFN HQFX80STREAM))
                                          (T (\HQFX80.DRAWCURVE3 X0 Y0 X1 Y1 DX DY D2X D2Y D3X D3Y 64 BRUSHBM
                                                  HQFX80DATA BBT NIL NIL HQFX80STREAM)))
                                       (SETQ X0 X1)
                                       (SETQ Y0 Y1)
                                       (SETQ DDX (FPLUS DDX DDDX/PER/SEG))
                                       (SETQ DDY (FPLUS DDY DDDY/PER/SEG))
                                       (SETQ DX (POLYEVAL TT X/PRIME/POLY 2))
                                       (SETQ DY (POLYEVAL TT Y/PRIME/POLY 2]
```

;;; Draw the final point on the curve.

```
                           (COND
                              (USERFN (\HQFX80.DRAWCURVE3 0 0 0 0 0 0 0 0 0 0 0 BRUSHBM HQFX80DATA BBT T USERFN
                                           HQFX80STREAM))
                              (T (\HQFX80.DRAWCURVE3 0 0 0 0 0 0 0 0 0 0 0 BRUSHBM HQFX80DATA BBT T NIL HQFX80STREAM])
```

(**HQFX80.DRAWCURVE3**

```
  [LAMBDA (X0 Y0 X1 Y1 DX DY DDX DDY DDDX DDDY N BRUSHBM HQFX80DATA BBT ENDING USERFN HQFX80STREAM)
                                                                (* hdj "19-Nov-86 12:18")
     (DECLARE (LOCALVARS . T)
          (USEDFREE BRUSHWIDTH BRUSHHEIGHT \CURX \OLDX \CURY \OLDY))
     ;; Puts a spline segment down.  Since it calls BitBlt directly, it must clip to both clipping region and the size of the destination bit map.
     (PROG (OLDX X Y OLDY DELTAX DELTAY DELTA TX TY OOLDX OOLDY)
          [COND
              ((NEQ N 0)
               [COND
                   (USERFN                                      ; if there is a user fn, stay in his coordinates.
                        (SETQ OLDX X0)
                        (SETQ OLDY Y0))
```

```
                           (T  ;; SUB1 on brush size is to cause the extra bit to be in the top left direction as is documented for lines.
                                  (SETQ OLDX (IDIFFERENCE X0 (LRSH (SUB1 BRUSHWIDTH)
                                                                 1)))
                                  (SETQ OLDY (IDIFFERENCE Y0 (LRSH (SUB1 BRUSHHEIGHT)
                                                                 1]                     ; draw origin point
                       (\HQFX80.SMOOTH-CURVE OLDX OLDY USERFN HQFX80STREAM)
                                                                       ; convert the derivatives to fractional representation.
                                                                       ; \CONVERTTOFRACTION always returns a large number box.
                                                                       ; This uses .49 because .5 causes rounding up.
                       (SETQ X (\CONVERTTOFRACTION (FPLUS OLDX 0.49)))
                       (SETQ Y (\CONVERTTOFRACTION (FPLUS OLDY 0.49)))
                       (SETQ DX (\CONVERTTOFRACTION DX))
                       (SETQ DY (\CONVERTTOFRACTION DY))
                       (SETQ DDX (\CONVERTTOFRACTION DDX))
                       (SETQ DDY (\CONVERTTOFRACTION DDY))
                       (SETQ DDDX (\CONVERTTOFRACTION DDDX))
                       (SETQ DDDY (\CONVERTTOFRACTION DDDY))
                       [for I from 1 to N do                                 ; uses \BOXIPLUS to save box and also set the new value of the
                                                                             ; variable.
                                       (\BOXIPLUS X DX)
                                       (\BOXIPLUS DX DDX)
                                       (\BOXIPLUS DDX DDDX)
                                       (\BOXIPLUS Y DY)
                                       (\BOXIPLUS DY DDY)
                                       (\BOXIPLUS DDY DDDY)
                                       (SETQ OOLDX OLDX)
                                       (SETQ OOLDY OLDY)
                                       (SETQ DELTAX (IDIFFERENCE (SETQ OLDX (\GETINTEGERPART X))
                                                                 OOLDX))
                                       (SETQ DELTAY (IDIFFERENCE (SETQ OLDY (\GETINTEGERPART Y))
                                                                 OOLDY))
                                       (SETQ DELTA (IMAX (IABS DELTAX)
                                                         (IABS DELTAY)))
                                       (COND
                                          ((EQ DELTA 1)
                                           (\HQFX80.SMOOTH-CURVE OLDX OLDY USERFN HQFX80STREAM)))
                                       (COND
                                          ((IGREATERP DELTA 1)
                                           (SETQ DELTAX (\CONVERTTOFRACTION (FQUOTIENT DELTAX DELTA)))
                                           (SETQ DELTAY (\CONVERTTOFRACTION (FQUOTIENT DELTAY DELTA)))
                                           (SETQ TX (\CONVERTTOFRACTION OOLDX))
                                           (SETQ TY (\CONVERTTOFRACTION OOLDY))
                                           (for I from 0 to DELTA do (\HQFX80.SMOOTH-CURVE (\GETINTEGERPART TX)
                                                                                          (\GETINTEGERPART TY)
                                                                               USERFN HQFX80STREAM)
                                                                (\BOXIPLUS TX DELTAX)
                                                                (\BOXIPLUS TY DELTAY]
                                                                     ; draw the end point
                       (COND
                          (USERFN (\HQFX80.SMOOTH-CURVE X1 Y1 USERFN HQFX80STREAM))
                          (T (\HQFX80.SMOOTH-CURVE (IDIFFERENCE X1 (LRSH (SUB1 BRUSHWIDTH)
                                                                         1))
                                     (IDIFFERENCE Y1 (LRSH (SUB1 BRUSHHEIGHT)
                                                           1))
                                 NIL HQFX80STREAM)))
                       (AND HQFX80STREAM (MOVETO (FIX X1)
                                                (FIX Y1)
                                                HQFX80STREAM]
                   (COND
                      (ENDING (\HQFX80.SMOOTH-CURVE (IPLUS \CURX \CURX (IMINUS \OLDX))
                                    (IPLUS \CURY \CURY (IMINUS \OLDY))
                                    USERFN HQFX80STREAM)
                              (\HQFX80.SMOOTH-CURVE (IPLUS \CURX \CURX (IMINUS \OLDX))
                                    (IPLUS \CURY \CURY (IMINUS \OLDY))
                                    USERFN HQFX80STREAM)))
                   (RETURN NIL])
```

(**HQFX80.LINEWITHBRUSH**
```
   [LAMBDA (X1 Y1 X2 Y2 BRUSH DASHLST HQFX80STREAM BBT)               ; Edited  5-Jan-87 16:57 by hdj

     ;; draws a line with a brush on a HQFX80STREAM

     (DECLARE (LOCALVARS . T))
     (WITH-HQFX80-DATA (HQFX80DATA HQFX80STREAM)
          (PROG (DESTINATION-BITMAP LEFT RIGHTPLUS1 TOP BOTTOM BRUSHWIDTH BRUSHHEIGHT LEFTMINUSBRUSH
                       BOTTOMMINUSBRUSH TOPMINUSBRUSH BRUSHBM DESTINATIONBASE BRUSHBASE RASTERWIDTH
                       BRUSHRASTERWIDTH OPERATION HEIGHTMINUS1 HALFBRUSHWIDTH HALFBRUSHHEIGHT DX DY YINC CDL
                       (DASHON T)
                       (DASHTAIL DASHLST)
                       (DASHCNT (CAR DASHLST))
                       (USERFN (AND (LITATOM BRUSH)
                                    BRUSH)))

            ;; many of these variables are used by the macro for \HQFX80.CURVEPT that passes them to \HQFX80.BBTCURVEPT and
            ;; .SETUP.FOR.\\HQFX80.BBTCURVEPT.  sets them up.

               [COND
```

```
                        ((NOT USERFN)
                          (.SETUP.FOR.\HQFX80.BBTCURVEPT. HQFX80DATA) ; SUB1 is so that the extra bit goes on the top and right as it is
                                                                      ; documented as doing for lines.
                          [SETQ X1 (- X1 (SETQ HALFBRUSHWIDTH (FOLDLO (SUB1 BRUSHWIDTH)
                                                                       2]
                          (SETQ X2 (- X2 HALFBRUSHWIDTH))
                          [SETQ Y1 (- Y1 (SETQ HALFBRUSHHEIGHT (FOLDLO (SUB1 BRUSHHEIGHT)
                                                                        2]
                                                                      ; take into account the brush thickness.
                          (SETQ Y2 (- Y2 HALFBRUSHHEIGHT]              ; arrange things so that dx is positive.
                    (COND
                      ((> X1 X2)                                       ; switch points
                        (swap X1 X2)
                        (swap Y1 Y2)))
                    (SETQ DX (ADD1 (- X2 X1)))
                    [SETQ DY (ADD1 (COND
                                     ((> Y2 Y1)
                                      (SETQ YINC 1)
                                      (- Y2 Y1))
                                     (T (SETQ YINC -1)
                                        (- Y1 Y2]
                    [SETQ CDL (HALF (COND
                                     ((> DX DY)                        ; set up the bucket so that the ends will be the same.
                                      (IREMAINDER DX DY))
                                     (T (IREMAINDER DY DX]
                    [COND
                      [USERFN                                          ; if user function is being called, don't bother bringing window to
                                                                       ; top uninterruptably.
                           (COND
                             ((IGEQ DX DY)                             ; X is the fastest mover.
                              (until (> X1 X2) do                      ; main loop
                                      (COND
                                        (DASHON (APPLY* USERFN X1 Y1 HQFX80STREAM)))
                                      [COND
                                        (DASHTAIL    ; do dashing.
                                               (COND
                                                 ((EQ 0 (SETQ DASHCNT (SUB1 DASHCNT)))
                                                  (SETQ DASHON (NOT DASHON))
                                                  (SETQ DASHTAIL (OR (LISTP (CDR DASHTAIL))
                                                                     DASHLST))
                                                  (SETQ DASHCNT (CAR DASHTAIL]
                                      [COND
                                        ((NOT (> DX (add CDL DY)))
                                         (add Y1 YINC)
                                         (COND
                                           ((COND
                                              ((EQ YINC -1)
                                               (ILESSP Y1 Y2))
                                              ((> Y1 Y2)))
                                            (RETURN)))
                                         (SETQ CDL (- CDL DX]
                                      (add X1 1)))
                             (T                                        ; Y is the fastest mover.
                                (until (COND
                                         ((EQ YINC -1)
                                          (ILESSP Y1 Y2))
                                         ((> Y1 Y2)))
                                    do                                 ; main loop
                                      (COND
                                        (DASHON (APPLY* USERFN X1 Y1 HQFX80STREAM)))
                                      [COND
                                        (DASHTAIL                      ; do dashing.
                                               (COND
                                                 ((EQ 0 (SETQ DASHCNT (SUB1 DASHCNT)))
                                                  (SETQ DASHON (NOT DASHON))
                                                  (SETQ DASHTAIL (OR (LISTP (CDR DASHTAIL))
                                                                     DASHLST))
                                                  (SETQ DASHCNT (CAR DASHTAIL]
                                      [COND
                                        ([NOT (> DY (SETQ CDL (+ CDL DX]
                                         (COND
                                           ((> (SETQ X1 (ADD1 X1))
                                               X2)
                                            (RETURN)))
                                         (SETQ CDL (- CDL DY]
                                      (add Y1 YINC]
                      (T (COND
                           [(IGEQ DX DY)                               ; X is the fastest mover.
                            (until (> X1 X2) do                        ; main loop
                                    (COND
                                      (DASHON (\HQFX80.CURVEPT X1 Y1)))
                                    [COND
                                      (DASHTAIL      ; do dashing.
                                             (COND
                                               ((EQ 0 (SETQ DASHCNT (SUB1 DASHCNT)))
                                                (SETQ DASHON (NOT DASHON))
                                                (SETQ DASHTAIL (OR (LISTP (CDR DASHTAIL))
```

```
                                                                        DASHLST))
                                                        (SETQ DASHCNT (CAR DASHTAIL]
                                        [COND
                                           ([NOT (> DX (SETQ CDL (+ CDL DY]
                                            (SETQ Y1 (+ Y1 YINC))
                                            (COND
                                                ((COND
                                                    ((EQ YINC -1)
                                                     (ILESSP Y1 Y2))
                                                    ((> Y1 Y2)))
                                                 (RETURN)))
                                            (SETQ CDL (- CDL DX]
                                        (SETQ X1 (ADD1 X1]
                (T                                              ; Y is the fastest mover.
                        (until (COND
                                   ((EQ YINC -1)
                                    (ILESSP Y1 Y2))
                                   ((> Y1 Y2)))
                            do                                  ; main loop
                                (COND
                                   (DASHON (\HQFX80.CURVEPT X1 Y1)))
                                [COND
                                   (DASHTAIL                    ; do dashing.
                                            (COND
                                               ((EQ 0 (SETQ DASHCNT (SUB1 DASHCNT)))
                                                (SETQ DASHON (NOT DASHON))
                                                (SETQ DASHTAIL (OR (LISTP (CDR DASHTAIL))
                                                                   DASHLST))
                                                (SETQ DASHCNT (CAR DASHTAIL]
                                [COND
                                   ([NOT (> DY (SETQ CDL (+ CDL DX]
                                    (COND
                                       ((> (SETQ X1 (ADD1 X1))
                                           X2)
                                        (RETURN)))
                                    (SETQ CDL (- CDL DY]
                                (SETQ Y1 (+ Y1 YINC]
                (RETURN NIL])
)

(DEFINEQ
```

(**HQFX80.BBTCURVEPT**

```
  [LAMBDA (X Y BBT LEFT BRUSHWIDTH LEFTMINUSBRUSH RIGHTPLUS1 TOPMINUSBRUSH DESTINATION-BITMAP BRUSHHEIGHT
           BOTTOMMINUSBRUSH TOP BRUSHBASE DESTINATIONBASE RASTERWIDTH BRUSHRASTERWIDTH HQFX80DATA)
                                            (* hdj " 6-Nov-86 14:36")

    ;; Called by \hqfx80.CURVEPT macro.  Draws a brush point by bitblting BRUSHBM to point X,Y in DestinationBitMap.  BBT is a BitBlt table where
    ;; everything is already set except the source and destination addresses, width and height.  In other words, only the easy stuff
                                                                 :
                                                                 ; set the width fields of the bbt
        [PROG (CLIPPEDTOP STY)
            [COND
                [(ILEQ Y TOPMINUSBRUSH)                          ; the top part of the brush is visible
                 (SETQ CLIPPEDTOP (IPLUS Y BRUSHHEIGHT))
                 (replace PBTSOURCE of BBT with BRUSHBASE)
                 (replace PBTHEIGHT of BBT with (IMIN BRUSHHEIGHT (IDIFFERENCE Y BOTTOMMINUSBRUSH]
                (T                                               ; only the bottom is visible
                 (SETQ CLIPPEDTOP TOP)
                 [replace PBTSOURCE of BBT with (\ADDBASE BRUSHBASE (ITIMES BRUSHRASTERWIDTH (SETQ STY
                                                                                               (IDIFFERENCE Y
                                                                                                 TOPMINUSBRUSH]
                 (replace PBTHEIGHT of BBT with (IDIFFERENCE (IMIN BRUSHHEIGHT (IDIFFERENCE Y BOTTOMMINUSBRUSH))
                                                   STY]
            (replace PBTDEST of BBT with (\ADDBASE DESTINATIONBASE (ITIMES RASTERWIDTH (\SFInvert DESTINATION-BITMAP
                                                                                          CLIPPEDTOP]
        [COND
            [(ILESSP X LEFT)                                     ; only the right part of the brush is visible
             (replace PBTDESTBIT of BBT with LEFT)
             (replace PBTSOURCEBIT of BBT with (IDIFFERENCE BRUSHWIDTH (replace PBTWIDTH of BBT with (IDIFFERENCE X
                                                                                                     LEFTMINUSBRUSH
                                                                                                     ]
            (T                                                   ; left edge is visible
             (replace PBTDESTBIT of BBT with X)
             (replace PBTSOURCEBIT of BBT with 0)                ; set width to the amount that is visible
             (replace PBTWIDTH of BBT with (IMIN BRUSHWIDTH (IDIFFERENCE RIGHTPLUS1 X]
        (\PILOTBITBLT BBT 0])
)

(DECLARE%: EVAL@COMPILE

(PUTPROPS \HQFX80.CURVEPT MACRO [OPENLAMBDA (X Y)

                                 ;; puts a brush shape at point X,Y.  Assumes X and Y have been corrected so that it is the lower left
                                 ;; corner of the brush.  Does a clipping to the region defined by LEFT RIGHTPLUS1 BOTTOM and
                                 ;; LEFTMINUSBRUSH TOPMINUSBRUSH BOTTOMMINUSBRUSH.
```

```
                                          (COND
                                             ((OR (ILEQ X LEFTMINUSBRUSH)
                                                  (IGEQ X RIGHTPLUS1)
                                                  (ILEQ Y BOTTOMMINUSBRUSH)
                                                  (IGEQ Y TOP))            ; Brush is entirely out of region
                                               NIL)
                                             ((NULL BBT)                   ; Special case of single point brush
                                              (\FBITMAPBIT DESTINATIONBASE X Y OPERATION HEIGHTMINUS1 RASTERWIDTH))
                                             (T                            ; Some part of the brush in the region
                                                (\HQFX80.BBTCURVEPT X Y BBT LEFT BRUSHWIDTH LEFTMINUSBRUSH RIGHTPLUS1
                                                        TOPMINUSBRUSH DESTINATION-BITMAP BRUSHHEIGHT BOTTOMMINUSBRUSH
                                                        TOP BRUSHBASE DESTINATIONBASE RASTERWIDTH BRUSHRASTERWIDTH
                                                        HQFX80DATA]))
)


(DEFMACRO \HQFX80.SMOOTH-CURVE (NEWX NEWY USERFN HQFX80STREAM)
    '(LET [(DX (IABS (- ,NEWX \OLDX)))
           (DY (IABS (- ,NEWY \OLDY]
          (COND
             ((OR (> DX 1)
                  (> DY 1))
                [COND
                   ((NEQ [+ (ADD1 (- \OLDX \OLDERX))
                            (ITIMES 3 (ADD1 (- \OLDY \OLDERY]
                         4)
                    [COND
                       (DASHON (COND
                                  (,USERFN (APPLY* ,USERFN \OLDX \OLDY ,HQFX80STREAM))
                                  (T (\HQFX80.CURVEPT \OLDX \OLDY]
                    (COND
                       (DASHTAIL (COND
                                    ((EQ 0 (SETQ DASHCNT (SUB1 DASHCNT)))
                                     (SETQ DASHON (NOT DASHON))
                                     (SETQ DASHTAIL (OR (LISTP (CDR DASHTAIL))
                                                        DASHLST))
                                     (SETQ DASHCNT (CAR DASHTAIL]
                   (SETQ \OLDERX \OLDX)
                   (SETQ \OLDERY \OLDY)
                   (SETQ \OLDX \CURX)
                   (SETQ \OLDY \CURY)))
             (SETQ \CURX ,NEWX)
             (SETQ \CURY ,NEWY)))


(DEFMACRO .SETUP.FOR.\HQFX80.BBTCURVEPT. (HQFX80DATA)
    '(LET [(CLIPPINGREGION (HQFX80DATA-CLIPPINGREGION ,HQFX80DATA]
          (SETQ BOTTOM (ffetch (REGION BOTTOM) of CLIPPINGREGION))
          (SETQ TOP (ffetch (REGION TOP) of CLIPPINGREGION))
          (SETQ RIGHTPLUS1 (ffetch (REGION RIGHT) of CLIPPINGREGION))
          (SETQ LEFT (ffetch (REGION LEFT) of CLIPPINGREGION))
          (SETQ DESTINATION-BITMAP (HQFX80DATA-BACKINGBITMAP ,HQFX80DATA))
          (SETQ OPERATION (HQFX80DATA-OPERATION ,HQFX80DATA))
          (SETQ BRUSHBM (\GETBRUSH BRUSH))
          (SETQ RASTERWIDTH (fetch (BITMAP BITMAPRASTERWIDTH) of DESTINATION-BITMAP))
          (SETQ DESTINATIONBASE (fetch (BITMAP BITMAPBASE) of DESTINATION-BITMAP))
          (SETQ BBT (\HQFX80.CREATE-BRUSH-BBT BRUSHBM ,HQFX80DATA BBT))
          (SETQ BRUSHBASE (fetch (BITMAP BITMAPBASE) of BRUSHBM))
```

          ;; keep Brush width and raster width in number of bits units.

```
          (SETQ BRUSHRASTERWIDTH (fetch (BITMAP BITMAPRASTERWIDTH) of BRUSHBM))
          [COND
             ((NULL BBT)
```

              ;; BBT is NIL if single point brush. Set the destination bitmap base.

```
              (SETQ HEIGHTMINUS1 (SUB1 (fetch (BITMAP BITMAPHEIGHT) of DESTINATION-BITMAP)))
              (COND
                 ((EQ (HQFX80DATA-OPERATION ,HQFX80DATA)
                      'INVERT)
```

                  ;; really do invert in single brush case.

```
                  (SETQ OPERATION 'INVERT]
          (SETQ BRUSHWIDTH (fetch (BITMAP BITMAPWIDTH) of BRUSHBM))
          (SETQ BRUSHHEIGHT (fetch (BITMAP BITMAPHEIGHT) of BRUSHBM))
          (SETQ LEFTMINUSBRUSH (IDIFFERENCE LEFT BRUSHWIDTH))
          (SETQ BOTTOMMINUSBRUSH (IDIFFERENCE BOTTOM BRUSHHEIGHT))
          (SETQ TOPMINUSBRUSH (IDIFFERENCE TOP BRUSHHEIGHT))))
```

;; character printing methods

```
(DEFINEQ
```

(**HQFX80.OUTCHAR**
```
  [LAMBDA (HQFX80STREAM CHARCODE)                                      ; Edited  4-Feb-87 15:11 by hdj
```
    ;; Displays the character and increments the Xposition on the HQFX80STREAM.

```
    (WITH-HQFX80-DATA (HQFX80DATA HQFX80STREAM)
```

```
                ;; If necessary, change the charset
             (if (NEQ (HQFX80DATA-CHARSET-CACHE HQFX80DATA)
                      (\CHARSET CHARCODE))
                 then (\HQFX80.CHANGE-CHARSET HQFX80DATA (\CHARSET CHARCODE)))
            (SELCHARQ CHARCODE
                  (^L                                                            ; form-feed
                       (\HQFX80.NEWPAGE HQFX80STREAM))
                  ((EOL CR LF)                                                   ; various line-enders
                       (\HQFX80.NEWLINE CHARCODE HQFX80STREAM))
                  (LET ((CHARWIDTH (\HQFX80.CHARWIDTH HQFX80STREAM CHARCODE)))

                       ;; if character will be at least partly visible, output it

                       (if (\HQFX80.CUR-POS-VISIBLE? HQFX80DATA)
                           then (IF (NEQ CHARCODE (CHARCODE SPACE))
                                   THEN ;; only bitblt real, printing characters -- pilotbbt won't do the right thing with amplified spaces

                                        (\HQFX80.BLT-CHAR CHARCODE CHARWIDTH HQFX80STREAM HQFX80DATA)))
                   (CL:INCF (HQFX80DATA-XPOS HQFX80DATA)
                         CHARWIDTH)

                       ;; if we've passed the margin, DING!, do a newline

                       (if (> (HQFX80DATA-XPOS HQFX80DATA)
                              (HQFX80DATA-RIGHTMARGIN HQFX80DATA))
                           then (\HQFX80.NEWLINE (CHARCODE EOL)
                                     HQFX80STREAM])
```

(**HQFX80.BLT-CHAR**

```
  [LAMBDA (CHARCODE CHARWIDTH HQFX80STREAM HQFX80DATA)                          ; Edited 12-Feb-87 14:17 by jds
```

     ;; puts a character on an HQFX80 stream. Much of the information needed by the BitBlt microcode is prestored by the routines that change it. This
     ;; is kept in the BitBltTable.   (* (DECLARE (LOCALVARS . T)))

```
    (LET* ((CURX (FIXR (HQFX80DATA-XPOS HQFX80DATA)))
           (CHAR8CODE (\CHAR8CODE CHARCODE))
           (RIGHT (+ CURX CHARWIDTH))
           (LEFT NIL)
           (CLIPPINGREGION (HQFX80DATA-CLIPPINGREGION HQFX80DATA))
           (RIGHT-CLIPPING-EDGE (fetch (REGION RIGHT) of CLIPPINGREGION))
           (LEFT-CLIPPING-EDGE (fetch (REGION LEFT) of CLIPPINGREGION))
           (PILOTBBT (HQFX80DATA-PILOTBBT HQFX80DATA)))
```

;;; clip the bitmap to fit the stream's clipping region

                ;; does character overlap right edge of clipping region?

```
        (SETQ RIGHT (MIN RIGHT-CLIPPING-EDGE RIGHT))
```

                ;; does character overlap left edge of clipping region?

```
        (SETQ LEFT (MAX CURX LEFT-CLIPPING-EDGE))
        (COND
           ((AND (< LEFT RIGHT)
                 (NEQ (fetch (PILOTBBT PBTHEIGHT) of PILOTBBT)
                      0))
            (UNINTERRUPTABLY
                (freplace (PILOTBBT PBTDESTBIT) of PILOTBBT with LEFT)
                (freplace (PILOTBBT PBTWIDTH) of PILOTBBT with (- RIGHT LEFT))
                (freplace (PILOTBBT PBTSOURCEBIT) of PILOTBBT with (- (+ \HQFX80.GET-CHARACTER-OFFSET CHAR8CODE
                                                                                HQFX80DATA)
                                                                       LEFT)
                                                                   CURX))

                (\PILOTBITBLT PILOTBBT 0))
            T])
)
```

;; printer code

```
(DEFINEQ
```

(**HQFX80.DUMP-PAGE-BUFFER**

```
  [LAMBDA (BITMAP HQFX80STREAM)                                                 ; Edited 23-Sep-88 10:25 by jds
```

;;; send a bitmap to the FX-80

     ;; how it works: we use a specially created bitblt table (HQFX80DATA-SERIALIZING-PILOTBBT) to turn eight-bit-high by one-bit-wide columns of
     ;; BITMAP into eight-bit-wide by one-bit-high bytes. This extraction is done by \HQFX80.BITMAP-LDB.

```
     (DECLARE (LOCALVARS . T))
     (WITH-HQFX80-DATA (HQFX80DATA HQFX80STREAM)
           (LET* ((WIDTH (BITMAPWIDTH BITMAP))
                  (WIDTH-MINUS-1 (SUB1 WIDTH))
                  [HEIGHT (FIX (TIMES \HQFX80.INCHES-PER-PAGE (if (HQFX80DATA-COMPRESSED? HQFX80DATA)
                                                                  then \HQFX80.LOW-SPEED-DOUBLE-MODE-DPI
                                                                  else \HQFX80.1-TO-1-MODE-DPI]
                  (HEIGHT-MINUS-1 (SUB1 HEIGHT))
                  (BACKING-STREAM (HQFX80DATA-BACKINGSTREAM HQFX80DATA))
                  (BITMAP-BASE (fetch (BITMAP BITMAPBASE) of BITMAP))
                  (BITMAP-WIDTH (fetch (BITMAP BITMAPRASTERWIDTH) of BITMAP))
```

```
                        (MAPPING-TABLE (HQFX80DATA-SERIALIZING-PILOTBBT HQFX80DATA))
                        (BYTE-BOX (HQFX80DATA-SERIALIZING-BOX HQFX80DATA))
                        (SCRATCH-SCANLINE-PILOTBBT (HQFX80DATA-SCRATCH-SCANLINE-PILOTBBT HQFX80DATA))
                        (EIGHT-LINES-BLANK (HQFX80DATA-EIGHT-LINES-BLANK HQFX80DATA))
                        (EIGHT-LINES-BLANK-PILOTBBT (HQFX80DATA-EIGHT-LINES-BLANK-PILOTBBT HQFX80DATA))
                        (COMPRESSED? (HQFX80DATA-COMPRESSED? HQFX80DATA)))
```

;; set the mode

```
                    (\HQFX80.PRINTER-MODE :UNIDIRECTIONAL-ON BACKING-STREAM)
                    (\HQFX80.PRINTER-MODE :EIGHT-SPACING-ON BACKING-STREAM)
```

;; pack the bitmap into FX80 format and ship it

```
                    [for EIGHT-SCANLINE-SWATH from 0 to HEIGHT-MINUS-1 by 8
                        do [COND
                              ((ILESSP (IDIFFERENCE HEIGHT-MINUS-1 EIGHT-SCANLINE-SWATH)
                                       8)                                       ; There are fewer than 8 scan lines left on the page image; only
                                                                               ; advance by that amount.
                                (\HQFX80.PRINTER-MODE :N-SPACING-ON BACKING-STREAM (IDIFFERENCE HEIGHT-MINUS-1
                                                                                              EIGHT-SCANLINE-SWATH]
                              (COND
                                ((\HQFX80.EIGHT-LINES-BLANK? BITMAP-BASE EIGHT-SCANLINE-SWATH BITMAP-WIDTH
                                       SCRATCH-SCANLINE-PILOTBBT EIGHT-LINES-BLANK-PILOTBBT EIGHT-LINES-BLANK)
                                  ;; skip the next eight blank lines
                                  (\HQFX80.PRINTER-MODE :EIGHT-SPACING-ON BACKING-STREAM)
                                  (\HQFX80.ADVANCE-8-LINES HQFX80STREAM))
                                (T  ;; something to print in the next eight scanlines; do so
                                  (\HQFX80.PRINTER-MODE :EIGHT-SPACING-ON BACKING-STREAM)
                                  (\HQFX80.GRAPHICS-MODE WIDTH COMPRESSED? BACKING-STREAM)
                                  (for COLUMN from 0 to WIDTH-MINUS-1 do (\HQFX80.BITMAP-LDB BITMAP-BASE COLUMN
                                                                              EIGHT-SCANLINE-SWATH MAPPING-TABLE
                                                                              BITMAP-WIDTH)
                                                                    (BOUT BACKING-STREAM (\GETBASEBYTE BYTE-BOX 0
                                                                                          )))
                                  (BOUT BACKING-STREAM (CHARCODE CR))
                                  (BOUT BACKING-STREAM (CHARCODE LF]
                    (\HQFX80.PRINTER-MODE :UNIDIRECTIONAL-OFF BACKING-STREAM)
                    (\HQFX80.PRINTER-MODE :TWELVE-SPACING-ON BACKING-STREAM])
```

(**HQFX80.ADVANCE-8-LINES**
```
  [LAMBDA (HQFX80STREAM)                                                        ; Edited 11-Feb-87 11:03 by jds
```

;; advance the printhead 8 raster lines. since we assume that we're in the :eight-spacing-on printermode, just send an LF

```
    (\HQFX80.BOUT HQFX80STREAM (CHARCODE LF])
```

```
)
```

```
(DEFMACRO \HQFX80.EIGHT-LINES-BLANK? (BITMAP-BASE Y-COORD BITMAP-WIDTH-IN-WORDS EIGHT-INTO-ONE-PBBT
                                     SCANLINE-INTO-WORD-PBBT WORD-BOX)
```

;; returns T if the next 8 lines of the bitmap are all blank. This is done by or'ing the 8 scanlines into a scratch bitmap, then or'ing the words of that
;; scanline into a result word, and comparing that to 0. We clear the scanline and word buffers at the end.

```
    `(LET ((EIGHT-INTO-ONE-PBBT ,EIGHT-INTO-ONE-PBBT)
           (SCANLINE-INTO-WORD-PBBT ,SCANLINE-INTO-WORD-PBBT)
           (WORD-BOX ,WORD-BOX)
           (BITMAP-WIDTH-IN-WORDS ,BITMAP-WIDTH-IN-WORDS))
          [FREPLACE (PILOTBBT PBTSOURCE) OF EIGHT-INTO-ONE-PBBT WITH (\ADDBASE ,BITMAP-BASE (TIMES
                                                                                              ,
                                                                                              BITMAP-WIDTH-IN-WORDS
                                                                                              ,Y-COORD]
          (\PILOTBITBLT EIGHT-INTO-ONE-PBBT 0)
          (\PILOTBITBLT SCANLINE-INTO-WORD-PBBT 0)
          (PROG1 (EQ (\GETBASE WORD-BOX 0)
                     0)
                 (\HQFX80.CLEAR-SCANLINE EIGHT-INTO-ONE-PBBT BITMAP-WIDTH-IN-WORDS)
                 (\HQFX80.CLEAR-WORD-BOX WORD-BOX))))
```

```
(DEFMACRO \HQFX80.BITMAP-LDB (BITMAP-BASE X Y PILOTBBT BITMAP-WIDTH-IN-WORDS)
```

;; point the serializing bitblt table at a new column of the bitmap. The X coord increases left to right, the Y coord increases top to bottom, and names
;; the uppermost pixel of the column we're moving.

```
    `(LET ((X ,X)
           (PILOTBBT ,PILOTBBT))
          [FREPLACE (PILOTBBT PBTSOURCE) OF PILOTBBT WITH (\ADDBASE ,BITMAP-BASE (+ (TIMES ,Y
                                                                                          ,
                                                                                          BITMAP-WIDTH-IN-WORDS
                                                                                          )
                                                                                   (FOLDLO X BITSPERWORD]
          (FREPLACE (PILOTBBT PBTSOURCEBIT) OF PILOTBBT WITH (LOGAND 15 X))
          (\PILOTBITBLT PILOTBBT 0)))
```

```
(DEFMACRO \HQFX80.CLEAR-SCANLINE (SCANLINE-PILOTBBT SCANLINE-WIDTH-IN-WORDS)
```

;; clear out the destination of the pilotbbt the fast way - store a zero in its last word and perform an overlapping blt (which runs back to front).

```
'(LET [(SCANLINE (FETCH (PILOTBBT PBTDEST) OF ,SCANLINE-PILOTBBT))
       (LAST-WORD (SUB1 ,SCANLINE-WIDTH-IN-WORDS]
      (\PUTBASE SCANLINE LAST-WORD 0)
      (\BLT SCANLINE (\ADDBASE SCANLINE 1)
            LAST-WORD)))
```

```
(DEFMACRO \HQFX80.CLEAR-WORD-BOX (WORD-BOX)
    '(\PUTBASE ,WORD-BOX 0 0))
```

```
(CL:DEFUN \HQFX80.SEND (PRINTER FILENAME &OPTIONAL OPTIONS)
```
;; send the file designated by FILENAME to PRINTER, obeying OPTIONS. Since we only have one fx-80 per machine, ignore PRINTER and send
;; to HQFX80-DEFAULT-DESTINATION
```
    (DECLARE (GLOBALVARS HQFX80-DEFAULT-DESTINATION))
    [LET ((COPIES (OR (LISTGET OPTIONS '%#COPIES)
                      1)))
         (FOR COPY FROM 1 TO COPIES DO
```
;; allow the user to abort it while running
```
                                    (WITH-ABORT-WINDOW ((THIS.PROCESS)
                                                        FILENAME PRINTER COPY)
                                        (COPYFILE FILENAME HQFX80-DEFAULT-DESTINATION
                                                  '((TYPE HQFX80])
```

```
(CL:DEFUN MAKE-HQFX80 (FILE HQFX80FILE &OPTIONAL FONTS HEADING TABS OPTIONS)
```
;; turn FILE into an HQFX80 master
```
    (TEXTTOIMAGEFILE FILE HQFX80FILE 'HQFX80 FONTS HEADING TABS OPTIONS))
```

```
(CL:DEFUN HQFX80FILEP (HQFX80FILE?)
```
;; is FILE (a filename or stream) an hqfx80 file?
```
    [LET [(FILE-TYPE (GETFILEINFO HQFX80FILE? 'TYPE]
         (IF (EQ FILE-TYPE 'HQFX80)
```
`                                                                                     ; if file has a type, and type=HQFX80, we win`
```
             THEN
                 T
             ELSE
```
`                                                                                     ; no filetype or filetype not HQFX80, so read the file`
```
                 (LET [(STREAM (OPENSTREAM (INTERLISP-NAMESTRING HQFX80FILE?)
                                          'INPUT
                                          'OLD
                                          '(SEQUENTIAL]
```
;; file looks like ESC@<signature>...
```
                      (PROG1 [AND (> (GETFILEINFO STREAM 'LENGTH)
                                     (+ 2 (NCHARS \HQFX80.FILE-SIGNATURE)))
                                  (EQ (CHARCODE ESC)
                                      (BIN STREAM))
                                  (EQ (CHARCODE @)
                                      (BIN STREAM))
                                  (FOR CH INSTRING \HQFX80.FILE-SIGNATURE ALWAYS (EQ CH (BIN STREAM]
                             (CLOSEF STREAM])
```

;; window hardcopy

```
(DEFINEQ
```

(**HQFX80.BITMAP-FILE**
```
    [LAMBDA (FILE BITMAP SCALEFACTOR REGION ROTATION TITLE)
```
`                                                              ; Edited  1-Jun-87 13:10 by Snow`

;; print a bitmap on the fx-80. ignore SCALEFACTOR and ROTATION for now.
```
    (LET* ((HQFX80STREAM (OPENIMAGESTREAM FILE 'HQFX80))
           (NEWBITMAP (COND
                         (REGION (BITMAPCREATE (fetch (REGION WIDTH) of REGION)
                                               (fetch (REGION HEIGHT) of REGION)))
                         (T BITMAP)))
           (WIDTH (BITMAPWIDTH NEWBITMAP))
           (HEIGHT (BITMAPHEIGHT NEWBITMAP))
           (PAGE-REGION (DSPCLIPPINGREGION NIL HQFX80STREAM))
           (PAGE-WIDTH (fetch (REGION WIDTH) of PAGE-REGION))
           (PAGE-HEIGHT (fetch (REGION HEIGHT) of PAGE-REGION)))
```
;; clip the bitmap, if requested
```
          (AND REGION (BITBLT BITMAP (fetch (REGION LEFT) of REGION)
                              (fetch (REGION BOTTOM) of REGION)
                              NEWBITMAP))
          (LET* ((PORTRAIT-OVERHANG (- WIDTH (fetch (REGION WIDTH) of PAGE-REGION)))
                 (LANDSCAPE-OVERHANG (- HEIGHT (fetch (REGION WIDTH) of PAGE-REGION)))
                 (BITS-LOST (AND (> PORTRAIT-OVERHANG 0)
                                 (> LANDSCAPE-OVERHANG 0)))
                 (LANDSCAPE-PRINT (> PORTRAIT-OVERHANG LANDSCAPE-OVERHANG)))
```
;; print the title of the image on the top of the page

```
                        (LET* ((IMAGE-TITLE (OR TITLE "Window Image"))
                               (TITLE-REGION (STRINGREGION IMAGE-TITLE HQFX80STREAM)))
                              (MOVETO (/ (- PAGE-WIDTH (fetch (REGION WIDTH) of TITLE-REGION))
                                         2)
                                      (- (- PAGE-HEIGHT 1)
                                         (FONTPROP HQFX80STREAM 'HEIGHT))
                                      HQFX80STREAM)
                              (PRIN1 IMAGE-TITLE HQFX80STREAM))
```
                ;; blt the bitmap onto the page. use replace mode so it will obscure title if need be
```
                        [COND
                            (BITS-LOST
```
                                ;; apologize and blt as much as will fit
```
                                (PRINTOUT PROMPTWINDOW "Bitmap is larger than FX-80 page - " "image will be clipped" T
                                        )
                                (BITBLT NEWBITMAP NIL NIL HQFX80STREAM 0 0 NIL NIL 'INPUT 'REPLACE))
                            (T
```
                                ;; center it on the page
                                ;; if there is more overhang in portrait than in landscape - rotate it remember to swap the height and width.
```
                                (AND LANDSCAPE-PRINT (SETQ NEWBITMAP (ROTATE-BITMAP NEWBITMAP))
                                     (swap WIDTH HEIGHT))
                                (BITBLT NEWBITMAP NIL NIL HQFX80STREAM (/ (- PAGE-WIDTH WIDTH)
                                                                          2)
                                        (/ (- PAGE-HEIGHT HEIGHT)
                                           2)
                                        NIL NIL 'INPUT 'REPLACE]
                        (CLOSEF HQFX80STREAM])
```

(**HQFX80.CONVERT-TEDIT**
```
  [LAMBDA (TEDIT-FILE IMAGESTREAM)                                              ; Edited 11-Dec-86 17:24 by hdj
```
    ;; Send the text to the printer.
```
    (SETQ TEDIT-FILE (OPENTEXTSTREAM TEDIT-FILE))
    (TEDIT.FORMAT.HARDCOPY TEDIT-FILE IMAGESTREAM T NIL NIL NIL 'HQFX80)
    (CLOSEF? IMAGESTREAM)
    IMAGESTREAM])
```
)

;; character transmission method

(DEFINEQ

(**HQFX80.BOUT**
```
  [LAMBDA (HQFX80SSTREAM BYTE)                                              (* hdj " 7-Nov-86 17:18")
```
    ;; send a byte to the fx80
```
    (WITH-HQFX80-DATA (DATA HQFX80SSTREAM)
          (BOUT (HQFX80DATA-BACKINGSTREAM DATA)
                BYTE])
```
)

;; handling font-information caching

(DEFINEQ

(**HQFX80.FIX-LINE-LENGTH**
```
  [LAMBDA (HQFX80STREAM)                                                    (* hdj "14-Nov-86 17:15")
```
    ;; HQFX80STREAM is a stream of type hqfx80. Called by RIGHTMARGIN LEFTMARGIN and \\hqfx80.fix-font to update the LINELENGTH field in
    ;; the stream. Also called when the stream is created.
```
    (WITH-HQFX80-DATA (DATA HQFX80STREAM)
          (freplace (STREAM LINELENGTH) of HQFX80STREAM with (MIN MAX.SMALLP (MAX 1 (IQUOTIENT (- (
                                                                                   HQFX80DATA-RIGHTMARGIN
                                                                                     DATA)
                                                                                  (
                                                                                   HQFX80DATA-LEFTMARGIN
                                                                                     DATA))
                                                                              (fetch (FONTDESCRIPTOR
                                                                                       FONTAVGCHARWIDTH
                                                                                       )
                                                                                 of (HQFX80DATA-FONT
                                                                                      DATA])
```

(**HQFX80.FIX-FONT**
```
  [LAMBDA (HQFX80STREAM HQFX80DATA)                                          (* hdj "10-Nov-86 16:37")
```
    ;; used to fix up those parts of the bitblt table which depend upon the FONT.
```
    (\\HQFX80.INVALIDATE-CACHE HQFX80DATA)
    (\\HQFX80.FIX-LINE-LENGTH HQFX80STREAM])
```

(**HQFX80.FIX-Y**
```
  [LAMBDA (HQFX80DATA CSINFO)                                               ; Edited 12-Feb-87 11:46 by jds

    ;; makes that part of the bitblt table of an HQFX80 stream which deals with the Y information consistent.  This is called from
    ;; \\HQFX80.change-charset whenever a character is being printed and the charset/y-position caches are invalid

    (PROG ((PBT (HQFX80DATA-PILOTBBT HQFX80DATA))
           (Y (HQFX80DATA-YPOS HQFX80DATA))
            TOP CHARTOP BM)
          [SETQ CHARTOP (FIXR (+ Y (CL:SETF (HQFX80DATA-CHARSET-ASCENT-CACHE HQFX80DATA)
                                            (ffetch CHARSETASCENT of CSINFO]
              [freplace PBTDEST of PBT with (\ADDBASE (fetch BITMAPBASE of (SETQ BM (HQFX80DATA-BACKINGBITMAP HQFX80DATA
                                                                                    )))
                                                     (TIMES (ffetch BITMAPRASTERWIDTH of BM)
                                                            (\SFInvert BM (SETQ TOP
                                                                            (FIXR (MAX (MIN (fetch (REGION TOP)
                                                                                                of (
                                                                                                    HQFX80DATA-CLIPPINGREGION
                                                                                                    HQFX80DATA))
                                                                                          CHARTOP)
                                                                                     0]
              [freplace PBTSOURCE of PBT with (\ADDBASE (ffetch BITMAPBASE of (SETQ BM (ffetch (CHARSETINFO CHARSETBITMAP)
                                                                                              of CSINFO)))
                                                       (TIMES (ffetch BITMAPRASTERWIDTH of BM)
                                                              (CL:SETF (HQFX80DATA-CHARHEIGHTDELTA HQFX80DATA)
                                                                       (FIXR (MIN (MAX (- CHARTOP TOP)
                                                                                      0)
                                                                                 MAX.SMALL.INTEGER]
              (freplace PBTHEIGHT of PBT with (FIXR (MAX [- TOP (MAX (- Y (CL:SETF (HQFX80DATA-CHARSET-DESCENT-CACHE
                                                                                   HQFX80DATA)
                                                                         (ffetch CHARSETDESCENT of CSINFO)))
                                                                    (fetch (REGION BOTTOM) of (
                                                                                              HQFX80DATA-CLIPPINGREGION
                                                                                              HQFX80DATA]
                                                   0]))
)
```

(DEFMACRO **HQFX80.INVALIDATE-CACHE** (HQFX80DATA)

```
    ;; marks the stream as needing to have its cached fields recomputed.  used when font changes, etc.

    `(PROGN (CL:SETF (HQFX80DATA-CHARSET-CACHE ,HQFX80DATA)
                     MAX.SMALLP)
            (CL:SETF (HQFX80DATA-CHARSET-ASCENT-CACHE ,HQFX80DATA)
                     MAX.SMALLP)))
```

(DEFMACRO **HQFX80.INVALIDATE-FONT-CACHE** (HQFX80DATA)
```
    `(PROGN (CL:SETF (HQFX80DATA-CHARSET-CACHE ,HQFX80DATA)
                     MAX.SMALLP)
            (CL:SETF (HQFX80DATA-CHARSET-ASCENT-CACHE ,HQFX80DATA)
                     MAX.SMALLP)))
```

(DEFMACRO **HQFX80.GET-CACHED-CHAR-WIDTH** (CHARCODE HQFX80DATA)

```
    ;; get the cached image width of CHARCODE

    `(\FGETIMAGEWIDTH (HQFX80DATA-IMAGE-WIDTHS-CACHE ,HQFX80DATA)
          ,CHARCODE))
```

(DEFMACRO **HQFX80.GET-CHARACTER-OFFSET** (CHAR8CODE HQFX80DATA)
```
    `(\GETBASE (HQFX80DATA-OFFSETS-CACHE ,HQFX80DATA)
          ,CHAR8CODE))
```

```
;; auxiliary functions
```

(CL:DEFUN **HQFX80.GRAPHICS-MODE** (ROWS COMPRESSED? BACKING-STREAM)

```
    ;; put the FX-80 in some graphics mode

    (BOUT BACKING-STREAM (CHARCODE ESC))
    (BOUT BACKING-STREAM (CHARCODE *))
    (BOUT BACKING-STREAM                                                    ; compressed prints at 120 dpi, regular at 72
          (if COMPRESSED?
              then 1
              else 5))
    (BOUT BACKING-STREAM (IREMAINDER ROWS 256))
    (BOUT BACKING-STREAM (FOLDLO ROWS 256)))
```

```
(DEFINEQ
```

(**HQFX80.PRINTER-MODE**
```
  [LAMBDA (FX80-MODE STREAM N-SPACING)                                      ; Edited 23-Sep-88 10:21 by jds

    ;; put the FX80 printer in some mode

    (CL:FLET [(SEND-PRINTER-COMMAND (COMMAND-STRING STREAM)
```

```
                     ;; Send an ESC, to tell the printer there is to be a mode change, and then the specific mode change byte
                     (BOUT STREAM (CHARCODE ESC))
                     (for CHAR instring COMMAND-STRING do (BOUT STREAM CHAR]
               (SELECTQ FX80-MODE
                   (:BOLD-ON (SEND-PRINTER-COMMAND "E" STREAM))
                   (:BOLD-OFF (SEND-PRINTER-COMMAND "F" STREAM))
                   (:COMPRESSED-ON
                       (SEND-PRINTER-COMMAND (CHARACTER 15)
                               STREAM))
                   (:COMPRESSED-OFF
                       (BOUT STREAM 18))
                   (:ELITE-ON (SEND-PRINTER-COMMAND "M" STREAM))
                   (:ELITE-OFF (SEND-PRINTER-COMMAND "P" STREAM))
                   (:ITALIC-ON (SEND-PRINTER-COMMAND "4" STREAM))
                   (:ITALIC-OFF (SEND-PRINTER-COMMAND "5" STREAM))
                   (:PICA-ON (SEND-PRINTER-COMMAND (CONCAT "P" (CHARACTER 18))
                                   STREAM))
                   (:SUBSCRIPT-ON
                       (SEND-PRINTER-COMMAND "S0" STREAM))
                   (:SCRIPT-OFF (SEND-PRINTER-COMMAND "T" STREAM))
                   (:SUPERSCRIPT-ON
                       (SEND-PRINTER-COMMAND "S1" STREAM))
                   (:EXPAND-ON (SEND-PRINTER-COMMAND "W1" STREAM))
                   (:EXPAND-OFF (SEND-PRINTER-COMMAND "W0" STREAM))
                   (:PROPORTIONAL-ON
                       (SEND-PRINTER-COMMAND "p1" STREAM))
                   (:PROPORTIONAL-OFF
                       (SEND-PRINTER-COMMAND "p0" STREAM))
                   (:UNIDIRECTIONAL-ON
                       (SEND-PRINTER-COMMAND "U1" STREAM))
                   (:UNIDIRECTIONAL-OFF
                       (SEND-PRINTER-COMMAND "U0" STREAM))
                   (:N-SPACING-ON                                       ; Space n/72 of an inch on LF.
                       (SEND-PRINTER-COMMAND (CONCAT "A" (CHARACTER N-SPACING))
                               STREAM))
                   (:SEVEN-SPACING-ON
                       (SEND-PRINTER-COMMAND "1" STREAM))
                   (:EIGHT-SPACING-ON
                       (SEND-PRINTER-COMMAND (CONCAT "A" (CHARACTER 8))
                               STREAM))
                   (:NINE-SPACING-ON                                    ; Space by 9 print dots per LF.  Mostly for graphics mode used in
                                                                        ; HQ FX-80.
                       (SEND-PRINTER-COMMAND "0" STREAM))
                   (:TWELVE-SPACING-ON                                  ; Restore normal 1/6" spacing
                       (SEND-PRINTER-COMMAND "2" STREAM))
                   (:NO-SKIP (SEND-PRINTER-COMMAND "O" STREAM))
                   NIL])
)


(DEFMACRO WITH-HQFX80-DATA ((VAR-NAME STREAM)
                                   &BODY
                                   (BODY DECLS ENV))
    `(LET [(,VAR-NAME (FETCH (STREAM IMAGEDATA) OF ,STREAM]
         ,@DECLS
         ,@BODY))
```

;; and miscellany

```
(DECLARE%: EVAL@COMPILE

(RPAQ \HQFX80.FILE-SIGNATURE "HQFX-80/Xerox/1.0 ")

(RPAQQ \HQFX80.1-TO-1-MODE-DPI 72)

(RPAQQ \HQFX80.LOW-SPEED-DOUBLE-MODE-DPI 120)

(CONSTANTS (\HQFX80.FILE-SIGNATURE "HQFX-80/Xerox/1.0 ")
       (\HQFX80.1-TO-1-MODE-DPI 72)
       (\HQFX80.LOW-SPEED-DOUBLE-MODE-DPI 120))
)

(RPAQ? \HQFX80.INCHES-PER-PAGE 11)

(RPAQ? \HQFX80.INCHES-PER-LINE 8.5)

(RPAQ? HQFX80-DEFAULT-DESTINATION "{TTY}")

(RPAQ? HQFX80-FONT-EXTENSIONS DISPLAYFONTEXTENSIONS)

(RPAQ? HQFX80-FONT-DIRECTORIES DISPLAYFONTDIRECTORIES)

(RPAQ? HQFX80-FONT-COERCIONS DISPLAYFONTCOERCIONS)

(RPAQ? HQFX80-MISSING-FONT-COERCIONS MISSINGDISPLAYFONTCOERCIONS)
```

```
(RPAQQ FX80-PRINTCOMS
       (;; The FXPrinter emulator
        (COMS ;; top level routine
              (FUNCTIONS FX80-PRINT))
        (COMS ;; how to print bitmaps
              (FUNCTIONS FX80-PRINT.BITMAP)
              (FUNCTIONS FX80-PRINT.PRINT-BITMAP FX80-PRINT.PRINT-BITMAP-PORTRAIT
                     FX80-PRINT.PRINT-BITMAP-LANDSCAPE))
        (COMS ;; how to print files
              (FUNCTIONS FX80-PRINT.FILE))))
```

;; The FXPrinter emulator

;; top level routine

```
(CL:DEFUN FX80-PRINT (THING-TO-PRINT &KEY LANDSCAPE? COMPRESS? HIGH-QUALITY?)
   "Prints thing-to-print on the FX-80 printer"
   (CL:ETYPECASE THING-TO-PRINT
       ((OR WINDOW BITMAP) (FX80-PRINT.BITMAP THING-TO-PRINT LANDSCAPE? COMPRESS?))
       ((OR CL:SYMBOL STRING PATHNAME) (FX80-PRINT.FILE THING-TO-PRINT HIGH-QUALITY?)))
   THING-TO-PRINT)
```

;; how to print bitmaps

```
(CL:DEFUN FX80-PRINT.BITMAP (BITMAP-OR-WINDOW LANDSCAPE? COMPRESS?)
   "Prints a bitmap or window on the FX-80 printer"
   (CL:ETYPECASE BITMAP-OR-WINDOW
       (WINDOW (LET* [(WINDOW-REGION (DSPCLIPPINGREGION NIL BITMAP-OR-WINDOW))
                      (BM (BITMAPCREATE (FETCH (REGION WIDTH) OF WINDOW-REGION)
                                        (FETCH (REGION HEIGHT) OF WINDOW-REGION]
                     (BITBLT BITMAP-OR-WINDOW NIL NIL BM)
                     (FX80-PRINT.BITMAP BM LANDSCAPE? COMPRESS?)))
       (BITMAP (FX80-PRINT.PRINT-BITMAP BITMAP-OR-WINDOW LANDSCAPE? COMPRESS?))))
```

```
(CL:DEFUN FX80-PRINT.PRINT-BITMAP (BITMAP LANDSCAPE? COMPRESS?)
   "Print a bitmap on the FX-80, either landscape or portrait"
   (IF LANDSCAPE?
       THEN (FX80-PRINT.PRINT-BITMAP-LANDSCAPE BITMAP COMPRESS?)
     ELSE (FX80-PRINT.PRINT-BITMAP-PORTRAIT BITMAP COMPRESS?)))
```

```
(CL:DEFUN FX80-PRINT.PRINT-BITMAP-PORTRAIT (BITMAP COMPRESS?)
   "Prints a bitmap on the FX-80 in portrait mode"
   [LET ((HQFX80STREAM (OPENIMAGESTREAM HQFX80-DEFAULT-DESTINATION 'HQFX80 (LIST 'COMPRESSED COMPRESS?)))
         (WIDTH (BITMAPWIDTH BITMAP))
         (HEIGHT (BITMAPHEIGHT BITMAP)))
        (WITH-HQFX80-DATA (DATA HQFX80STREAM)
            (LET ((PAGE-WIDTH (fetch (REGION WIDTH) of (HQFX80DATA-CLIPPINGREGION DATA)))
                  (PAGE-HEIGHT (fetch (REGION HEIGHT) of (HQFX80DATA-CLIPPINGREGION DATA))
                      (HQFX80DATA-CLIPPINGREGION DATA)))

                ;; center it if possible

                (BITBLT BITMAP NIL NIL HQFX80STREAM (MAX 0 (/ (- PAGE-WIDTH WIDTH)
                                                              2))
                        (MAX 0 (/ (- PAGE-HEIGHT HEIGHT)
                                  2))
                        NIL NIL 'INPUT 'REPLACE)
                (CLOSEF HQFX80STREAM])
```

```
(CL:DEFUN FX80-PRINT.PRINT-BITMAP-LANDSCAPE (BITMAP COMPRESS?)
   "Prints a bitmap on the FX-80 in landscape mode"
   [LET ((HQFX80STREAM (OPENIMAGESTREAM HQFX80-DEFAULT-DESTINATION 'HQFX80 (LIST 'COMPRESSED COMPRESS?)))
         (WIDTH (BITMAPHEIGHT BITMAP))
         (HEIGHT (BITMAPWIDTH BITMAP))
         (ROTATED-BITMAP (ROTATE-BITMAP BITMAP)))
        (WITH-HQFX80-DATA (DATA HQFX80STREAM)
            (LET ((PAGE-WIDTH (fetch (REGION WIDTH) of (HQFX80DATA-CLIPPINGREGION DATA)))
                  (PAGE-HEIGHT (fetch (REGION HEIGHT) of (HQFX80DATA-CLIPPINGREGION DATA))
                      (HQFX80DATA-CLIPPINGREGION DATA)))
                (BITBLT ROTATED-BITMAP NIL NIL HQFX80STREAM (MAX 0 (/ (- PAGE-WIDTH WIDTH)
                                                                     2))
                        (MAX 0 (/ (- PAGE-HEIGHT HEIGHT)
                                  2))
                        NIL NIL 'INPUT 'REPLACE)
                (CLOSEF HQFX80STREAM])
```

;; how to print files

```
(CL:DEFUN FX80-PRINT.FILE (FILE-NAME HIGH-QUALITY?)
    "Prints a file on the FX-80"
    (SEND.FILE.TO.PRINTER (INTERLISP-NAMESTRING FILE-NAME)
            (IF HIGH-QUALITY?
                THEN 'HQFX80
              ELSE 'FASTFX80)))
```

;; common routines

```
(DEFMACRO WITH-ABORT-WINDOW ((PROCESS FILE-NAME PRINTER-NAME COPY#)
                                          &BODY
                                          (FORMS DECLS))
    "executes FORMS, allowing termination by menu selection"
    '(LET [(WINDOW (\FX80.CREATE-SEND-ABORT-WINDOW ,PROCESS ,FILE-NAME ,PRINTER-NAME ,COPY#]
          (CL:UNWIND-PROTECT
              (PROGN ,@DECLS (BLOCK 3000)
                      ,@FORMS)
              (CLOSEW WINDOW))))
```

```
(CL:DEFUN \FX80.CREATE-SEND-ABORT-WINDOW (SENDING-PROCESS FILE-OR-STREAM PRINTER-NAME COPY#)
    (LET* [(DOCUMENT-TYPE-AND-NAME-STRING (IF (STREAMP FILE-OR-STREAM)
                                              THEN (IF (FETCH (STREAM NAMEDP) OF FILE-OR-STREAM)
                                                      THEN (CONCAT "the file " (FULLNAME FILE-OR-STREAM))
                                                    ELSE "an unnamed document")
                                            ELSE FILE-OR-STREAM))
          (WINDOW-WIDTH (WIDTHIFWINDOW 270))
          (WINDOW-HEIGHT (HEIGHTIFWINDOW 120))
          (ABORT-MENU-ITEM "Abort")
          (ABORT-MENU-FONT (FONTCREATE 'GACHA 12 'BRR))
          (ABORT-WINDOW (CREATEW (CREATEREGION (RAND 0 (- SCREENWIDTH WINDOW-WIDTH))
                                              (- SCREENHEIGHT WINDOW-HEIGHT)
                                              WINDOW-WIDTH WINDOW-HEIGHT)))
          (ABORT-WINDOW-FONT (DSPFONT NIL ABORT-WINDOW))
          (BOLD-ABORT-WINDOW-FONT (FONTCOPY ABORT-WINDOW-FONT 'WEIGHT 'BOLD]
        (PRINTOUT ABORT-WINDOW "Sending copy " COPY# " of " .FONT BOLD-ABORT-WINDOW-FONT
                DOCUMENT-TYPE-AND-NAME-STRING .FONT ABORT-WINDOW-FONT " to " .FONT BOLD-ABORT-WINDOW-FONT
                PRINTER-NAME .FONT ABORT-WINDOW-FONT "." T)
        (PRINTOUT ABORT-WINDOW "Select %"Abort%" below to stop printing " " this and all subsequent copies." T)
        (ADDMENU (CREATE MENU
                     ITEMS _ '[(,ABORT-MENU-ITEM (PROGN (PROCESS.EVAL ,SENDING-PROCESS '(ERROR!))
                                                          (PRINTOUT ,ABORT-WINDOW T "... printing aborted.")
                                                          (BLOCK 2000)
                                                          (CLOSEW ,ABORT-WINDOW))
                                            ,(CONCAT "Stops printing this and all subsequent copies of "
                                                      DOCUMENT-TYPE-AND-NAME-STRING "."]
                     MENUFONT _ ABORT-MENU-FONT)
                 ABORT-WINDOW
                 (CREATEPOSITION (/ (- WINDOW-WIDTH (STRINGWIDTH ABORT-MENU-ITEM ABORT-MENU-FONT))
                                    2)
                          20))
          ABORT-WINDOW))
```

```
(CL:DEFUN \ADD-TO-FONTPROFILE (FONTPROFILE FONTCLASS DEVICE FONT-DESCRIPTION)
    ;; sets the DEVICE component of the FONTCLASS entry of FONTPROFILE to be FONT-DESCRIPTION.

    (LET ((BUCKET (FASSOC FONTCLASS FONTPROFILE)))
        (IF (NULL BUCKET)
            THEN (ERROR "No such fontclass as " FONTCLASS)
          ELSE ;; the bucket looks like

                ;; (fontclass prettyfont# displayfont pressfont interpressfont

                ;;     (dev1 dev1-font) (dev2 dev2-font) ... )

                [SELECTQ DEVICE
                    (DISPLAY (CL:SETF (CL:THIRD BUCKET)
                                    FONT-DESCRIPTION))
                    (PRESS (CL:SETF (CL:FOURTH BUCKET)
                                  FONT-DESCRIPTION))
                    (INTERPRESS (CL:SETF (CL:FIFTH BUCKET)
                                        FONT-DESCRIPTION))
                    (DESTRUCTURING-BIND (CLASS-NAME PRETTY-FONT# DISPLAY-FONT PRESS-FONT INTERPRESS-FONT
                                                                                      . A-LIST)
                          BUCKET
                          (IF (NULL A-LIST)
                              THEN (RPLACD (LAST BUCKET)
                                          (LIST (LIST DEVICE FONT-DESCRIPTION)))
                            ELSE (PUTASSOC DEVICE (LIST FONT-DESCRIPTION)
                                          A-LIST]
                BUCKET)))
```

```
(CL:DEFUN \GET-FROM-FONTPROFILE (FONTPROFILE FONTCLASS DEVICE)
```

```
        ;; Retunrs the DEVICE component of the FONTCLASS entry of FONTPROFILE.
    [LET ((BUCKET (FASSOC FONTCLASS FONTPROFILE)))
         (IF  (NULL BUCKET)
             THEN (ERROR "No such fontclass as " FONTCLASS)
           ELSE  ;; the bucket looks like
                 ;; (fontclass prettyfont# displayfont pressfont interpressfont
                 ;;      (dev1 dev1-font) (dev2 dev2-font) ... )
                 (SELECTQ DEVICE
                     (DISPLAY (CL:THIRD BUCKET))
                     (PRESS (CL:FOURTH BUCKET))
                     (INTERPRESS (CL:FIFTH BUCKET))
                     (DESTRUCTURING-BIND (CLASS-NAME PRETTY-FONT# DISPLAY-FONT PRESS-FONT INTERPRESS-FONT
                                                                                            . A-LIST)
                            BUCKET
                          (IF (NULL A-LIST)
                              THEN NIL
                            ELSE (CADR (FASSOC DEVICE A-LIST]))
```

;;; initialization

(DECLARE%: DONTEVAL@LOAD DOCOPY

(**HQFX80.INIT**)

(**FASTFX80.INIT**)
)

(PUTPROPS **FX-80DRIVER FILETYPE** CL:COMPILE-FILE)

(PUTPROPS **FX-80DRIVER COPYRIGHT** ("Venue & Xerox Corporation" 1986 1987 1988 1990))

FUNCTION INDEX

MACRO INDEX

VARIABLE INDEX

CONSTANT INDEX

STRUCTURE INDEX

PROPERTY INDEX