File created: 9-Feb-89 13:52:01 **{ERINYES}<LISPUSERS>MEDLEY>NOVAFONT.;5**

changes to:
```
(FNS \\READNOVAFONTFILE.IP)
(VARS NOVAFONTCOMS)
```

previous date: 8-Feb-89 11:09:51 {ERINYES}<LISPUSERS>MEDLEY>NOVAFONT.;3

Read Table: XCL

Package: INTERLISP

Format: XCCS

```
(RPAQQ NOVAFONTCOMS
       (
```

;;; user callable functions (either load-on-demand or load them all

```
        (FNS NOTICE-NOVAFONT-FILE LOAD-NOVAFONT-FILE)
```

;;; the parts necessary for using with FONTCREATE

```
        (FNS \\READNOVAFONTFILE.DISPLAY \\READNOVAFONTFILE.IP)
```

;;; modified versions of functions from the default font handling system

```
        (FNS \\READDISPLAYFONTFILE.NOVA \\CREATECHARSET.IP.NOVA)
```

;;; the parts for general hacking of the NOVAFONT files

```
        (FNS DESCRIBE-FONT SELECT-FONT ENUMERATE-FONTS VIEWPOINT-FONT-FILE-P)
        (VARS (*WARN-ON-KERNING* NIL))
        (GLOBALVARS *WARN-ON-KERNING*)
```

;;; where the NovaFont files are likely to be

```
        (VARIABLES *NOVAFONT-PATHNAME-DEFAULTS*)
```

;;; things for dealing with the structure of what we read

```
        (MACROS READSWAPPEDFIXP)
        (FNS READ-BLOCK-OF-BYTES READ-NOVAFONT-CHARACTERSET READ-NOVAFONT-FILEHEADER READ-NOVAFONT-FONTHEADER
             \\TEXTBLT)
```

;;; the datastructures that we use and their sizes

```
        (DECLARE\: EVAL@COMPILE DONTCOPY (RECORDS FONTTREENODEBLOCK CHARSETBLOCK FONTDESCRIPTION)
               (CONSTANTS (FONTTREENODEBLOCKBYTESIZE (CONSTANT (ITIMES BYTESPERWORD (INDEXF (FETCH (
                                                                                          FONTTREENODEBLOCK

                                                                                 DUMMY-LAST-FIELD-DONT-REFERENCE-THIS
                                                                                                              )
                                                                                                          OF T)))))
                      (CHARSETBLOCKBYTESIZE (CONSTANT (ITIMES BYTESPERWORD (INDEXF (FETCH (CHARSETBLOCK
                                                                                 DUMMY-LAST-FIELD-DONT-REFERENCE-THIS
                                                                                                          )
                                                                                               OF T)))))
                      (FONTDESCRIPTIONBYTESIZE (CONSTANT (ITIMES BYTESPERWORD (INDEXF (FETCH (FONTDESCRIPTION

                                                                                 DUMMY-LAST-FIELD-DONT-REFERENCE-THIS
                                                                                                        )
                                                                                                     OF T)))))))
        (DECLARE\: EVAL@COMPILE DONTEVAL@LOAD DOCOPY (INITRECORDS FONTTREENODEBLOCK CHARSETBLOCK FONTDESCRIPTION
                                                                   ))
```

;;; the mapping from font family number to font family name for those fonts which don't have the name embedded in the font file.

```
        (CONSTANTS \\NOVAFONTFAMILYNAMES)
```

;;; initialize the "noticed" fonts structure and set up the extensions so we can use the font files

```
        (VARIABLES *NOVAFONT-INFO*)
```

;;; correct some omissions in the family aliases and printwheel fonts

```
        (P (LISTPUT INTERPRESSFAMILYALIASES 'XEROXLOGO 'LOGOTYPES-XEROX)
           (|pushnew| INTERPRESSPRINTWHEELFAMILIES 'SCIENTIFICTHIN 'OCRB 'OCRA))
```

;;; some things we need for compiling. Also need EXPORTS.ALL

```
        (DECLARE\: EVAL@COMPILE DONTCOPY (FILES (LOADCOMP)
                                                INTERPRESS LLCHAR))
```

;;; install this:

```
        (DECLARE\: DONTEVAL@LOAD DOCOPY (P (MOVD? '\\READDISPLAYFONTFILE '\\NO-NOVA-READDISPLAYFONTFILE)
                                           (MOVD? '\\CREATECHARSET.IP '\\NO-NOVA-CREATECHARSET.IP)
                                           (MOVD '\\READDISPLAYFONTFILE.NOVA '\\READDISPLAYFONTFILE)
                                           (MOVD '\\CREATECHARSET.IP.NOVA '\\CREATECHARSET.IP)))
```

;;; some hints for the compiler (system generated)

```
        (DECLARE\: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA)
                                                                               (NLAML)
                                                                               (LAMA READ-NOVAFONT-FONTHEADER
                                                                                     READ-NOVAFONT-FILEHEADER)))))
```

;;; user callable functions (either load-on-demand or load them all

```
(DEFINEQ

(NOTICE-NOVAFONT-FILE
  (LAMBDA (NOVAFONT-FILENAME NOTRACE)                            ; Edited  6-Dec-88 12:22 by Briggs
    (CL:WITH-OPEN-FILE (NOVAFONT-STREAM NOVAFONT-FILENAME :INPUT)
           (LET ((MINIMAL-FILE-NAME (CL:ENOUGH-NAMESTRING (CL:MAKE-PATHNAME :VERSION NIL :DEFAULTS
                                                                         NOVAFONT-STREAM)
                                           *NOVAFONT-PATHNAME-DEFAULTS*))
                 NUMBER-OF-FONTS FONT-OFFSETS)
                (CL:MULTIPLE-VALUE-SETQ (NUMBER-OF-FONTS FONT-OFFSETS)
                       (READ-NOVAFONT-FILEHEADER NOVAFONT-STREAM))
                (|for| FONT-NUMBER |from| 1 |to| NUMBER-OF-FONTS |bind| FONT-NAME NOVAFONT-DESCRIPTOR
                                                                   CHARACTER-SET-OFFSETS FONT-FACE CHARACTER-SETS
                                                                   FAMILY-INFO SIZE-INFO FACE-INFO
                   |do| (CL:MULTIPLE-VALUE-SETQ (FONT-NAME NOVAFONT-DESCRIPTOR CHARACTER-SET-OFFSETS)
                               (READ-NOVAFONT-FONTHEADER NOVAFONT-STREAM (ELT FONT-OFFSETS FONT-NUMBER)))
                        (SETQ FONT-FACE (+ (|fetch| (FONTDESCRIPTION EMPHASIS) |of| NOVAFONT-DESCRIPTOR)
                                           (CL:* 2 (|fetch| (FONTDESCRIPTION WEIGHT) |of| NOVAFONT-DESCRIPTOR))))

                        ;; was (list (cl:ecase (|fetch| (fontdescription weight) |of| novafont-descriptor) (0 'light) (1 'medium) (2 'bold)) (cl:ecase
                        ;; (|fetch| (fontdescription emphasis) |of| novafont-descriptor) (0 'regular) (1 'italic)) 'regular)

                        (CL:UNLESS NOTRACE
                            (CL:FORMAT T "~A~D~A~S" FONT-NAME (|fetch| (FONTDESCRIPTION SIZE) |of|
                                                                                                  NOVAFONT-DESCRIPTOR
                                                                               )
                                   (LIST (CL:ECASE (|fetch| (FONTDESCRIPTION WEIGHT) |of| NOVAFONT-DESCRIPTOR)
                                                   (0 'LIGHT)
                                                   (1 'MEDIUM)
                                                   (2 'BOLD))
                                               (CL:ECASE (|fetch| (FONTDESCRIPTION EMPHASIS) |of| NOVAFONT-DESCRIPTOR)
                                                   (0 'REGULAR)
                                                   (1 'ITALIC)))
                                   (|for| I |from| 0 |to| (- (ARRAYSIZE CHARACTER-SET-OFFSETS)
                                                             1)
                                      |when| (NEQ (ELT CHARACTER-SET-OFFSETS I)
                                                  0)
                                      |collect| I)))
                        (|for| I |from| 0 |to| (- (ARRAYSIZE CHARACTER-SET-OFFSETS)
                                                  1)
                           |when| (NEQ (ELT CHARACTER-SET-OFFSETS I)
                                       0)
                           |do| (CL:SETF (CL:GETF (CL:GETF (CL:GETF (CL:GETF *NOVAFONT-INFO* (MKATOM (U-CASE
                                                                                                     FONT-NAME
                                                                                                        )))
                                                                    (|fetch| (FONTDESCRIPTION SIZE) |of|
                                                                                                  NOVAFONT-DESCRIPTOR
                                                                                   ))
                                                                 FONT-FACE)
                                                       I)
                                     (LIST MINIMAL-FILE-NAME (+ (ELT CHARACTER-SET-OFFSETS I)
                                                                (ELT FONT-OFFSETS FONT-NUMBER))))))
                NUMBER-OF-FONTS))))

(LOAD-NOVAFONT-FILE
  (LAMBDA (|filename|)                                           ; Edited  9-Jul-87 16:23 by mdd
    (CL:WITH-OPEN-FILE (|stream| |filename| :INPUT)
           (CL:MULTIPLE-VALUE-BIND (|nfonts| |fontaddrs|)
               (READ-NOVAFONT-FILEHEADER |stream|)

             ;;  loop through the font nodes. fontAddrs are relative to wd 0 of file and have been converted to byte offsets when read in.
```

```
               (|for| |fontnumber| |from| 1 |to| |nfonts| |bind| |name| |fontdescriptor| |charsetaddrs| |fontpos| |font|
                                                               |font-face| (|rasterinfos| _ (ARRAY 256 'WORD 0 0))
                                                               (|fontprinterwidths| _ (ARRAY 256 'WORD 0 0))
                                                               (|fontspacingwidths| _ (ARRAY 256 'BYTE 0 0))
                  |do| (SETQ |fontpos| (ELT |fontaddrs| |fontnumber|))
                       ;; read the second level FontTreeNode (known as a font header, since it collects the character sets of the font)

                       ;; reads font header located at fontBlockPos into fontHeaderBuffer. Returns number of character sets in ncharSets, allocates
                       ;; an array to hold their word offsets from beginning of font block, and reads in those offsets.

                       (CL:MULTIPLE-VALUE-SETQ (|name| |fontdescriptor| |charsetaddrs|)
                              (READ-NOVAFONT-FONTHEADER |stream| |fontpos|))

                       ;; now read the third level char set nodes

                       (SETQ |font-face| (LIST (CL:ECASE (|fetch| (FONTDESCRIPTION WEIGHT) |of| |fontdescriptor|)
                                                   (0 'LIGHT)
                                                   (1 'MEDIUM)
                                                   (2 'BOLD))
                                               (CL:ECASE (|fetch| (FONTDESCRIPTION EMPHASIS) |of| |fontdescriptor|)
                                                   (0 'REGULAR)
                                                   (1 'ITALIC))
                                               'REGULAR))
                       (SETQ |name| (MKATOM (U-CASE |name|)))
                       (SETQ |font|
                        (|create| FONTDESCRIPTOR
                               FONTDEVICE _ 'DISPLAY
                               FONTFAMILY _ |name|
                               FONTSIZE _ (|fetch| (FONTDESCRIPTION SIZE) |of| |fontdescriptor|)
                               FONTFACE _ |font-face|
                               |\\SFAscent| _ 0
                               |\\SFDescent| _ 0
                               |\\SFHeight| _ 0
                               ROTATION _ 0
                               FONTDEVICESPEC _ (LIST |name| (|fetch| (FONTDESCRIPTION SIZE) |of| |fontdescriptor|)
                                                      |font-face| 0 'DISPLAY)))
                       (|for| \j |from| 0 |to| (- (ARRAYSIZE |charsetaddrs|)
                                                  1)
                          |bind| |csinfo| |charsetoffset|
                          |do| (SETQ |charsetoffset| (ELT |charsetaddrs| \j))
                              (|if| (NEQ |charsetoffset| 0)
                                  |then| ;; read in enough to get charSet # and bc,ec

                                        (SETQ |csinfo| (\\READNOVAFONTFILE.DISPLAY |stream| (PLUS |fontpos|
                                                                                                 |charsetoffset|)
                                                               NIL NIL NIL \j |rasterinfos| |fontprinterwidths|
                                                               |fontspacingwidths|))
                                        (|replace| |\\SFAscent| |of| |font| |with| (IMAX (|fetch| |\\SFAscent| |of| |font|)
                                                                                         (|fetch| CHARSETASCENT
                                                                                            |of| |csinfo|)))
                                        (|replace| |\\SFDescent| |of| |font| |with| (IMAX (|fetch| |\\SFDescent|
                                                                                             |of| |font|)
                                                                                          (|ffetch| CHARSETDESCENT
                                                                                             |of| |csinfo|)))
                                        (|replace| |\\SFHeight| |of| |font| |with| (IPLUS (|fetch| |\\SFAscent|
                                                                                             |of| |font|)
                                                                                          (|ffetch| |\\SFDescent|
                                                                                             |of| |font|)))
                                        (\\SETCHARSETINFO (|ffetch| FONTCHARSETVECTOR |of| |font|)
                                               \j |csinfo|)))
                       (|replace| (FONTDESCRIPTOR FONTAVGCHARWIDTH) |of| |font| |with| (\\AVGCHARWIDTH |font|))
                       (SETFONTDESCRIPTOR |name| (|fetch| (FONTDESCRIPTION SIZE) |of| |fontdescriptor|)
                              |font-face| 0 'DISPLAY |font|))))))

)


;;; the parts necessary for using with FONTCREATE

(DEFINEQ

(\\READNOVAFONTFILE.DISPLAY
  (LAMBDA (STREAM OFFSET FAMILY SIZE FACE CHARSET RASTERINFOS FONTPRINTERWIDTHS FONTSPACINGWIDTHS)
                                                                          ; Edited  9-Jul-87 16:23 by mdd
    (DECLARE (GLOBALVARS \\SYSPILOTBBT))
    (SETFILEPTR STREAM OFFSET)
    (LET ((CHARSETINFO (|create| CHARSETINFO
                              IMAGEWIDTHS _ (\\CREATECSINFOELEMENT)))
          (CHARSETHEADER (READ-BLOCK-OF-BYTES STREAM CHARSETBLOCKBYTESIZE))
          RASTEROFFSET RAWRASTERS)
         (|replace| (CHARSETINFO CHARSETASCENT) |of| CHARSETINFO |with| (|fetch| (CHARSETBLOCK ASCENT) |of| CHARSETHEADER
                                                                                ))
         (|replace| (CHARSETINFO CHARSETDESCENT) |of| CHARSETINFO |with| (|fetch| (CHARSETBLOCK DESCENT) |of|
                                                                                                          CHARSETHEADER
                                                                                 ))
         (SETFILEPTR STREAM (+ OFFSET CHARSETBLOCKBYTESIZE))
         ;; read the raster information, spacing for the printer (not used here) and spacing for the display as they are stored in the novafont file

         (OR (AND RASTERINFOS (ARRAYP RASTERINFOS)
          (EQ (ARRAYSIZE RASTERINFOS)
              256))
     (SETQ RASTERINFOS (ARRAY 256 'WORD 0 0)))
 (AIN RASTERINFOS 0 (ARRAYSIZE RASTERINFOS)
      STREAM)
 (OR (AND FONTPRINTERWIDTHS (ARRAYP FONTPRINTERWIDTHS)
          (EQ (ARRAYSIZE FONTPRINTERWIDTHS)
              256))
     (SETQ FONTPRINTERWIDTHS (ARRAY 256 'WORD 0 0)))
 (AIN FONTPRINTERWIDTHS 0 (ARRAYSIZE FONTPRINTERWIDTHS)
      STREAM)
 (OR (AND FONTSPACINGWIDTHS (ARRAYP FONTSPACINGWIDTHS)
          (EQ (ARRAYSIZE FONTSPACINGWIDTHS)
              256))
     (SETQ FONTSPACINGWIDTHS (ARRAY 256 'BYTE 0 0)))
 (AIN FONTSPACINGWIDTHS 0 (ARRAYSIZE FONTSPACINGWIDTHS)
      STREAM)
 ;; position to the start of the rasters, after the rasterinfo (256 words), printer width (256 words) and spacing width (256 bytes) arrays (this
 ;; should be a noop if there's no padding)
 (SETFILEPTR STREAM (+ OFFSET CHARSETBLOCKBYTESIZE (+ 256 (CL:* 256 BYTESPERWORD 2))))
 ;; the rasters should be all the remaining storage in the character set block.
 (SETQ RAWRASTERS (READ-BLOCK-OF-BYTES STREAM (- (CL:* BYTESPERWORD (|fetch| (CHARSETBLOCK SIZE)
                                                                          |of| CHARSETHEADER))
                                                 (+ (+ 256 (CL:* 256 BYTESPERWORD 2))
                                                    CHARSETBLOCKBYTESIZE))))
 ;; process the novafont format information to that required for a regular font descriptor.  We must compute the actual image width based on
 ;; the kerning information (bits 15 and 16) passed in the raster infos.  The "slug" is always the first character in a novafont.
 (|for| CHARACTER |from| 1 |to| 255 |bind| (SLUGRASTERINFO _ (ELT RASTERINFOS 0))
    |first| ;; we set up the slug first, then process all the other characters
          (\\FSETOFFSET (|fetch| (CHARSETINFO OFFSETS) |of| CHARSETINFO)
                 0 0)
          (\\FSETWIDTH (|fetch| (CHARSETINFO WIDTHS) |of| CHARSETINFO)
                 0
                 (ELT FONTSPACINGWIDTHS 0))
          (\\FSETIMAGEWIDTH (|fetch| (CHARSETINFO IMAGEWIDTHS) |of| CHARSETINFO)
                 0
                 (+ (ELT FONTSPACINGWIDTHS 0)
                    (LOGAND (RSH (ELT RASTERINFOS 0)
                                 14)
                           1)
                    (RSH (ELT RASTERINFOS 0)
                         15)))
          (SETQ RASTEROFFSET (\\FGETIMAGEWIDTH (|fetch| (CHARSETINFO IMAGEWIDTHS) |of| CHARSETINFO)
                                    0))
    |do| (\\FSETWIDTH (|fetch| (CHARSETINFO WIDTHS) |of| CHARSETINFO)
                CHARACTER
                (ELT FONTSPACINGWIDTHS CHARACTER))
         (\\FSETIMAGEWIDTH (|fetch| (CHARSETINFO IMAGEWIDTHS) |of| CHARSETINFO)
                CHARACTER
                (+ (ELT FONTSPACINGWIDTHS CHARACTER)
                   (LOGAND (RSH (ELT RASTERINFOS CHARACTER)
                                14)
                          1)
                   (RSH (ELT RASTERINFOS CHARACTER)
                        15)))
         (|if| (NOT (EQUAL (ELT RASTERINFOS CHARACTER)
                           SLUGRASTERINFO))
             |then| (\\FSETOFFSET (|fetch| (CHARSETINFO OFFSETS) |of| CHARSETINFO)
                           CHARACTER RASTEROFFSET)
                   (SETQ RASTEROFFSET (+ RASTEROFFSET (\\FGETIMAGEWIDTH (|fetch| (CHARSETINFO IMAGEWIDTHS)
                                                                           |of| CHARSETINFO)
                                                             CHARACTER)))))
 ;; we used the rasteroffset calculated above to determine the width of the character bitmap that we must create  -- otherwise this would be
 ;; folded into the previous loop.  We also allocate some extra bits in case we have to fake the space character
 (|replace| (CHARSETINFO CHARSETBITMAP) |of| CHARSETINFO |with| (BITMAPCREATE
                                                                  (+ RASTEROFFSET (\\FGETIMAGEWIDTH
                                                                                   (|fetch| (CHARSETINFO
                                                                                                 IMAGEWIDTHS)
                                                                                      |of| CHARSETINFO)
                                                                                   0))
                                                                  (|fetch| (CHARSETBLOCK HEIGHT) |of|
                                                                                                  CHARSETHEADER
                                                                         )))
 ;; set up the slug first to speed up the check in the next loop
 (\\TEXTBLT \\SYSPILOTBBT (\\ADDBASE RAWRASTERS (LOGAND (ELT RASTERINFOS 0)
                                                       16383))
        (\\FGETIMAGEWIDTH (|fetch| (CHARSETINFO IMAGEWIDTHS) |of| CHARSETINFO)
               0)
        (|fetch| (CHARSETBLOCK HEIGHT) |of| CHARSETHEADER)
        (|fetch| (CHARSETINFO CHARSETBITMAP) |of| CHARSETINFO)
        (\\FGETOFFSET (|fetch| (CHARSETINFO OFFSETS) |of| CHARSETINFO)
                    0))
        ;; extract the bitmaps for all the characters in the font.
        (|for| CHARACTER |from| 1 |to| 255 |bind| (HEIGHT _ (|fetch| (CHARSETBLOCK HEIGHT) |of| CHARSETHEADER))
                                           (SLUGOFFSET _ (\\FGETOFFSET (|fetch| (CHARSETINFO OFFSETS)
                                                                           |of| CHARSETINFO)
                                                                  0))
           |when| (NOT (EQL SLUGOFFSET (\\FGETOFFSET (|fetch| (CHARSETINFO OFFSETS) |of| CHARSETINFO)
                                              CHARACTER)))
           |do| (\\TEXTBLT \\SYSPILOTBBT (\\ADDBASE RAWRASTERS (LOGAND (ELT RASTERINFOS CHARACTER)
                                                                      16383))
                       (\\FGETIMAGEWIDTH (|fetch| (CHARSETINFO IMAGEWIDTHS) |of| CHARSETINFO)
                              CHARACTER)
                       HEIGHT
                       (|fetch| (CHARSETINFO CHARSETBITMAP) |of| CHARSETINFO)
                       (\\FGETOFFSET (|fetch| (CHARSETINFO OFFSETS) |of| CHARSETINFO)
                              CHARACTER)))
        ;; if this is character set 0, and the space is a slug then we've got to fix up a space at the end of the bitmaps.  For now we'll make it a 1 ex
        ;; space.
        (|if| (AND (EQ CHARSET 0)
                   (EQL (\\FGETOFFSET (|fetch| (CHARSETINFO OFFSETS) |of| CHARSETINFO)
                               0)
                        (\\FGETOFFSET (|fetch| (CHARSETINFO OFFSETS) |of| CHARSETINFO)
                               (CHARCODE SP))))
            |then| (\\FSETOFFSET (|fetch| (CHARSETINFO OFFSETS) |of| CHARSETINFO)
                          (CHARCODE SP)
                          RASTEROFFSET)
                   (\\FSETWIDTH (|fetch| (CHARSETINFO WIDTHS) |of| CHARSETINFO)
                          (CHARCODE SP)
                          (\\FGETWIDTH (|fetch| (CHARSETINFO WIDTHS) |of| CHARSETINFO)
                                 (CHARCODE \x)))
                   (\\FSETIMAGEWIDTH (|fetch| (CHARSETINFO IMAGEWIDTHS) |of| CHARSETINFO)
                          (CHARCODE SP)
                          (\\FGETWIDTH (|fetch| (CHARSETINFO WIDTHS) |of| CHARSETINFO)
                                 (CHARCODE \x))))
        ;; finally, return the newly created charsetinfo
        CHARSETINFO)))

(\\READNOVAFONTFILE.IP
  (LAMBDA (STREAM OFFSET CHARSET CHARSETINFO)                     ; Edited  9-Feb-89 13:51 by Briggs
    (SETFILEPTR STREAM OFFSET)
    (LET ((CHARSETHEADER (READ-BLOCK-OF-BYTES STREAM CHARSETBLOCKBYTESIZE))
          FONTPRINTERWIDTHS MINUS-FBBOY RASTERINFOS)
        ;; Descent from -FBBOY -- note that -FBBOY *could* be negative (lose!)
        (|if| (ILESSP (SETQ MINUS-FBBOY (IMINUS (SIGNED (|fetch| (CHARSETBLOCK FBBOY) |of| CHARSETHEADER)
                                                        BITSPERWORD)))
                     0)
            |then| (|replace| (CHARSETINFO CHARSETDESCENT) |of| CHARSETINFO |with| 0)
                   (|replace| (CHARSETINFO CHARSETASCENT) |of| CHARSETINFO |with| (|fetch| (CHARSETBLOCK FBBDY)
                                                                                      |of| CHARSETHEADER))
          |else| (|replace| (CHARSETINFO CHARSETDESCENT) |of| CHARSETINFO |with| MINUS-FBBOY)
                 (|replace| (CHARSETINFO CHARSETASCENT) |of| CHARSETINFO |with| (IDIFFERENCE (|fetch| (CHARSETBLOCK
                                                                                                       FBBDY)
                                                                                                |of| CHARSETHEADER)
                                                                                         MINUS-FBBOY)))
        (SETFILEPTR STREAM (+ OFFSET CHARSETBLOCKBYTESIZE))
        ;; read the raster information, spacing for the printer (not used here) and spacing for the display as they are stored in the novafont file
        (SETQ RASTERINFOS (ARRAY 256 'WORD 0 0))
        (AIN RASTERINFOS 0 (ARRAYSIZE RASTERINFOS)
             STREAM)
        (SETQ FONTPRINTERWIDTHS (ARRAY 256 'WORD 0 0))
        (AIN FONTPRINTERWIDTHS 0 (ARRAYSIZE FONTPRINTERWIDTHS)
             STREAM)
        (|for| CHARACTER |from| 0 |to| 255 |do| (\\FSETWIDTH (|fetch| (CHARSETINFO WIDTHS) |of| CHARSETINFO)
                                                 CHARACTER
                                                 (ELT FONTPRINTERWIDTHS CHARACTER)))
        CHARSETINFO)))
)

;;; modified versions of functions from the default font handling system

(DEFINEQ

(\\READDISPLAYFONTFILE.NOVA
  (LAMBDA (FAMILY SIZE FACE ROTATION DEVICE CHARSET)               ; Edited  6-Dec-88 14:59 by Briggs
    (OR (LET ((CS (CL:GETF (CL:GETF (CL:GETF (CL:GETF *NOVAFONT-INFO* (U-CASE FAMILY))
                                             SIZE)
                                    (+ (CL:ECASE (CADR FACE)
                                              (REGULAR 0)
                                              (ITALIC 1))
                                    (CL:* 2 (CL:ECASE (CAR FACE)
                                                (LIGHT 0)
                                                (MEDIUM 1)
                                                (BOLD 2)))))
                            (OR CHARSET (SETQ CHARSET 0)))))
            (CL:WHEN CS
                (CL:WITH-OPEN-FILE (STREAM (CL:MERGE-PATHNAMES (CAR CS)
                                                   *NOVAFONT-PATHNAME-DEFAULTS*)
                                           :DIRECTION :INPUT)
                           (\\READNOVAFONTFILE.DISPLAY STREAM (CADR CS)
                                 FAMILY SIZE FACE CHARSET))))
        (\\NO-NOVA-READDISPLAYFONTFILE FAMILY SIZE FACE ROTATION DEVICE CHARSET))))

(\\CREATECHARSET.IP.NOVA
  (LAMBDA (FAMILY PSIZE FACE ROTATION DEVICE CHARSET FONTDESC NOSLUG?)
                                                                  ; Edited  6-Dec-88 15:08 by Briggs
    (OR (LET ((CS (CL:GETF (CL:GETF (CL:GETF (CL:GETF *NOVAFONT-INFO* (U-CASE FAMILY))
                                             PSIZE)
                                 (+ (CL:ECASE (CADR FACE)
                                           (REGULAR 0)
                                           (ITALIC 1))
                                    (CL:* 2 (CL:ECASE (CAR FACE)
                                                (LIGHT 0)
                                                (MEDIUM 1)
                                                (BOLD 2)))))
                            (OR CHARSET (SETQ CHARSET 0)))))
            (AND CS (CL:WITH-OPEN-FILE (STREAM (CL:MERGE-PATHNAMES (CAR CS)
                                                   *NOVAFONT-PATHNAME-DEFAULTS*)
                                           :DIRECTION :INPUT)
                           (\\READNOVAFONTFILE.IP STREAM (CADR CS)
                                  CHARSET
                                  (|create| CHARSETINFO)))))
        (\\NO-NOVA-CREATECHARSET.IP FAMILY PSIZE FACE ROTATION DEVICE CHARSET FONTDESC NOSLUG?))))

)
```

;;; the parts for general hacking of the NOVAFONT files

```
(DEFINEQ

(DESCRIBE-FONT
  (LAMBDA (NAME FONTDESCRIPTOR CHARSETHEADER FONTSPACINGWIDTHS FONTPRINTERWIDTHS FONTRASTER)
                                                                  (* |briggs| "11-Nov-86 22:58")
    (|if| (AND (BOUNDP 'LASTFONTDESCRIPTOR)
               (NEQ FONTDESCRIPTOR LASTFONTDESCRIPTOR))
        |then| (SETQ LASTFONTDESCRIPTOR FONTDESCRIPTOR)
               (PRINTOUT T T NAME ":" \, (|fetch| (FONTDESCRIPTION SIZE) |of| FONTDESCRIPTOR)
                      "pt. "
                      (CASE (|fetch| (FONTDESCRIPTION WEIGHT) |of| FONTDESCRIPTOR)
                          (0 "light ")
                          (1 "medium ")
                          (2 "bold ")
                          (OTHERWISE "unknown "))
                      (CASE (|fetch| (FONTDESCRIPTION EMPHASIS) |of| FONTDESCRIPTOR)
                          (0 "regular")
                          (1 "italic")
                          (OTHERWISE "unknown"))
                      " character sets: "))
    (PRINTOUT T (|fetch| (CHARSETBLOCK CHARSETNUM) |of| CHARSETHEADER)
           \,)))

(SELECT-FONT
  (LAMBDA (NAME FONTDESCRIPTOR CHARSETHEADER FONTSPACINGWIDTHS FONTPRINTERWIDTHS FONTRASTER)
                                                                  (* BRIGGS " 6-Nov-86 23:14")
    (|if| (EQL 12 (|fetch| (FONTDESCRIPTION SIZE) |of| FONTDESCRIPTOR))
        |then| (CL:ASSERT NIL))))

(ENUMERATE-FONTS
  (LAMBDA (STREAM PROC READ-RASTERS-P)                            (* |briggs| "11-Nov-86 23:03")

    ;; "assumes stream is open to a viewpoint font file with read access. Calls PROC for each font in the file.  Used in listing the contents of a
    ;; ViewPoint screenfont file. Returns error TRUE of an error of any kind occurs in working through the file (I/O errors, format errors, etc.)"

    (SETFILEPTR STREAM 0)
    (OR (VIEWPOINT-FONT-FILE-P STREAM)
        (CL:ERROR "not a font file"))
    (LET (FONTADDRS NFONTS)
         (CL:MULTIPLE-VALUE-SETQ (NFONTS FONTADDRS)
                (READ-NOVAFONT-FILEHEADER STREAM))

         ;;  loop through the font nodes. fontAddrs are relative to wd 0 of file and have been converted to byte offsets when read in.

         (|for| FONTNUMBER |from| 1 |to| NFONTS |bind| NAME FONTDESCRIPTOR CHARSETADDRS FONTPOS
            |do| (SETQ FONTPOS (ELT FONTADDRS FONTNUMBER))

                  ;; read the second level FontTreeNode (known as a font header, since it collects the character sets of the font)
```

```
                    ;; reads font header located at fontBlockPos into fontHeaderBuffer. Returns number of character sets in ncharSets, allocates an
                    ;; array to hold their word offsets from beginning of font block, and reads in those offsets.

                    (CL:MULTIPLE-VALUE-SETQ (NAME FONTDESCRIPTOR CHARSETADDRS)
                           (READ-NOVAFONT-FONTHEADER STREAM FONTPOS))

                    ;; now read the third level char set nodes

                    (|for| J |from| 0 |to| (- (ARRAYSIZE CHARSETADDRS)
                                              1)
                       |bind| CHARSETHEADER FONTSPACINGWIDTHS FONTPRINTERWIDTHS FONTRASTER RASTEROFFSETS CHARSETOFFSET
                       |do| (SETQ CHARSETOFFSET (ELT CHARSETADDRS J))
                            (|if| (NEQ CHARSETOFFSET 0)
                                |then| ;; read in enough to get charSet # and bc,ec

                                      (CL:MULTIPLE-VALUE-SETQ (CHARSETHEADER FONTSPACINGWIDTHS FONTPRINTERWIDTHS
                                                                  FONTRASTER RASTEROFFSETS)
                                             (READ-NOVAFONT-CHARACTERSET STREAM (+ FONTPOS CHARSETOFFSET)
                                                    READ-RASTERS-P))
                                      (|if| PROC
                                          |then| (APPLY* PROC NAME FONTDESCRIPTOR CHARSETHEADER FONTSPACINGWIDTHS
                                                        FONTPRINTERWIDTHS FONTRASTER RASTEROFFSETS))

                                      ;; Get pointsize (? bits) pitch (either fixed or variable) weight (light medium heavy other) posture (roman
                                      ;; italic) and character set - charsetheader includes height descent

                                      ;; SETQ STARTOFRASTERS (+ CHARSTART 1280 (* 2 FONTSEGMENTHEADERSIZE)) (LET ((FONT
                                      ;; (create FONTDESCRIPTOR FONTDEVICE _ (QUOTE DISPLAY) FONTFAMILY _ (MKATOM (U-CASE
                                      ;; NAME)) FONTSIZE _ SIZE FONTFACE _ FACE \SFAscent _ 0 \SFDescent _ 0 \SFHeight _ 0 ROTATION
                                      ;; _ ROTATION)) (CSI (CREATE CHARSETINFO)))) (FILLINFONTOBJECT FONT FONTINFO
                                      ;; CHARSETHEADER STARMODE) (FUNCALL PROC FONT BC EC FONTPOS CHARSETPOS)

                                      ))))))

(VIEWPOINT-FONT-FILE-P
  (LAMBDA (STREAM)                                                (* |briggs| "11-Nov-86 22:44")
    "assumes stream is open to a file with read access.  returns TRUE iff the file is a ViewPoint screen font
    file"
    (LET (FILEHEADER FIRSTFONTADDR FIRSTCHARSETADDR)

         ;; read first 12 words of file & check for pattern in first 3 words

         (SETQ FILEHEADER (READ-BLOCK-OF-BYTES STREAM FONTTREENODEBLOCKBYTESIZE))
         (|if| (|with| FONTTREENODEBLOCK FILEHEADER (AND (EQ ID 0)
                                                         (EQ TYPE 65535)
                                                         (EQL (GETEOFPTR STREAM)
                                                              (+ FONTTREENODEBLOCKBYTESIZE
                                                                 (CL:* BYTESPERWORD (+ (CL:* 2 NCHILDREN)
                                                                                       SIZEFILLER1 SIZECHILDREN)))))
                 )
             |then| ;; "at this point, we could have either a ViewPoint or Star font. First follow offsets to the first char set of the first font"

                   (SETQ FIRSTFONTADDR (PROGN (SETFILEPTR STREAM FONTTREENODEBLOCKBYTESIZE)
                                              (READSWAPPEDFIXP STREAM)))
                   (SETQ FIRSTCHARSETADDR (PROGN (SETFILEPTR STREAM (+ FIRSTFONTADDR FONTTREENODEBLOCKBYTESIZE))
                                                 (READSWAPPEDFIXP STREAM)))

                   ;; "Viewpoint files contain (here) a 256 word array of kern + offset, then a 256 word array of printer widths.  Offset words are
                   ;; zero if no character, otherwise are monotonically increasing, since bitmaps are inserted in character code order.  Printer
                   ;; widths are initialized to a default constant value for characters with no bitmap.  So if first 256 words (masked to low order 14
                   ;; bits and not counting 0 values) are monotonically increasing, we have a ViewPoint file. Legal ViewPoint arrays have
                   ;; monotonically increasing non-zero elements, whereas star arrays will be mixed in with printer widths and will not be
                   ;; monotonically increasing."

;;; Punt for now

                   T))))

)

(RPAQQ *WARN-ON-KERNING* NIL)

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS *WARN-ON-KERNING*)
)


;;; where the NovaFont files are likely to be


(DEFGLOBALVAR *NOVAFONT-PATHNAME-DEFAULTS* (PATHNAME (AND (BOUNDP 'DISPLAYFONTDIRECTORIES)
                                                          (LISTP DISPLAYFONTDIRECTORIES)
                                                          (CAR DISPLAYFONTDIRECTORIES))))


;;; things for dealing with the structure of  what we read

(DECLARE\: EVAL@COMPILE
(PUTPROPS READSWAPPEDFIXP DMACRO (OPENLAMBDA (STREAM)
                                       (+ (LOGOR (LLSH (BIN STREAM)
                                                       8)
                                                 (BIN STREAM))
                                          (CL:ASH (LOGOR (LLSH (BIN STREAM)
                                                               8)
                                                         (BIN STREAM))
                                                 16))))
)

(DEFINEQ

(READ-BLOCK-OF-BYTES
  (LAMBDA (STREAM NUMBER-OF-BYTES)                                  (* |briggs| " 9-Nov-86 23:16")
    (LET ((RESULT (\\ALLOCBLOCK (FOLDHI NUMBER-OF-BYTES BYTESPERCELL)
                        UNBOXEDBLOCK.GCT)))
         (\\BINS STREAM RESULT 0 NUMBER-OF-BYTES)

         ;; (|for| byteindex |from| 0 |to| (- number-of-bytes 1) |do| (\\putbasebyte result byteindex (bin stream)))

         RESULT)))

(READ-NOVAFONT-CHARACTERSET
  (LAMBDA (STREAM OFFSETTOCHARSET READ-RASTERS-P)                   (* |briggs| "11-Nov-86 23:18")
    (DECLARE (GLOBALVARS *WARN-ON-KERNING* \\SYSPILOTBBT))
    (SETFILEPTR STREAM OFFSETTOCHARSET)
    (LET ((CHARSETHEADER (READ-BLOCK-OF-BYTES STREAM CHARSETBLOCKBYTESIZE))
          RASTERINFOS FONTPRINTERWIDTHS FONTSPACINGWIDTHS FONTRASTER RAWRASTER OFFSETSBLOCK)

         ;; The header portion of a CharacterSet contains information such as character set number, height (which is constant for all characters), max
         ;; width, ascender & descender, and font bounding box.

         ;; reads the raster infos array of the character set located at charsetPos and determines bc:ec

         (SETFILEPTR STREAM (+ OFFSETTOCHARSET CHARSETBLOCKBYTESIZE))

         ;; The rasterinfos field is basically an array 14 bit word offsets in the fontrasters array of where each bitmap starts.  The offsets are relative to
         ;; the start of the fontRasters field. fontprinterwidths & fontspacingwidths are initialized to certain default values, and fontrasters starts out
         ;; with a 'missing character' bitmap - a black rectangle with a one pixel white outline at each side, sitting on the baseline, and running up to
         ;; the ascend of the font, such that the whole thing is exactly font height by max width in size. Bitmaps include sufficient white space so that
         ;; they can be placed contiguously (or in the case of kerned ones, overlapping previous by one pixel) without additional adjustments for
         ;; spacing.  I.e. they are in the right format for TextBlt.  Padding is added so that all FontTreeNodes and CharacterSets begin on four word
         ;; boundries, for some reason that is lost in antiquity.

         (SETQ RASTERINFOS (ARRAY 256 'WORD 0 0))
         (AIN RASTERINFOS 0 (ARRAYSIZE RASTERINFOS)
              STREAM)
         (SETQ FONTPRINTERWIDTHS (ARRAY 256 'WORD 0 0))
         (AIN FONTPRINTERWIDTHS 0 (ARRAYSIZE FONTPRINTERWIDTHS)
              STREAM)
         (SETQ FONTSPACINGWIDTHS (ARRAY 256 'BYTE 0 0))
         (AIN FONTSPACINGWIDTHS 0 (ARRAYSIZE FONTSPACINGWIDTHS)
              STREAM)
         (|if| *WARN-ON-KERNING*
             |then| (LET ((KERNS (|for| I |from| 0 |to| 255 |when| (> (ELT RASTERINFOS I)
                                                                      16383)
                                    |collect| I)))
                         (|if| KERNS
                             |then| (CL:WARN "Kerning on characters~{ ~S~}." KERNS))))
         (|for| I |from| 0 |to| 255 |do| (|if| (>= (ELT FONTSPACINGWIDTHS I)
                                                 (|fetch| (CHARSETBLOCK MAXWIDTH) |of| CHARSETHEADER))
                                         |then| (|replace| (CHARSETBLOCK MAXWIDTH) |of| CHARSETHEADER
                                                   |with| (ELT FONTSPACINGWIDTHS I))))
         (|if| READ-RASTERS-P
             |then| (SETQ FONTRASTER (BITMAPCREATE (CL:* (|fetch| (CHARSETBLOCK MAXWIDTH) |of| CHARSETHEADER)
                                                         256)
                                            (|fetch| (CHARSETBLOCK HEIGHT) |of| CHARSETHEADER)))
                    (SETFILEPTR STREAM (+ OFFSETTOCHARSET 1280 CHARSETBLOCKBYTESIZE))
                    (SETQ RAWRASTER (READ-BLOCK-OF-BYTES STREAM (- (CL:* BYTESPERWORD (|fetch| (CHARSETBLOCK SIZE)
                                                                                             |of| CHARSETHEADER))
                                                                   (+ 1280 CHARSETBLOCKBYTESIZE))))
                    (SETQ OFFSETSBLOCK (\\CREATECSINFOELEMENT))
                    (|for| I |from| 0 |to| 254 |bind| (OFFSET _ 0)
                                                   SPACINGWIDTH
                       |do| (|if| (OR (EQL I 0)
                                      (NOT (EQL (LOGAND (ELT RASTERINFOS I)
                                                        16383)
                                                (LOGAND (ELT RASTERINFOS 0)
                                                        16383))))
                                |then| (SETQ SPACINGWIDTH (+ (ELT FONTSPACINGWIDTHS I)
                                                             (LOGAND (RSH (ELT RASTERINFOS I)
                                                                          14)
                                                                     1)
                                                             (RSH (ELT RASTERINFOS I)
                                                                  15)))
                                      (CL:ASSERT (< (+ OFFSET SPACINGWIDTH)
                                                    (BITMAPWIDTH FONTRASTER))
                                             NIL "Attempted to blt beyond end of bitmap")
                                      (\\TEXTBLT \\SYSPILOTBBT (\\ADDBASE RAWRASTER (LOGAND (ELT RASTERINFOS I)
                                                                                        16383))
                                             SPACINGWIDTH
                                             (|fetch| (CHARSETBLOCK HEIGHT) |of| CHARSETHEADER)
                                             FONTRASTER OFFSET)
                                       (\\FSETOFFSET OFFSETSBLOCK I OFFSET)
                                       (SETQ OFFSET (+ OFFSET (ELT FONTSPACINGWIDTHS I)))
                              |else| (\\FSETOFFSET OFFSETSBLOCK I 0))))
         (CL:VALUES CHARSETHEADER FONTSPACINGWIDTHS FONTPRINTERWIDTHS (|if| READ-RASTERS-P
                                                                            |then| FONTRASTER
                                                                          |else| NIL)
                (|if| READ-RASTERS-P
                    |then| OFFSETSBLOCK
                  |else| NIL)))))
```

(**READ-NOVAFONT-FILEHEADER**

```
  (CL:LAMBDA (STREAM)                                         ; Edited 24-Nov-86 22:41 by BRIGGS
```

;; reads file header of an open viewpoint font file to determine number of fonts, allocates an array to hold their offsets from beginning of file,
;; and reads in those font offsets. While reading it converts from WORD offsets to BYTE offsets. This function also verifies that what it is
;; passed is a plausible NOVAFONT format file.

```
         (LET (NFONTS FILEHEADERBUFFER FONTADDRS FILESIZE)
              (SETQ FILESIZE (GETEOFPTR STREAM))

              ;; verify that there are enough bytes to be a plausible font file

              (CL:ASSERT (>= FILESIZE FONTTREENODEBLOCKBYTESIZE)
                     NIL "~(~A~) is not a NOVAFONT format font file." (FULLNAME STREAM))
              (SETFILEPTR STREAM 0)
              (SETQ FILEHEADERBUFFER (READ-BLOCK-OF-BYTES STREAM FONTTREENODEBLOCKBYTESIZE))

              ;; check that what we read is a plausible font file header

              (CL:ASSERT (|with| FONTTREENODEBLOCK FILEHEADERBUFFER
                                (AND (EQ ID 0)
                                     (EQ TYPE 65535)
                                     (EQL FILESIZE (+ FONTTREENODEBLOCKBYTESIZE (CL:* BYTESPERWORD
                                                                                  (+ (CL:* 2 NCHILDREN)
                                                                                     SIZEFILLER1 SIZECHILDREN
                                                                                     ))))))
                     NIL "~(~A~) is not a NOVAFONT format font file." (FULLNAME STREAM))
              (SETQ NFONTS (|fetch| (FONTTREENODEBLOCK NCHILDREN) |of| FILEHEADERBUFFER))
              (SETQ FONTADDRS (ARRAY NFONTS 'FIXP 0 1))
              (|for| I |from| 1 |to| NFONTS |do| (SETA FONTADDRS I (CL:* (READSWAPPEDFIXP STREAM)
                                                                        BYTESPERWORD)))
              (CL:VALUES NFONTS FONTADDRS))))
```

(**READ-NOVAFONT-FONTHEADER**

```
  (CL:LAMBDA (STREAM FONTPOS)                                 ; Edited  6-Dec-88 10:47 by Briggs
```

;; reads font header located at fontPos into fontHeaderBuffer. Returns number of character sets in ncharSets, allocates an array to hold their
;; word offsets from beginning of font block, and reads in those offsets.

;; reads font header located at fontBlockPos into fontHeaderBuffer. Also returns the name of the font

```
         (LET (FONTHEADER MAXCHARSETNUMBER CHARSETADDRS FONTDESCRIPTOR NAME)
              (SETFILEPTR STREAM FONTPOS)
              (SETQ FONTHEADER (READ-BLOCK-OF-BYTES STREAM FONTTREENODEBLOCKBYTESIZE))
              (SETQ MAXCHARSETNUMBER (|fetch| (FONTTREENODEBLOCK NCHILDREN) |of| FONTHEADER))
              (SETQ CHARSETADDRS (ARRAY MAXCHARSETNUMBER 'FIXP 0 0))
              (SETFILEPTR STREAM (+ FONTPOS FONTTREENODEBLOCKBYTESIZE))
              (|for| I |from| 0 |to| (- MAXCHARSETNUMBER 1) |do| ;; contains swapped count of 16-bit words, turn into count of number of
                                                                  ;; bytes

                                                                  (SETA CHARSETADDRS I (CL:* 2 (READSWAPPEDFIXP STREAM)
                                                                                              )))

              ;; skip to the nodeInfo field, which is of type FontInfo

              (SETFILEPTR STREAM (+ FONTPOS FONTTREENODEBLOCKBYTESIZE (CL:* 2 (+ (CL:* 2 MAXCHARSETNUMBER)
                                                                                 (|fetch| (FONTTREENODEBLOCK
                                                                                             SIZEFILLER1)
                                                                                    |of| FONTHEADER)))))
              (CL:ASSERT (EQL (BIN16 STREAM)
                              3325))
              (SETQ FONTDESCRIPTOR (READ-BLOCK-OF-BYTES STREAM FONTDESCRIPTIONBYTESIZE))
              (BIN STREAM)                                    ; a piece of junk
              (SETQ NAME (LET* ((SIZE (BIN STREAM))
                                (STRING (ALLOCSTRING SIZE)))
                               (|for| I |from| 1 |to| SIZE |do| (RPLCHARCODE STRING I (BIN STREAM)))
                               STRING))
              (|if| (ZEROP (NCHARS NAME))
                  |then| ;; ugh! no name, try to guess from the family number

                         (SETQ NAME (OR (CDR (ASSOC (|fetch| (FONTDESCRIPTION FAMILY) |of| FONTDESCRIPTOR)
                                                    \\NOVAFONTFAMILYNAMES))
                                        (CONCAT "UnknownFont-" (|fetch| (FONTDESCRIPTION FAMILY) |of| FONTDESCRIPTOR)
                                               )))))
              (CL:VALUES NAME FONTDESCRIPTOR CHARSETADDRS))))
```

(**\\TEXTBLT**

```
  (LAMBDA (PILOTBBT |SourceHunk| |SourceWidth| |SourceHeight| |DestinationBitMap| |DestinationLeft|)
                                                                    ; Edited 12-Mar-87 18:00 by Briggs
    (\\DTEST PILOTBBT 'PILOTBBT)
    (\\DTEST |DestinationBitMap| 'BITMAP)
    (UNINTERRUPTABLY
        (|freplace| (PILOTBBT PBTFLAGS) |of| PILOTBBT |with| 0)
        (|freplace| (PILOTBBT PBTDESTBPL) |of| PILOTBBT |with| (UNFOLD (|ffetch| (BITMAP BITMAPRASTERWIDTH) |of|
                                                                                              |DestinationBitMap|
                                                                             )
                                                                    BITSPERWORD))
        (|freplace| (PILOTBBT PBTDESTBIT) |of| PILOTBBT |with| |DestinationLeft|)
        (|freplace| (PILOTBBT PBTUSEGRAY) |of| PILOTBBT |with| NIL)     ; the raster width of the source
        (|freplace| (PILOTBBT PBTSOURCEBPL) |of| PILOTBBT |with| |SourceWidth|)
        (|freplace| (PILOTBBT PBTWIDTH) |of| PILOTBBT |with| |SourceWidth|)
        (|freplace| (PILOTBBT PBTHEIGHT) |of| PILOTBBT |with| |SourceHeight|)
        (|freplace| (PILOTBBT PBTSOURCEBIT) |of| PILOTBBT |with| 0)
        (|freplace| (PILOTBBT PBTDISJOINT) |of| PILOTBBT |with| T)
        (|freplace| (PILOTBBT PBTSOURCE) |of| PILOTBBT |with| |SourceHunk|)
        (|freplace| (PILOTBBT PBTDEST) |of| PILOTBBT |with| (|ffetch| (BITMAP BITMAPBASE) |of| |DestinationBitMap|))
        (\\SETPBTFUNCTION PILOTBBT 'INPUT 'PAINT)
        (\\PILOTBITBLT PILOTBBT 0))))

)
```

;;; the datastructures that we use and their sizes

```
(DECLARE\: EVAL@COMPILE DONTCOPY

(DECLARE\: EVAL@COMPILE

(BLOCKRECORD FONTTREENODEBLOCK ((ID FIXP)
                                (TYPE WORD)
                                (NCHILDREN WORD)
                                (SIZEFILLER1 SWAPPEDFIXP)
                                (SIZENODEINFO SWAPPEDFIXP)
                                (SIZEFILLER2 SWAPPEDFIXP)
                                (SIZECHILDREN SWAPPEDFIXP)
                                (DUMMY-LAST-FIELD-DONT-REFERENCE-THIS WORD)))

(BLOCKRECORD CHARSETBLOCK ((SIZE SWAPPEDFIXP)
                           (VERSION WORD)
                           (CHARSETNUM WORD)
                           (MAXWIDTH WORD)
                           (HEIGHT WORD)
                           (ASCENT WORD)
                           (DESCENT WORD)
                           (FBBOX WORD)
                           (FBBOY WORD)
                           (FBBDX WORD)
                           (FBBDY WORD)
                           (DUMMY-LAST-FIELD-DONT-REFERENCE-THIS WORD)))

(BLOCKRECORD FONTDESCRIPTION ((SIZE BITS 8)
                              (WEIGHT BITS 2)
                              (EMPHASIS BITS 1)
                              (UNDERLINE BITS 1)
                              (STRIKEOUT BITS 1)
                              (PLACEMENT BITS 3)
                              (MBZ1 BITS 1)
                              (PITCH BITS 1)
                              (ORNATENESS BITS 1)
                              (FAMILY BITS 12)
                              (MBZ2 BITS 1)
                              (MBZ3 BITS 1)
                              (OFFSET BITS 14)
                              (MBZ4 BITS 1)
                              (MBZ5 BITS 1)
                              (DOUBLEUNDERLINE BITS 1)
                              (UNUSED BITS 14)
                              (DUMMY-LAST-FIELD-DONT-REFERENCE-THIS WORD)))
)

(DECLARE\: EVAL@COMPILE

(RPAQ FONTTREENODEBLOCKBYTESIZE (CONSTANT (ITIMES BYTESPERWORD (INDEXF (FETCH (FONTTREENODEBLOCK
                                                                              DUMMY-LAST-FIELD-DONT-REFERENCE-THIS
                                                                                          )
                                                                                  OF T)))))

(RPAQ CHARSETBLOCKBYTESIZE (CONSTANT (ITIMES BYTESPERWORD (INDEXF (FETCH (CHARSETBLOCK
                                                                              DUMMY-LAST-FIELD-DONT-REFERENCE-THIS
                                                                                    )
                                                                             OF T)))))

(RPAQ FONTDESCRIPTIONBYTESIZE (CONSTANT (ITIMES BYTESPERWORD (INDEXF (FETCH (FONTDESCRIPTION
                                                                              DUMMY-LAST-FIELD-DONT-REFERENCE-THIS
```

```
                                                                                          )
                                                                                  OF T)))))

(CONSTANTS (FONTTREENODEBLOCKBYTESIZE (CONSTANT (ITIMES BYTESPERWORD (INDEXF (FETCH (FONTTREENODEBLOCK
                                                                             DUMMY-LAST-FIELD-DONT-REFERENCE-THIS
                                                                                                )
                                                                                   OF T)))))
       (CHARSETBLOCKBYTESIZE (CONSTANT (ITIMES BYTESPERWORD (INDEXF (FETCH (CHARSETBLOCK
                                                                             DUMMY-LAST-FIELD-DONT-REFERENCE-THIS
                                                                                   )
                                                                          OF T)))))
       (FONTDESCRIPTIONBYTESIZE (CONSTANT (ITIMES BYTESPERWORD (INDEXF (FETCH (FONTDESCRIPTION
                                                                             DUMMY-LAST-FIELD-DONT-REFERENCE-THIS
                                                                                      )
                                                                             OF T))))))
)
)

(DECLARE\: EVAL@COMPILE DONTEVAL@LOAD DOCOPY)
```

;;; the mapping from font family number to font family name for those fonts which don't have the name embedded in the font file.

```
(DECLARE\: EVAL@COMPILE

(RPAQQ \NOVAFONTFAMILYNAMES
       ((0 . |Classic|)
        (1 . |Modern|)
        (2 . |Titan|)
        (3 . |Pica|)
        (4 . |Trojan|)
        (5 . |Vintage|)
        (6 . |Elite|)
        (7 . |LetterGothic|)
        (8 . |Master|)
        (9 . |Cubic|)
        (10 . |Roman|)
        (11 . |Scientific|)
        (12 . |Gothic|)
        (13 . |Bold|)
        (14 . |OcrB|)
        (15 . |Spokesman|)
        (16 . |XeroxLogo|)
        (17 . |CenturyThin|)
        (18 . |ScientificThin|)
        (19 . |Helvetica|)
        (20 . |HelveticaCondensed|)
        (21 . |Optima|)
        (22 . |Times|)
        (23 . |Baskerville|)
        (24 . |Spartan|)
        (25 . |Bodoni|)
        (26 . |Palatino|)
        (27 . |Caledonia|)
        (28 . |Memphis|)
        (29 . |Excelsior|)
        (30 . |Olympian|)
        (31 . |Univers|)
        (32 . |UniversCondensed|)
        (33 . |Trend|)
        (34 . |BoxPS|)
        (35 . |Terminal|)
        (36 . |OcrA|)
        (37 . |Logo1|)
        (38 . |Logo2|)
        (39 . |Logo3|)
        (40 . |Geneva2|)
        (41 . |Times2|)
        (42 . |Square3|)
        (43 . |Courier|)
        (44 . |Futura|)
        (45 . |Prestige|)
        (46 . |ALLetterGothic|)
        (47 . |CenturySchoolBook|)
        (48 . |Spare1|)
        (49 . |Spare2|)
        (50 . |Spare3|)
        (51 . |Spare4|)
        (52 . |Spare5|)
        (53 . |Melior|)
        (54 . |PCTerminal|)
        (55 . |ITCAmericanTypewriter|)
        (56 . |ITCAvantGardeGothic|)
        (57 . |ITCAvantGardeGothicCondensed|)
        (58 . |ITCBauhaus|)
        (59 . |ITCBarcelona|)
        (60 . |ITCBenguiat|)
        (61 . |ITCBenguiatCondensed|)
        (62 . |ITCBenguiatGothic|)
        (63 . |ITCBerkeleyOldStyle|)
        (64 . |ITCBookman|)
        (65 . |ITCCaslonNo224|)
        (66 . |ITCCentury|)
        (67 . |ITCCheltenham|)
        (68 . |ITCClearface|)
        (69 . |ITCCushing|)
        (70 . |ITCEras|)
        (71 . |ITCFenice|)
        (72 . |ITCFranklinGothic|)
        (73 . |ITCFrizQuadrata|)
        (74 . |ITCGalliard|)
        (75 . |ITCGaramond|)
        (76 . |ITCIsbell|)
        (77 . |ITCItalia|)
        (78 . |ITCKabel|)
        (79 . |ITCKorinna|)
        (80 . |ITCLubalinGraph|)
        (81 . |ITCModernNo216|)
        (82 . |ITCNewBaskerville|)
        (83 . |ITCNewtext|)
        (84 . |ITCNovarese|)
        (85 . |ITCQuorum|)
        (86 . |ITCSerifGothic|)
        (87 . |ITCSouvenir|)
        (88 . |ITCSymbol|)
        (89 . |ITCTiffany|)
        (90 . |ITCUsherwood|)
        (91 . |ITCWeidemann|)
        (92 . |ITCVeljovic|)
        (93 . |ITCZapfBook|)
        (94 . |ITCZapfChancery|)
        (95 . |ITCZapfDingbats|)
        (96 . |ITCZapfInternational|)
        (97 . |Cipher|)
        (98 . |FlemishScriptII|)
        (99 . |Quartz|)
        (100 . |QuartzA|)
        (101 . |QuartzT|)
        (102 . |Souvenir|)
        (103 . |Shimmer|)
        (104 . APL)
        (105 . |Arrows|)
        (106 . |BravoX|)
        (107 . |ClassicPiOne|)
        (108 . |ClassicPiTwo|)
        (109 . |Cream|)
        (110 . |Cyrillic|)
        (111 . |Dots|)
        (112 . |Gacha|)
        (113 . |Gates|)
        (114 . |HelveticaD|)
        (115 . |Hippo|)
        (116 . |Keyhole|)
        (117 . |Laurel|)
        (118 . |LogoOutline|)
        (119 . |LSIGates|)
        (120 . |MarqHippo|)
        (121 . |MarqRoman|)
        (122 . |Math|)
        (123 . |Mathology|)
        (124 . |OldEnglish|)
        (125 . |RomanPS|)
        (126 . |Sigma|)
        (127 . |Splunk|)
        (128 . |Template|)
        (129 . |Testfont|)
        (130 . |TimesRoman|)
        (131 . |TimesRomanD|)
        (132 . |TitanLegal|)
        (133 . WSSA)
        (134 . |XeroxBook|)
        (135 . |LucidaRoman|)
        (136 . |MonoSpace|)
        (137 . |Spare6|)
        (138 . |Spare7|)
        (139 . |Spare8|)
        (140 . |Spare9|)
        (141 . |Spare10|)))

(CONSTANTS \\NOVAFONTFAMILYNAMES)
)


;;; initialize the "noticed" fonts structure and set up the extensions so we can use the font files
```

```
(DEFGLOBALVAR *NOVAFONT-INFO* NIL)
```

;;; correct some omissions in the family aliases and printwheel fonts

```
(LISTPUT INTERPRESSFAMILYALIASES 'XEROXLOGO 'LOGOTYPES-XEROX)

(|pushnew| INTERPRESSPRINTWHEELFAMILIES 'SCIENTIFICTHIN 'OCRB 'OCRA)
```

;;; some things we need for compiling. Also need EXPORTS.ALL

```
(DECLARE\: EVAL@COMPILE DONTCOPY

(FILESLOAD (LOADCOMP)
       INTERPRESS LLCHAR)
)
```

;;; install this:

```
(DECLARE\: DONTEVAL@LOAD DOCOPY

(MOVD? '\\READDISPLAYFONTFILE '\\NO-NOVA-READDISPLAYFONTFILE)

(MOVD? '\\CREATECHARSET.IP '\\NO-NOVA-CREATECHARSET.IP)

(MOVD '\\READDISPLAYFONTFILE.NOVA '\\READDISPLAYFONTFILE)

(MOVD '\\CREATECHARSET.IP.NOVA '\\CREATECHARSET.IP)
)
```

;;; some hints for the compiler (system generated)

```
(DECLARE\: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS

(ADDTOVAR NLAMA )

(ADDTOVAR NLAML )

(ADDTOVAR LAMA READ-NOVAFONT-FONTHEADER READ-NOVAFONT-FILEHEADER)
)

(PUTPROPS NOVAFONT COPYRIGHT ("Xerox Corporation" 1986 1987 1988 1989))
```

{MEDLEY}<lispusers>NOVAFONT.;1   28-Jun-2024 18:34:03
-- Listed on 30-Jun-2024 13:14:28 --

FUNCTION INDEX

CONSTANT INDEX

VARIABLE INDEX

RECORD INDEX

MACRO INDEX