11 Soft Font Creation

Introduction

A font that is downloaded (transferred) from a computer to a printer is called a soft font. A PCL soft font contains a **font header** and a set of **character definitions**. The font header and character definitions contain all the information needed to format a font for use in the HP LaserJet printers.

Every PCL font header begins with a font descriptor, which identifies the basic characteristics common to all characters of a font, such as font type, baseline position, character cell width and height, character orientation, symbol set, etc.

Every PCL character definition contains a character descriptor and a body of character data. Furthermore, the character definition always consists of one or more character data blocks. Each character data block begins with its own character data block header.

The character descriptor is a block of data that identifies the characteristics for a specific character, such as its position, and the cursor position after printing. The character data which follows defines the shape of the character.

This chapter describes the font header and character definition formats for PCL Bitmap fonts, Intellifont scalable and TrueType scalable fonts. By formatting a font consistent with the header format requirements, a user may download this information (the font) to the printer using the Font Header command and the Character Descriptor/Data command. One additional command, the Character Code command, required to identify the ASCII character code assigned to each character, is also described in this chapter.

The definition of a font with a quantity *n* characters would appear as shown below.

Table 11-1

Font ID Command
Font Header
Character Code$_1$
Character Descriptor$_1$
Character Data$_1$
Character Code$_2$
Character Descriptor$_2$
Character Data$_2$
⋮
Character Code$_n$
Character Descriptor$_n$
Character Data$_n$

Font Classifications

There are three basic classifications of fonts accepted by the HP LaserJet printer: **PCL bitmap**, **Intellifont scalable**, and **TrueType scalable**. Several different font header and character descriptor formats are available for the different font classifications. All are presented in this chapter.

Note Not all font classifications are supported in all HP LaserJet Family printers. Refer to the *PCL 5 Comparison Guide* or the printer *User's Manual* for specific information.

With the information provided in the section for bitmap fonts, it is possible to format a PCL bitmap character/font for the printer. However, to format an Intellifont or TrueType scalable font, additional information is required.

Intellifont scalable fonts are formatted to use Agfa Scaling Technology. Intellifont scalable fonts are described in detail in the document, *Intellifont Scalable Typeface Format*, available from Agfa Division, Miles Inc. (Refer to *Related Documents*, located in the front of this manual, for information on how to obtain this document.)

TrueType scalable fonts are described in detail in the document, *True Type Font Files*. (Refer to *Related Documents*, located in the front of this manual, for information on how to obtain this document.)

Note

The documents, *Intellifont Scalable Typeface Format* and *True Type Font Files* do not contain descriptions of scalable PCL fonts. Instead, they contain descriptions of files from which PCL fonts can be built.

Coordinate System

Both bitmap and scalable characters are designed in an area referred to as a *cell* or *window,* and each has its own coordinate system and set of units.

Bitmap Fonts

Characters of a bitmap font are designed within a rectangular area referred to as a cell. The bitmap character cell is illustrated in Figure 11-3, Figure 11-6, and Figure 11-7. The physical coordinate system is defined in terms of the directions of raster scan (X) and paper motion (Y), as illustrated in the following figure.

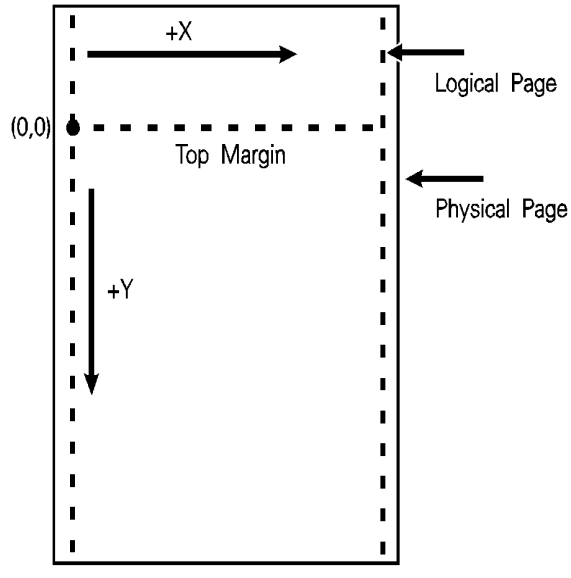


Figure 11-1 Bitmap Physical Coordinate System

Note The LaserJet IID, IIP, 2000, and all LaserJet PCL 5 printers rotate fonts to match the paper's physical coordinate system.

Intellifont Scalable Fonts

Characters of an Intellifont scalable font are designed within a rectangular area known as the Agfa Design Window (Figure 11-2). The units of this coordinate system are .01mm square.

The master font design size is 250 points (a CG point=.01383 inches). There are 8782 units per Em at the Master Font Size.

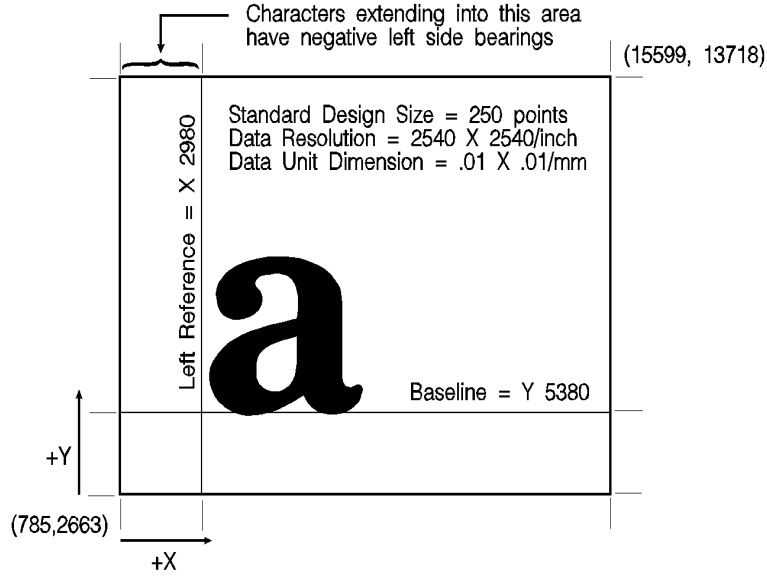


Figure 11-2 Agfa Design Window

TrueType Scalable Fonts

TrueType scalable font character coordinates are typically based on a system of 2048 units per Em. The baseline is defined by y-coordinate=0. The left reference is usually defined by x-coordinate=0 (although there is some variation among fonts). To determine the left reference line for an individual TrueType character, subtract the LSB value found in the hmtx table from the xMin value found in the glyf table. See *True Type Font Files* for more information.

Font Header Command

The Font Header command is used to download font header data to the printer.

$^{E}_{C}$ **) s # W** [font header data]

Default = 0
Range = 0 - 32767

The value field (#) identifies the number of bytes in the font header. The font header fields are described under *Font Header Format*, later in this chapter.

Note Two examples for downloading a Font Header are provided under *Font Header Examples*, after the *Font Header Format* section, later in this chapter.

Font Header Format

The **font header** describes those characteristics of a font which are common to all its characters. Five font headers are included here:

- **Format 0 Font Header for PCL Bitmapped Fonts** - This font header, shown in Table 11-2, is not recommended for LaserJet 4 and later printers. It is included in this manual to maintain backward-compatibility with earlier versions of PCL.
- **Format 20 Font Header for Resolution-Specified Bitmapped Fonts** - This header replaces the previous bitmap header, and has the added capability to specify a font's resolution. This font header is shown in Table 11-3.
- **Format 10 Font Header for Intellifont Bound Scalable Fonts** - For creating Intellifont scalable fonts which are restricted (bound) to a single symbol set. This font header is shown in .
- **Format 11 Font Header for Intellifont Unbound Scalable Fonts** - For creating Intellifont scalable fonts which are not bound to a single symbol set, but are instead composed of a range of compatible symbol collections. This font header is shown in .
- **Format 15 TrueType Scalable Font Header** - This new font header supports TrueType scalable fonts (bound or unbound). This font header is shown in Table 11-6.

Note

Use the Font ID command to designate a unique ID number prior to the download of a font header. If an existing font is already associated with this ID, the existing font is deleted upon the download of the font header. Unless otherwise specified, inappropriate values in a font header field invalidates the font download process; a font is not created and the associated font data is discarded.

Intellifont scalable font formatting also requires the *Intellifont Scalable Typeface Format* document, which supplements the information provided here. For information on how to obtain this document, refer to “Related Documents” in the front of this manual.

The figures that follow illustrate the font header formats for the various font classifications. The individual fields for the font headers are described following the figures.

Notes

Although some LaserJet printer models do not use all of the data in the font header and thus ignore many of the fields, a font creator should use valid values in all of the font header fields. This ensures font compatibility across the LaserJet printer family and with future printers, which may use these fields.

Those font header fields identified as “reserved” should be set to zero.

Table 11-2 Format 0 Font Header (for PCL Bitmapped Fonts)

Byte	15 (MSB) 8	7 (LSB) 0
0	Font Descriptor Size (64)	
2	Header Format (0)	Font Type
4	Style MSB	Reserved
6	Baseline Position	
8	Cell Width	
10	Cell Height	
12	Orientation	Spacing
14	Symbol Set	

Table 11-2 Format 0 Font Header (for PCL Bitmapped Fonts)

16	Pitch (Default HMI)	
18	Height	
20	x-Height	
22	Width Type	Style LSB
24	Stroke Weight	Typeface LSB
26	Typeface MSB	Serif Style
28	Quality	Placement
30	Underline Position (Distance)	Underline Thickness (Height)
32	Text Height	
34	Text Width	
36	First Code	
38	Last Code	
40	Pitch Extended	Height Extended
42	Cap Height	
44-47	Font Number ⋮	
48-63	Font Name ⋮	
64	Copyright (optional)	

Table 11-3 Format 20 Font Header (for Resolution-Specified Bitmapped)

Byte	**15 (MSB) 8**	**7 (LSB) 0**
0	Font Descriptor Size (68)	
2	Header Format (20)	Font Type
4	Style MSB	Reserved
6	Baseline Position	

Table 11-3 Format 20 Font Header (for Resolution-Specified Bitmapped) (continued)

8	Cell Width	
10	Cell Height	
12	Orientation	Spacing
14	Symbol Set	
16	Pitch (Default HMI)	
18	Height	
20	x-Height	
22	Width Type	Style LSB
24	Stroke Weight	Typeface LSB
26	Typeface MSB	Serif Style
28	Quality	Placement
30	Underline Position (Distance)	Underline Thickness (Height)
32	Text Height	
34	Text Width	
36	First Code	
38	Last Code	
40	Pitch Extended	Height Extended
42	Cap Height	
44-47	Font Number ⋮	
48-63	Font Name ⋮	
64	X Resolution	
66	Y Resolution	
n	Copyright (optional) ⋮	

Table 11-4 Format 10 Font Header (for Intellifont Bound Scalable)

Byte	15 (MSB) 8	7 (LSB) 0
0	Font Descriptor Size (minimum 80)	
2	Header Format (10)	Font Type
4	Style MSB	Reserved
6	Baseline Position	
8	Cell Width	
10	Cell Height	
12	Orientation	Spacing
14	Symbol Set	
16	Pitch (default HMI)	
18	Height	
20	x-Height	
22	Width Type	Style LSB
24	Stroke Weight	Typeface LSB
26	Typeface MSB	Serif Style
28	Quality	Placement
30	Underline Position (Distance)	Underline Thickness (Height)
32	Text Height	
34	Text Width	
36	First Code	
38	Last Code	
40	Pitch Extended	Height Extended
42	Cap Height	
44-47	Font Number ⋮	

Table 11-4 Format 10 Font Header (for Intellifont Bound Scalable) (continued)

48-63	Font Name ⋮	
64	Scale Factor	
66	X Resolution	
68	Y Resolution	
70	Master Underline Position	
72	Master Underline Thickness (Height)	
74	OR Threshold	
76	Global Italic Angle	
Desc. Size-2	Global Intellifont Data Size	
80	Global Intellifont Data ⋮	
n	Copyright (optional) ⋮	
	Reserved (0)	Checksum

Table 11-5 Format 11 Font Header (for Intellifont Unbound Scalable Fonts)

Byte	**15 (MSB) 8**	**7 (LSB) 0**
0	Font Descriptor Size (minimum 88)	
2	Header Format (11)	Font Type (10)
4	Style MSB	Reserved
6	Baseline Position	
8	Cell Width	
10	Cell Height	
12	Orientation	Spacing
14	Symbol Set	

Table 11-5 Format 11 Font Header (for Intellifont Unbound Scalable Fonts) (continued)

16	Pitch (default HMI)	
18	Height	
20	x-Height	
22	Width Type	Style LSB
24	Stroke Weight	Typeface LSB
26	Typeface MSB	Serif Style
28	Quality	Placement
30	Underline Position (Distance)	Underline Thickness
32	Text Height	
34	Text Width	
36	Reserved	
38	Number of Contours (Characters)	
40	Pitch Extended	Height Extended
42	Cap Height	
44-47	Font Number ⋮	
48-63	Font Name ⋮	
64	Scale Factor	
66	X Resolution	
68	Y Resolution	
70	Master Underline Position	
72	Master Underline Thickness	
74	OR Threshold	
76	Global Italic Angle	
78-85	Character Complement	

Table 11-5 Format 11 Font Header (for Intellifont Unbound Scalable Fonts) (continued)

Desc. Size-2	Global Intellifont Data Size	
Desc. Size	Global Intellifont Data ⋮	
n	Copyright (optional) ⋮	
	Reserved (0)	Checksum

Table 11-6 Format 15 Font Header (for TrueType Scalable Fonts)

Byte	**15 (MSB) 8**	**7 (LSB)0**
0	Font Descriptor Size (minimum 72)	
2	Header Format (15)	Font Type
4	Style MSB	Reserved
6	Baseline Position	
8	Cell Width	
10	Cell Height	
12	Orientation	Spacing
14	Symbol Set	
16	Pitch (default HMI)	
18	Height	
20	x-Height	
22	Width Type	Style LSB
24	Stroke Weight	Typeface LSB
26	Typeface MSB	Serif Style
28	Quality	Placement
30	Underline Position (Distance)	Underline Thickness

Table 11-6 Format 15 Font Header (for TrueType Scalable Fonts) (continued)

32	Text Height	
34	Text Width	
36	First Code	
38	Last Code/Number of Characters	
40	Pitch Extended	Height Extended
42	Cap Height	
44-47	Font Number ⋮	
48-63	Font Name ⋮	
64	Scale Factor	
66	Master Underline Position	
68	Master Underline Thickness	
70	Font Scaling Technology	Variety
72	*[additional data may be inserted here]* ⋮	
Desc. Size	Segmented Font Data ⋮	
# - 2	Reserved (0)	Checksum

Data Types

In the font header and character descriptor information that follows, the abbreviations shown below are used to define the data type of each field:

Table 11-7 Font Header Field Data Type Notation

(B)	: Boolean	(0, 1)
(UB)	: Unsigned Byte	(0 . . 255)
(SB)	: Signed Byte	(-128 . . 127)
(UI)	: Unsigned Integer	(0 . . 65535)

Table 11-7 Font Header Field Data Type Notation (continued)

(SI)	: Signed Integer	(-32768 . . 32767)
(ULI)	: Unsigned Long Integer	(0 . . 2^{32}-1)
(SLI)	: Signed Long Integer	(-2^{31} . . 2^{31}-1)
(ASCxx)	: ASCII string	array (0 . . xx-1) of characters

Font Descriptor Size (UI)

Specifies the number of bytes in the font descriptor. See the font header figure for the appropriate font descriptor size.

Header Format (UB)

The Header Format byte identifies the font to format (see below).

Table 11-8 Header Format Values

Value	Format
0	PCL Bitmap
10	Intellifont Bound Scalable
11	Intellifont Unbound Scalable
15	TrueType Scalable (bound or unbound)
20	Resolution-Specified Bitmap

Font Type (UB)

Font type describes the font's relation to symbol sets.

Table 11-9 Font Type Values

Value	Font Type
0	Bound font. Character codes 32 to 127 [decimal] are printable.
1	Bound font. Character codes 32 to 127 [decimal] and 160 to 255 [decimal] are printable.

Table 11-9 Font Type Values (continued)

2	Bound font. All character codes 0 to 255 are printable, except 0, 7 to 15, and 27 [decimal] (see note below).
10	Unbound font. Character codes correspond to HP MSL numbers (for Intellifont unbound scalable fonts).
11	Unbound font. Character codes correspond to Unicode numbers (for TrueType unbound scalable fonts).

Note Access to those codes which are unprintable, yet have a character defined, requires the use of the Transparent Print Data command (refer to Chapter 8 for more information).

Style MSB (UI)

The Style MSB (byte 4) is combined with the Style LSB (byte 23) to make the style word. The contents of the style word are described below. The Style word (decimal) is calculated using the formula:

Style Word = Posture + (4 x Width) + (32 x Structure)

The binary structure of the Style word is shown below.

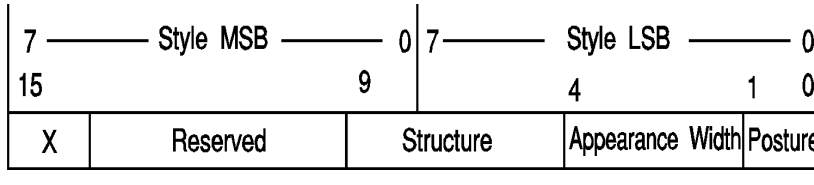


Table 11-10

Value	Posture (StyleWord partial sum)
0	Upright
1	Italic
2	Alternate Italic
3	Reserved

Table 11-11

Value	Appearance Width (multiply by 4 for StyleWord partial sum)
0	Normal
1	Condensed
2	Compressed or Extra Condensed
3	Extra Compressed
4	Ultra Compressed
5	Reserved
6	Extended or Expanded
7	Extra Extended or Extra Expanded

Table 11-12

Value	Structure (multiply by 32 for StyleWord partial sum)
0	Solid
1	Outline
2	Inline
3	Contour, Distressed (edge effects)
4	Solid with Shadow
5	Outline with Shadow
6	Inline with Shadow
7	Contour with Shadow
8-11	Patterned (complex patterns, subject to type family)
12-15	Patterned with Shadow
16	Inverse
17	Inverse in Open Border
18-30	Reserved
31	Unknown Structure

Note The reserved bits (15 - 10) should be set to zero.

If a value is requested, and a match not made, the request is ignored and the current font selection process continues as if the parameter was never requested (but it is saved in the attribute table).

Example

Assuming a font style of “italic compressed contour” is desired, the value (#) would be:

$$1 + (2\times4) + (3\times32) = 105$$

Baseline Position (UI)

Bitmap Font - Specifies the distance from the top of the cell to the baseline. The baseline is the dot row on which all of the characters in a given line appear to stand (see). The measurement of this distance is in font resolution dots, as defined in the Resolution Field of a Format 20 font header (default=300 dpi).

Intellifont Scalable - Specifies a Y-coordinate in the design window (refer to Figure 11-2.)

TrueType Scalable - Baseline Position must be set to zero.

Cell Width (UI)

Specifies the width of the cell. The cell must be wide enough to accept the widest character. The cell width range is 1 to 65535.

Bitmap Font - Specified in PCL coordinate system dots.

Scalable Font - Specified in design units.

Cell Height (UI)

Specifies the height of the cell. The design cell for a font must be tall enough to accept the tallest character and greatest descender. The legal range is 1 to 65535.

Bitmap Font - Specified in PCL coordinate system dots.

Scalable Font - Specified in design units.

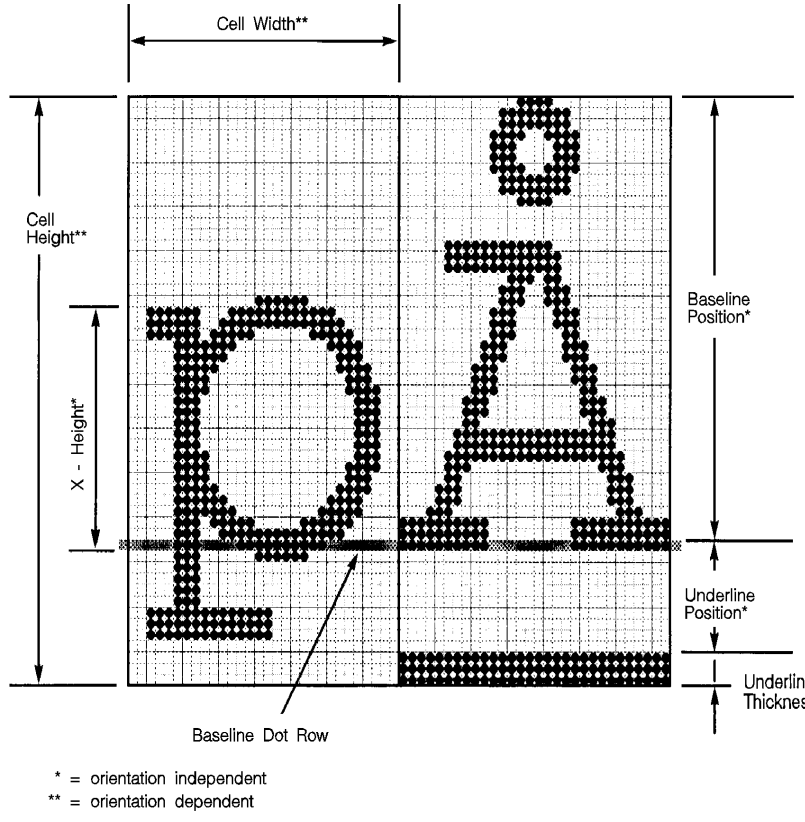


Figure 11-3 Character Cell - Bitmap

Orientation (UB)

Specifies the orientation of the font. All characters within the font must have the same orientation as those specified in the font header; otherwise they are discarded as they are downloaded.

0 = portrait (0 degrees; the orientation of the raster scan of the printer)
1 = landscape (90 degrees counterclockwise)
2 = reverse portrait (180 degrees counterclockwise)
3 = reverse landscape (270 degrees counterclockwise)

Bitmap Font - Unsupported values invalidate font creation.

Scalable Font - set to zero.

Note

Hewlett-Packard recommends that bitmap soft fonts be designed in portrait (0°), using the paper motion and raster scan direction of the HP LaserJet Plus and LaserJet series II printers. The HP LaserJet IID, IIP, 2000, and PCL 5 LaserJet printers rotate the fonts to match the paper's physical coordinate system for the various paper sizes.

Spacing (B)

Specifies the spacing of the font. A value of zero (0) specifies fixed spacing and one (1) specifies proportional spacing.

Symbol Set (UI)

Specifies the symbol set for the font. This value is computed by taking the value of the value field for the symbol set, multiplying it by 32, adding the decimal (ASCII) value of the termination character (the symbol set ID character value) of the escape sequence, and subtracting 64.

$$\text{Font Descriptor Symbol Set Value} = \left(\begin{array}{l}\text{Escape Sequence}\\ \text{Value Field Value X 32}\end{array}\right) + \left(\begin{array}{l}\text{Decimal Value of Escape Sequence}\\ \text{Termination Character} - 64\end{array}\right)$$

For example, to compute the value for the ASCII (ISO-6) symbol set (Value = **0**, ID = U):

$$0U = (0\times32) + (85–64) = 21$$

The legal range of symbol set escape sequence field values is 0 to 2047. Refer to Appendix C in the *PCL 5 Comparison Guide* for the HP defined symbol set values for use in the font header.

HP reserves the right to define the symbol set escape sequence value field values of 0 to 1023. Symbol set escape sequence value field values 1024 to 2047 are available for use by independent font vendors.

Symbol set escape sequence termination characters can be any upper case ASCII character “A” through “Z.” “Q” is reserved for use with HP “Specials” symbol sets and is not recommended for general use.

Notes

Symbol set ID’s of “@” and “X” do not have a corresponding Symbol Set selection command, sets marked as such can be selected only with the Font ID.

This field must have a value of 56 for a type 10 or 11 font (unbound Intellifont scalable) to be valid.

Pitch (UI)

Bitmap Font - Specifies the pitch of the font in quarter dots (four quarter-dot units, also known as radix dots, equal one dot). It combines with Pitch Extended to specify the pitch of the font in 1024^{th}-dots. Pitch defines the default HMI for the font.

For example, a a 17 cpi font designed at 300 dpi has a pitch value of 70 radix dots as calculated:

$$\frac{1\ \text{inch}}{17\ \text{char.}} \times \frac{300\ \text{dots}}{\text{inch}} \times \frac{4\ \text{radix-dots}}{\text{dot}} = 70.588\ \text{radix dot}$$

The remainder 0.588 is converted back to dots and then to 1024^{th}-dots as shown below:

$$\frac{0.588\ \text{radix dot}}{4\ \text{radix dot}} \times \frac{1024\ \text{units}}{\text{dot}} = 150\ \text{units/dot}$$

Pitch Extended is set to 150 1024^{ths}-units.

Note

For a proportional font, the width “printed” for a control code Space is determined by the pitch value, unless an HMI command is received following the selection of the font.

Scalable Fonts - Contains the master design space width (escapement) of the font in design units.

Height (UI)

Bitmap Font - Specifies the design height of the font in quarter-dots (radix dots). This value, converted to points, is used as the height characteristic value of the font. A PCL point is $\frac{1}{72}$ (0.01389) inch. It combines with Height Extended to specify the design height of the font in 1024th-dot (fonts designed at 300 dpi).

For example, a 10 point font at 300 dpi has a height of 166 quarter-dots (radix dots) (1200 quarter dots/inch, $\frac{1}{72}$ inch/point) as calculated:

$$\frac{10 \text{ point}}{} \times \frac{1 \text{ inch}}{72 \text{ point}} \times \frac{300 \text{ dots}}{\text{inch}} \times \frac{4 \text{ quarter-dots}}{\text{dot}} = 166.667$$

The remainder 0.667 is converted back to dots and then to

1024th-dot for a value of 170 1024th-dot for the Height Extended field (similar to that shown in the example for Pitch, above).

Intellifont Scalable - Specifies the master design height of the font in $\frac{1}{8}$ points. A typical value for this field is 2000.

TrueType Scalable - Set the Height field to zero.

xHeight (UI)

Bitmap Font - Specifies the height of the lower case “x” in quarter-dots (radix dots).

Scalable Fonts - Specifies the distance from the baseline to the lower case “x” height in design units.

Width Type (SB)

Specifies the proportionate width of characters in the font.

Table 11-13 Width Type Values

Value	Width Type
-5	Ultra Compressed
-4	Extra Compressed
-3	Compressed or Extra Condensed
-2	Condensed
0	Normal
2	Expanded
3	Extra Expanded
Additional width types may be added by HP.	

Style LSB (UB)

The least significant byte (LSB) of the Style word. Refer to Style MSB for a description of the Style word.

Stroke Weight (SB)

Specifies the thickness of the strokes used in designing the font. The supported stroke weight values are -7 through 7. The thinnest stroke available is -7; the thickest stroke weight is 7. The standard stroke weight for a medium font is 0; the standard stroke weight for a bold font is 3; and, the standard stroke weight for a light font is -3.

Table 11-14 Stroke Weight Values

Value	Stroke Weight
-7	Ultra Thin
-6	Extra Thin
-5	Thin
-4	Extra Light

Table 11-14 Stroke Weight Values (continued)

-3	**Light**
-2	Demi Light
-1	Semi Light
0	**Medium, Book, or Text**
1	Semi Bold
2	Demi Bold
3	**Bold**
4	Extra Bold
5	Black
6	Extra Black
7	Ultra Black

Typeface (UB)

This field specifies the HP typeface number of the font. The current version of this field, supported by the &payette; printer, is described first. Then a previous field, supported in earlier printers, is described.

Current Usage

In the LaserJet 4 printer version of this field, an unsigned short integer is assembled from the two unsigned bytes of data. Printers, when seeking to match a typeface request with available font resources, may treat the typeface number as a single value. If an exact match cannot be made, the request may be ignored (for selection purposes, however, the font select table is updated).

The procedure for allocating typeface numbers for the font products of various vendors, however, will consider the typeface number to be composed of two distinct fields: a vendor field (consisting of the four most significant bits) and a typeface family field (consisting of the 12 least significant bits). The following diagram illustrates this scheme:

Table 11-15 Typeface Family Value (Current)

15 12	11 0
Vendor	Typeface Family

Vendor Number - Bits 15 - 12. This value is assigned by HP and is between decimal values 0 and 15.

Table 11-16 Current Vendor Number Values

Value	Vendor
= 0	Reserved
= 1	Agfa Division, Miles Inc.
= 2	Bitstream Inc.
= 3	Linotype Company
= 4	The Monotype Corporation plc
= 5	Adobe Systems Inc.
= 6-15	(Reserved)

Typeface Family Number - Bits 11 - 0 This value is between 0 and 4095. See Appendix C in the *PCL 5 Comparison Guide*.

Typeface Family Values are calculated according to the following formula:

$$\text{Typeface Base Value} + \left(\text{Vendor Value} \times 4096\right) = \text{Typeface Family}$$

Example

The HP typeface number for Agfa Dom Casual typeface is 4157 (vendor value=1, and typeface value=61):

61 + (**1** × 4096) = 4157

Previous Usage

The previous treatment of the Typeface field supported the LaserJet IIP, IID and LaserJet III family printers. It consisted of the Typeface Least Significant Byte (LSB; the original, one-byte typeface value used prior to the LaserJet IID printer) and the Typeface Most Significant Byte (MSB) in the font header.

The previous typeface family value field is shown below. It included a 4-bit field to specify the vendor number, a 2-bit field for the version number, and a 9-bit field which contained the typeface base number. The most significant bit of the MSB was always zero.

Table 11-17 Typeface Family Value (Previous)

15	14	10	8	0
0	Vendor	Version	Typeface Base Value	

Table 11-18 Previous Vendor Number Values

Value	Vendor
0,1	Reserved
2	Agfa Division, Miles Inc.
4	Bitstream Inc.
6	Linotype Company
8	The Monotype Corporation plc
10	Adobe Systems Inc.
3,5,7,9,11-15	(Reserved)

Vendor-Version The Vendor-version (bits 10 and 9) value was from 0 to 3. It changed when the vendor changed the width or design of the characters in a font.

Typeface Base Value The Typeface Base Number (bits 0 through 8) ranged from 0 to 511. Some of these values referred to the styles that vary by structure and appearance width (such as Helvetica Condensed, Helvetica Outline, etc.). Do not use these values in new designs since they are being deleted. Refer to Appendix C in the *PCL 5 Comparison Guide* for a list of typeface families and their typeface base values.

Note For future compatibility, use the two-byte (typeface MSB/LSB) typeface family value. All scalable fonts use the larger typeface family value. Older bitmap fonts use the smaller typeface base value.

Serif Style (UB)

Specifies one of the following defined serif styles.

Serif Style values 0-63 (the lower six bits of the style field) are ignored by the printer for bitmap fonts. However, the upper two bits (bits 6 and 7) are used by a scalable font header to determine the serif style of the typeface insensitive characters to complement the font. Serif style values for the lower six bits are listed in the table below. Serif style values for the upper two bits are listed in the following table.

Table 11-19 Serif Style Values

Value	Serif Style
0	Sans Serif Square
1	Sans Serif Round
2	Serif Line
3	Serif Triangle
4	Serif Swath
5	Serif Block
6	Serif Bracket
7	Rounded Bracket
8	Flair Serif, Modified Sans
9	Script Nonconnecting
10	Script Joining

Table 11-19 Serif Style Values (continued)

11	Script Calligraphic
12	Script Broken Letter
13-63	Reserved
Values for bits 6 & 7	
64	Sans Serif
128	Serif
192	Reserved

Quality (UB)

This field specifies the quality of the font.

Table 11-20 Quality Values

Value	Quality
0	Data processing (draft)
1	Near Letter Quality
2	Letter Quality

Placement (SB)

Placement specifies the position of character patterns relative to the baseline.

Bitmap Font - The placement values for bitmap fonts are listed in the following table.

Table 11-21 Bitmap Font Placement Values

Value	Placement
1	Superior
0	Normal
-1	Inferior

Scalable Font - Set the Placement field to zero.

Underline Position (Distance) (SB)

Bitmap Font - Specifies the distance from the baseline to the top dot row of the underline in font design dots. Zero specifies an underline position at the baseline. A positive value specifies an underline position above the baseline. A negative value specifies an underline position below the baseline.

Scalable Font - Set Underline Position to zero. The Master Underline Position field (see below) identifies this information for scalable fonts.

Underline Thickness (UB)

Specifies the thickness of the underline in font design dots for a bitmap font.

Bitmap Font - A bitmap font prints three-dot thick underlines at 300 dpi (six-dot thick at 600 dpi).

Scalable Font - Set Underline Thickness to zero. The Master Underline Thickness field (see below) identifies this information for scalable fonts.

Text Height (UI)

Specifies the font's optimum inter-line spacing. This value is typically equal to 120% of the height of the font.

Bitmap Font - Specified in quarter-dots (radix dots).

Scalable Fonts - Specified in design units.

Text Width (UI)

Specifies the font's average lowercase character width. (This average width may be weighted on the basis of relative frequency.)

Bitmap Font - Specified in quarter-dots (radix dots).

Scalable Font - Specified in design units.

First Code (UI)

First Code specifies the character code of the first printable character in the font. This value is between 0 and 255 inclusive. The Space Character may be printable and will print an image if one is defined, otherwise a Space control code is executed. Currently, PCL 5 LaserJet printers use the Font Type field to determine the first and last codes of the symbol set, as shown below:

Table 11-22

Font Type	First Code../..Last Code
0	32/127
1	32/127 - 160/255
2	0/255
10	Set to 0 (for unbound font)
11	Set to 0 (for unbound font)

Last Code / Number of Characters (UI)

Bound Font: Specifies the last code in the font. This value may be greater than the last code of the symbol set as implied by the font type because there may be components of compound characters that are not part of the symbol set but must be downloaded. The printable codes are implied by the font type (refer to first code described above).

Unbound Font: For an unbound font (type 10 or 11), this field specifies the maximum number of characters that can be down-loaded into the font.

Pitch Extended (UB)

Bitmap Font - This is an addition to the Pitch field which extends the pitch an extra eight bits. The value of this field is in font design units. For example, a 17 pitch font designed at 300 dpi has a Pitch field of 70 (17.5 dots, or 17.1429 cpi) and a Pitch Extended field of 150 (0.1465 dots additional, which adds to 17.6465 dots, or 17.0005 pitch).

An example for calculating the Pitch and Pitch Extended fields is provided in the Pitch field description, above.

Scalable Font - Set Pitch Extended field to zero.

Height Extended (UB)

Bitmap Font - This is an addition to the Height field which extends the height an extra eight bits. The value of this field is in font design units. For example, a 10 point font designed at 300 dpi would have a Height field of 166 (41.5 dots, or 9.96 points) and a Height Extended field of 170 (0.1660 dots additional, which adds to 9.9998 points). This field is calculated similar to the Pitch Extended field. Refer to the Pitch description, above.

Scalable Font - Set The Height Extended field to zero.

Cap Height (UI)

Cap Height is a percentage of the Em of the font and is used to calculate the distance from the capline (top of an unaccented, uppercase letter, such as an “H”) to the baseline.

Bitmap Font - Fonts containing a 0 in this field are assumed to have a cap height percentage of 70.87% of Em (Em being a measure, in points, of the height of the body of the font.

The Cap Height data is represented as the product of the cap height percentage and the maximum unsigned integer:

$$0.7087 \times 65535 = 46{,}445$$

For non-zero values, the Cap Height % is calculated as follows:

$$\% = \frac{\text{Cap Height Data}}{65535} \times 100$$

Scalable Font - Contains the cap height in design units.

Font Number (ULI)

The Font Number field uses four bytes (byte 44, 45, 46, and 47). The lower three bytes (45, 46, and 47) contain the font number in hexadecimal. This is the number the vendor assigns to their typeface. The most significant byte (byte 44) consists of a flag in the most significant bit indicating whether the font is in its native (0) format or has been converted (1) from another format. The remaining lower seven bits contain the ASCII decimal value for the first initial of the font vendor's name (this is assigned by Hewlett-Packard). The following initials have been assigned:

Table 11-23

Initial	HexValue	Vendor Name
A	41	Adobe Systems Inc.
B	42	Bitstream Inc.
C	43	&AGFA;
H	48	Bigelow && Holmes
L	4C	Linotype Company
M	4D	Monotype Corporation plc

For example, the number that Agfa assigns for a CG Times Bold Italic, native format, font is 92505. This number is converted to hexadecimal and used for the lower three bytes of the Font Number. Bit 8 of byte 44 is 0, since the native format is used and the lower seven bits are the ASCII value for "C" (C for Compugraphic; 0100 0011). This process is summarized below.

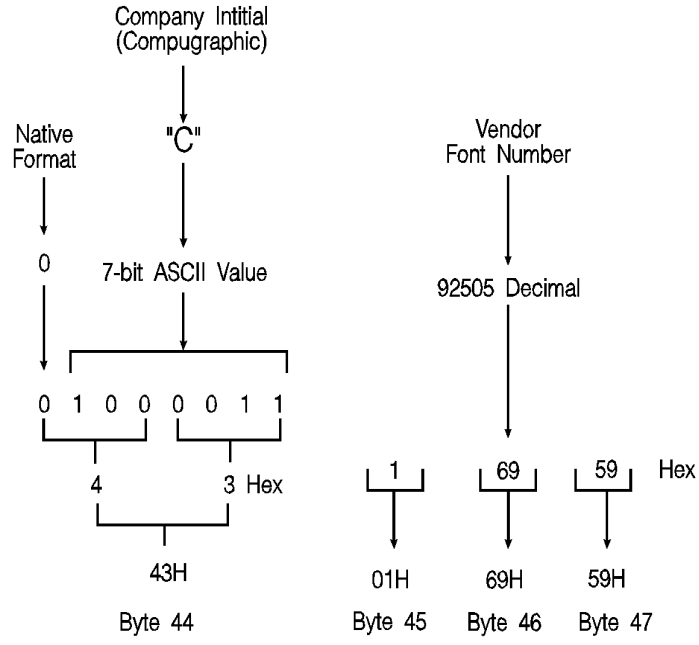


Figure 11-4

This field is ignored by the printer for bitmap fonts.

Font Name (ASC16)

This is a 16 character ASCII field to which you may assign a font name. The font name is used in the Typeface List (or Font List printout), under **Name** or **Typeface** (if the printer does not have a name string assigned to the typeface family code in its font selection table).

X Resolution (UI)

The X Resolution field is the pixel resolution in the X scan direction at which the font was designed.

Note This field is not present in the Format 15 Font Header and is not necessary for TrueType fonts.

Y Resolution (UI)

The Y Resolution field is the pixel resolution in the Y scan direction at which the font was designed.

Note This field is not present in the Format 15 Font Header and is not necessary for TrueType fonts.

Scale Factor (UI)

The Scale Factor field indicates the number of design units per Em, and is the unit used for all scalable metrics in the font header. It is used for TrueType and Intellifont scalable soft fonts.

Master Underline Position (SI)

The Master Underline Position is the top of the PCL floating underline with respect to the baseline in design units.

Note For scalable fonts, the Master Underline Position field replaces the 1-byte Underline Position field.

Master Underline Thickness (Height) (UI)

The Master Underline Thickness field defines thickness of the floating underline in design units.

Font Scaling Technology (UB)

For scalable fonts, this field species the technology to be used for font scaling.

Table 11-24

Value	Font Scaling Technology
0	Intellifont
1	TrueType

Variety (UB)

The interpretation of this field depends on the value of the preceding (Font Scaling Technology) byte. For TrueType fonts, this field must be set to zero.

OR Threshold (UI)

Formerly called the “LRE Threshold,” this is the pixel size in design units above which the missing pixel recovery process is switched on in Intellifont scaling and rasterization.

Note The size of a pixel (in design units) increases as point size and device resolution decrease.

Global Italic Angle (SI)

The Global Italic Angle field contains the tangent of the italic angle times 2^{15} (relative to the vertical). Set this field to zero for upright fonts. For detailed information on Global Italic Angle, refer to *Intellifont Scalable Typeface Format*.

Global Intellifont Data Size (UI)

The Global Intellifont Data Size identifies the size of the Global Intellifont data block. For detailed information about Global Intellifont Data Size, refer to *Intellifont Scalable Typeface Format*.

Global Intellifont Data

For detailed information on Global Intellifont Data refer to *Intellifont Scalable Typeface Format*.

Character Complement (Array of UB)

This 8-byte field qualifies the compatibility of a type 10 or 11 font with various character sets. Each bit is independently interpreted with the exception of the least significant three bits. (Bit 63 refers to the most significant bit of the first byte, and bit 0 refers to the least significant bit of the eighth byte.)

Note In the Format 15 Font Header for TrueType Scalable Fonts, the data in this field is contained in the “CC” (Character Complement) field in the Segmented Font Data section immediately following the descriptor data. See “Segmented Font Data” later in this chapter

.

Table 11-25 MSL Symbol Index

Bit Field	Designated Use
58-63	Reserved for Latin fonts.
55-57	Reserved for Cyrillic fonts.
52-54	Reserved for Arabic fonts.
50-51	Reserved for Greek fonts.
48-49	Reserved for Hebrew fonts.
3-47	Miscellaneous uses (South Asian, Armenian, other alphabets, bar codes, OCR, Math, PC Semi-graphics, etc.).
0-2	Symbol Index field. 111 - MSL Symbol Index

Table 11-26 Unicode Symbol Index

Bit Field	Designated Use
32-63	Miscellaneous uses (South Asian, Armenian, other alphabets, bar codes, OCR, Math, etc.).
28-31	Reserved for Latin fonts.
22-27	Reserved for platform/application variant fonts.
3-21	Reserved for Cyrillic, Arabic, Greek and Hebrew fonts.
0-2	Symbol Index field. 110 - Unicode Symbol Index

Individually defined bits are shown in the following two tables:

Table 11-27 MSL Symbol Index Character Complement Bits

Bit	Value
63	0 if font is compatible with standard Latin character sets (e.g., Roman-8, ISO 8859-1 Latin 1); 1 otherwise.
62	0 if font is compatible with East European Latin character sets (e.g., ISO 8859-2 Latin 2); 1 otherwise.
61	0 if font contains Turkish character sets (e.g., ISO 8859/9 Latin 5); 1 otherwise.
34	0 if font has access to the math characters of the Math-8, PS Math and Ventura Math character sets; 1 otherwise.
33	0 if font has access to the semi-graphic characters of the PC-8, PC-850, etc. character sets; 1 otherwise.
32	0 if font is compatible with ITC Zapf Dingbats series 100, 200, etc.; 1 otherwise.
2, 1, 0	111 if font is arranged in MSL Symbol Index order.

Table 11-28 Unicode Symbol Index Character Complement Bits

Bit	Value
31	0 if font is compatible with 7-bit ASCII; 1 otherwise.
30	0 if font is compatible with ISO 8859/1 Latin 1 (West Europe) character sets; 1 otherwise.
29	0 if font is compatible with ISO 8859/2 Latin 2 (East Europe) character sets; 1 otherwise.
28	0 if font is compatible with Latin 5 (Turkish) character sets (e.g., ISO 8859/9 Latin 5, PC-Turkish); 1 otherwise.
27	0 if font is compatible with Desktop Publishing character sets (e.g., Windows 3.1 Latin 1, DeskTop, MC Text); 1 otherwise.

Table 11-28 Unicode Symbol Index Character Complement Bits

26	0 if font is compatible with character sets requiring a wider selection of accents (e.g., MC Text, ISO 8859/1 Latin 1); 1 otherwise.
25	0 if font is compatible with traditional PCL character sets (e.g., Roman-8, Legal, ISO 4 United Kingdom); 1 otherwise.
24	0 if font is compatible with the Macintosh character set (MC Text); 1 otherwise.
23	0 if font is compatible with PostScript Standard Encoding (PS Text); 1 otherwise.
22	0 if font is compatible with Code Pages (e.g., PC-8, PC 850, PC-Turk, etc.); 1 otherwise.
2,1,0	110 if font is arranged in Unicode Symbol Index order.

There are no invalid Character Complement field values. Examples of values for the field include:

Table 11-29

Bit Field	**Designated Use**
Value (hex)	**Meaning**
MSL:	
“0000000000000000”	Default complement; font is compatible with any character set.
“7fffffffffffffff”	Font is indexed in MSL and is compatible only with standard West Latin character sets.
“ffffffeffffffff”	Font is indexed in MSL and is compatible only with ITC Zapf Dingbat character sets.
Unicode:	
“ffffffff3ffffffe”	Font is indexed in Unicode and is compatible only with standard West Latin character sets.

Table 11-29 (continued)

“ffffffff5ffffffe”	Font indexed in Unicode and is compatible only with East Europe Latin character sets.

Checksum

The Checksum field is over bytes 64 through the end of the header. The checksum should contain a value which, when added to the sum of byte 64 through the reserved byte, equals a value which, when divided by 256 (modulo 256 arithmetic), results in a remainder of 0. For example, if the sum = 10,234 then, 10,234 mod 256 = 250. Therefore, the checksum should = 6 (since 250+6 = 256 which would produce 0 [mod 256]).

Note In the Format 15 Font Header for TrueType Scalable Fonts, this field is located at the end of the Segmented Font Data section immediately following the descriptor data. See “Segmented Font Data” later in this chapter.

Copyright

This field contains ASCII data and is optional.

Note In the Format 15 Font Header for TrueType Scalable Fonts, this field is located in the Segmented Font Data section immediately following the descriptor data. See “Segmented Font Data” later in this chapter.

Segmented Font Data (Format 15)

The Segmented Font Data section immediately follows the main body of a Format 15 Header for TrueType Scalable Fonts. Each segment contains three parts: a **Segment Identifier, Segment Size,** and **Data Segment.**

The Segmented Font Data section is terminated by the Null Segment. (In the deviant case where no Null Segment is encountered prior to the end of the font header &emdash; as defined in the Font Header command &emdash; the font is invalidated. A font also is invalidated in the event that a Null Segment is encountered too soon.)

Table 11-30 below shows the structure of the Segmented Font Data section.

Table 11-30 Segmented Font Data

Byte	15 (MSB) 8	7 (LSB) 0
x + 0	First segment, Segment Identifier	
x + 2	First segment, Segment Size	
x + 4 ⋮	First segment, Data Segment ⋮	
x + 4 + 1st seg size	Second segment: Segment Identifier, Size, Data Segment ⋮	
⋮	⋮	
# – 6	Null Segment Identifier (FFFF - hex)	
# – 4	Null Segment Size (0)	
# – 2	Reserved	Checksum
x = Font Descriptor Size. # = Font header length (as defined in Font Header command).		

Segment Identifier (UI)

Each entry in the Segmented Font Data Section has its own unique identification number. The following values are defined:

Table 11-31

Value	Mnemonic	Data Segment
17219	CC	Character Complement
17232	CP	Copyright
18260	GT	Global TrueType Data
18758	IF	Intellifont Face Data
20545	PA	PANOSE Description
22618	XW	XWindows Font Name
65535		Null Segment

Data segments with an unrecognized identifier are ignored.

Segment Size (UI)

For each entry in the Segmented Font Data section, the Segment Size indicates the number of bytes in the immediately following Data Segment. The size for the Null Segment is 0.

Formats of Data Segments

AP (Application Support Segment) The definition of this segment is reserved.

CC (Character Complement) This field has the same form (*i.e.,* 8 unsigned bytes) and function as does the Character Complement of Format 11 fonts. The Character Complement field should be present with type 10 and 11 (unbound) fonts, but has no role to play in type 0, 1 and 2 (bound) fonts.

CP, copyright This field will consist of ASCII data and is optional.

GI (Global Intellifont Data) Reserved for future use.

GT (Global TrueType Data) This data segment contains first a Table Directory, then five or more tables used by the TrueType font scaler. Every TrueType font needs to have this segment.

The Table Directory is patterned after the initial segment of the TrueType font file as described in *True Type Font Files*. The Table Directory has a 12-byte header and 16 bytes per entry in the Table Directory. The Table Directory is organized in alphabetical order by the 4-byte table names. For each entry, there is an offset relative to the beginning of the soft font's Global TrueType Data Segment.

The **Global TrueType Data** for every TrueType font entity must contain a **head, hhea, hmtx** and **maxp** table.

Another required table is the **gdir** table. When the font header is downloaded, the **gdir** table should have a size of 0 and an offset of 0. The **gdir** table is then built in RAM to accommodate the maximum number of glyphs to be downloaded to the given font — with 2 or 4 bytes of offset and 2 bytes of length per glyph. This maximum number of glyphs is obtained from the numGlyphs field of the **maxp** table. Entries in the **gdir** table are filled in by the TrueType rasterizer as characters are downloaded.

The optional **cvt, fpgm** and **prep** tables, as defined in *True Type Font Files*, typically appear in the Global TrueType Data Segments of hinted TrueType soft fonts, but should not appear in unhinted fonts.

IF (Intellifont Face Data) Reserved for future use.

PA (PANOSE Description) This data segment of variable length may be used for the purpose of font selection and substitution. Its definition continues to evolve. A 10-field (10-byte) version sufficient for the description of most Latin fonts appears under the OS/2 table in *True Type Font Files*.

PF (PS-Compatible Font Name) Reserved for future use.

XW (x-windows font name) This ASCII field contains standard X-Windows font names.

Checksum

The value of this byte, when added to the sum of all of the bytes from byte 64 of the descriptor through the Reserved byte, should equal 0 in modulo 256 arithmetic.

Font Header Examples

Two examples for downloading a Font Header are provided below; one for a bitmap font and one for an Intellifont scalable font.

Bitmap Example

To download a bitmap font header for a portrait HP Roman-8, 10 pitch, 12 point, upright, medium, Courier font, with an ID number of one, send:

$^{E}_{C}$***c1D** (set Font ID to 1)

$^{E}_{C}$**)s#W** (# = 64 bytes of font descriptor data + *x* bytes of optional data)

An example of the bitmap header is shown on the following page.

Table 11-32

FIELD NAME	VALUE	DESCRIPTION
Font Descriptor Size	64	Bytes
Header Format	0	Bitmap Font Format
Font Type	1	Eight Bit
Style MSB	0	
Reserved	0	
Baseline Position	40	
Cell Width	30	
Cell Height	53	
Orientation	0	Portrait
Spacing	0	Fixed Pitch
Symbol Set	277	8U: Roman-8
Pitch	120	Quarter Dots (30.00 Dots)
Height	200	Quarter Dots (50.00 Dots)
x Height	88	Quarter Dots (22.00 Dots)
Appearance Width	0	Normal

Table 11-32 (continued)

Style LSB	0	Upright, Normal Width, Solid (0,0,0)
Stroke Weight	0	Medium
Typeface LSB	3	Body Text
Typeface MSB	0	No Font Vendor ID
Serif Style	2	Serif Line
Quality	0	∅
Placement	0	∅
Underline Position	-10	∅
Underline Thickness	3	∅
Text Height	200	Quarter dots (50.00 Dots)
Text Width	120	Quarter Dots (30.00 Dots)
First Code	33	∅
Last Code	254	∅
Pitch Extended	0	∅
Height Extended	0	∅
Cap Height	36713	56.02% of Em
Font Number	0	No Font Vendor Number
Font Name	Courier	
....Copyright Statement .... (optional) }= *x* bytes added to header data		
....Application Support ....(optional) }= *x* bytes added to header data		

Intellifont Scalable Example

To download an Intellifont scalable header for an HP Roman-8, upright, medium, CG Times scalable font, with an ID number of one, send:

<sup>E</sub>C***c1D** (set Font ID to 1)

<sup>E</sub>C**)s#W** (# = 80 bytes of font descriptor data + *x* bytes of Global Intellifont data + *x* bytes of optional data)

Table 11-33

FIELD NAME	VALUE	DESCRIPTION
Descriptor Size	80	Bytes
Header Format	10	Scalable Font Format
Font Type	1	Eight Bit
Style MSB	0	
Reserved	0	
Baseline Location	5380	Y reference in Design Window
Cell Width	0	..not defined for Intellifont
Cell Height	0	..not defined for Intellifont
Orientation	0	..not defined for scalable fonts
Spacing	1	Proportional
Symbol Set	277	8U: Roman-8
Pitch	2602	29.63% Em Default HMI
Design Height	2000	250 Points * 8
x-Height	4009	45.65% Em, 68.52% Cap Height
Appearance Width	0	Normal
Style LSB	0	Upright, Normal Width, Solid (0,0,0)
Stroke Weight	0	Medium or Text Weight
Typeface LSB	5	Times Roman (generic design family)
Typeface MSB	16	Agfa
Serif Style	134	Serif, Bracketed (2,6)
Quality	0	..not defined for scalable fonts

Table 11-33 (continued)

Placement	0	..not defined for scalable fonts
Underline Position	0	..not defined for scalable fonts
Underline Thickness	0	..not defined for scalable fonts
Text Height	0	..not defined for Intellifont
Text Width	4391	Width of En Space
First code	33	
Last Code	273	Compound Pieces Present (n255)
Pitch Extended	0	..not defined for scalable fonts
Height Extended	0	..not defined for scalable fonts
Cap Height	5851	66.7% Em
Font Number	hex 43 01 69 54	Native, Agfa, CG Times (0,C,92500)
Font Name	“CG Times ”	(16 character ACSII field)
Scale Factor	8782	
X Resolution	2540	
Y Resolution	2540	
Master Underline Position	-1747	
Master Underline Thickness	449	
OR Threshold	176	
Global Italic Angle	0	
Global Intellifont Data Size	112	

Table 11-33 (continued)

....Global Intellifont Data		}= 112 bytes added to header data
....Copyright Statement (optional)		}= *x* bytes added to header data
....Application Support (optional)		}= *x* bytes added to header data

Character Definitions

Following the font header, the individual characters must be defined. Every PCL character definition contains a character descriptor and a body of character data. The character definition always consists of one or more character data blocks. Each character data block begins with its own header. The character data block header always has a size of 2 bytes.

The first data block of a character definition must always have a character descriptor immediately after its 2-byte header.

Character descriptor/data is downloaded using the Character Definition command preceding every character (see *Character Definition Command*).

Notes

A unique character code, using the Character Code command, must be designated prior to the download of a character descriptor and data. If the font being downloaded already contains a character with this code, the existing character is deleted during the download of the character descriptor and data.

Unless otherwise specified, inappropriate values in a character descriptor field invalidates the character download process; a character is not created, and the associated descriptor and data are discarded.

An undefined printable character is one which is in the printable range of the font type but has no defined pattern. Attempts to print an undefined printable character from a font result in the execution of a Space control code.

Character descriptor fields identified as “reserved” should be set to zero.

If the total byte count of the character descriptor and data exceeds 32767 bytes, then the remaining data must be sent using the continuation descriptor.

Character Code Command

The Character Code command establishes the decimal code that is associated with the next character downloaded. This value is used to reference the character for printing.

$$E_C * c \# E$$

\# =character code

Default = 0
Range = 0 - 65535

Notes

For unbound fonts, the character code for a given character equals its symbol index value.

For TrueType fonts, a special code must be used to download glyphs which never stand alone as characters. FFFF (hex) should be used for this purpose.

Example

To designate the character code for an ASCII lower-case “p”, send:

$$E_C * c 112 E$$

Character Definition Command

The Character Descriptor and Data command is used to download character data blocks to the printer for both bitmap and scalable fonts.

$^{E}_{C}$ **(s # W** [character descriptor and data]

Default = N/A
Range = 0 - 32767

The value field (#) identifies the number of bytes in the immediately following character data block. The maximum number is 32767.

For a detailed description of the Character Descriptor fields for bitmap fonts refer to *Character Descriptor and Data Format for PCL Bitmap Fonts.* For Intellifont scalables, refer to “Character Descriptor and Data Format for Intellifont Scalable Fonts.” For TrueType fonts, refer to “Character Descriptor and Data Format for TrueType Fonts.”

Note

Examples for defining a bitmapped portrait and landscape character are provided under *Character Definition Examples*, after the *Character Descriptor Formats* section, later in this chapter.

Character Descriptor Formats

Character definition formats for PCL Bitmap, Intellifont Scalable and TrueType Scalable fonts are shown on the following pages.

Note The following notation is used to define the data type of each field in the character descriptors.

Table 11-34 Character Descriptors/Data Continuation Block

(B)	: Boolean	(0,1)
(UB)	: Unsigned Byte	(0 . . 255)
(SB)	: Signed Byte	(-128 . . 127)
(UI)	: Unsigned Integer	(0 . . 65535)
(SI)	: Signed Integer	(-32768 . . 32767)

Character Descriptor and Data Format for PCL Bitmap Fonts

The descriptor of a PCL bitmap character is at least 14 bytes long and contains information such as the character's width and height.

The character data is binary (raster) data that identifies the shape of the character.

Table 11-35 shows the format of the bitmap character descriptor and data.

Table 11-35 PCL Bitmap Character Descriptor and Data Format

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (4)	Continuation (0)
2	Descriptor Size (14)	Class (1)
4	Orientation	Reserved (0)

Table 11-35 PCL Bitmap Character Descriptor and Data Format (continued)

6	Left Offset	
8	Top Offset	
10	Character Width	
12	Character Height	
14	Delta X	
16	Raster Character Data: (in bytes) ⋮	

Table 11-36 PCL Bitmap Continuation Character Descriptor and Data Format

Byte	**15 (MSB) 8**	**7 (LSB) 0**
0	Format (4)	Continuation (non-zero)
2	Raster Character Data: (in bytes) ⋮	

Format (UB)

This is the first byte of every character data block header. It specifies the format of the character descriptor and data. The format number used for bitmap fonts is 4. This format must match that of the Font Header.

Table 11-37

Value	Format
4	LaserJet Family (Raster)
10*	Intellifont Scalable
15*	TrueType Scalable
* These are described later in this chapter.	

If the format number is different from that expected by the device, the character is discarded.

Continuation (B)

This is the second (and last) byte of every character data block header. It specifies whether the following data is the first (0) data block of a new character definition, or a continuation (1) block for a character definition which has already been received by the printer. Because the value field in a Character Definition command is limited to 32767 bytes, characters whose byte count exceed this must be sent in two or more blocks.

Descriptor Size (UB)

This is the first byte of the character descriptor. It specifies the size of the character descriptor in bytes. The descriptor size used by the HP LaserJet printer family for bitmap fonts is 14.

Class (UB)

Specifies the format of the character data. For bitmap fonts only values 1 and 2 are used, as described below.

Table 11-38

Value	Class
1	Bitmap
2	Compressed Bitmap
3*	Contour (Intellifont Scalable)
4*	Compound Contour (Intellifont Scalable)
15*	TrueType Scalable
* These are described later in this chapter.	

Class 1 - Bitmap Data

Class 1 or bitmap (raster) character data is a string of bytes containing the dot-per-bit image of the character, no data compression. If a bit is set to one, the corresponding dot is printed. The data is grouped in dot rows. A row describes a one-dot-high strip of the character from left to right, in the direction of the printer's raster scan (see the Portrait Bitmap Character Data Example, at the end of this chapter). Zeroed bits must be added to the end of each row to make it contain an integral number of bytes. The dot rows are organized from top to bottom of the character. For example, the first dot row of data corresponds to the top dot row of the character.

The number of bytes of the character data should be exactly Character Width (in bytes) times Character Height. If more

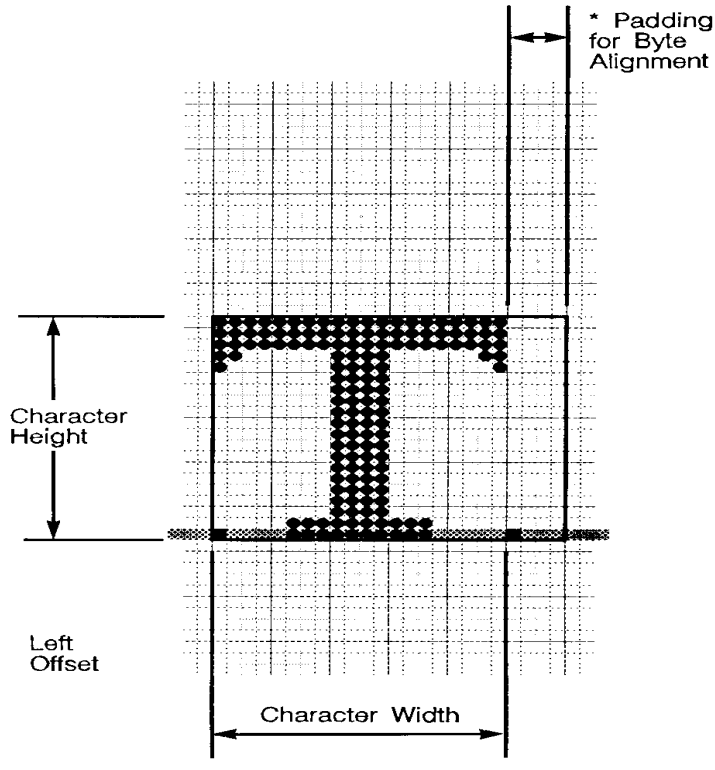
data is received, it is discarded; if less data is received, the character consists of only the data downloaded.

Class 2 - Compressed Bitmap Data

For a compressed bitmap character, the data is composed of a string of bytes using a run-length encoding with line repetition compressed format (see Figure 11-2). The first byte indicates the number of times the first raster row is repeated after its initial occurrence. It is assumed that the first pixel in a row is white, hence the second byte indicates how many white pixels start the row. The third byte indicates how many black pixels, the fourth byte indicates the number of white pixels again, etc. If the first pixel in a row is black, the white pixel indicator (the second byte) is 0. If there are more than 255 pixels in a row of the same type, there is a byte containing 255, followed by a 0 byte, followed by a byte containing the count of remaining pixels of the current type.

The width of each row is determined by the character width (in dots) as specified in the character descriptor for the character. The pixel count (number of 1's and 0's bits) for each row in the character cell must equal the character width. For example, in Figure 11-5, the cell width is 20, thus each row (excluding the repetition count byte) adds up to 20.

Once the row has been filled, the row is duplicated as indicated in its first byte, then a new row is started.



Line Repetition	# white pixels	# black pixels	# white pixels	# black pixels	# white pixels	# black pixels
2	0	20	-	-	-	-
0	0	2	6	4	6	2
0	0	1	7	4	7	1
12	8	4	8	-	-	-
1	5	10	5	-	-	-

Uncompressed - 60 Bytes
Compressed - 25 Bytes

* Byte alignment is necessary only for raster data i.e. (Not necessary for compressed raster data.)

Figure 11-5 Class 2 Character Data

Orientation (UB)

Orientation byte specifies the orientation of the character. The orientation of the character must match the orientation of the font.

Table 11-39

Value	Orientation
0	Portrait
1	Landscape
2	Reverse portrait
3	Reverse landscape

If the orientation is not supported or is different from the orientation specified in the font header, the character is discarded.

Left Offset (SI)

Left offset specifies the distance in dots from the reference point to the left side of the character pattern on the physical page coordinate system (this value is orientation dependent). The left and top offsets locate the character reference point about the cursor position (see Figure 11-6 and Figure 11-7).

PCL 5 printers support kerning (both negative left and right side bearings) of both fixed-pitch and proportionally-spaced fonts. Note that large offsets could place the character off the printable area of the page causing the character to be clipped.

The legal range for the left offset is –16384 to 16384 dots.

Top Offset (SI)

Top offset specifies the distance in dots from the reference point to the top of the character pattern on the physical coordinate system (this value is orientation dependent.) The left and top offsets locate the character reference point about the cursor position (see Figure 11-6 and Figure 11-7). The legal range for the top offset is –16384 to 16384 dots.

Character Width (UI)

The Character Width, used for bitmap fonts only, identifies the width of the character in dots on the physical coordinate system. Generally, this width is from the farthest left black dot to the farthest right black dot. Character width is orientation dependent.

The legal range for character width is 1 to 16384 dots.

Character Height (UI)

Character Height specifies the height of the character in dots on the physical coordinate system. Character height is orientation dependent.

The legal range for character height is 1 to 16384 dots.

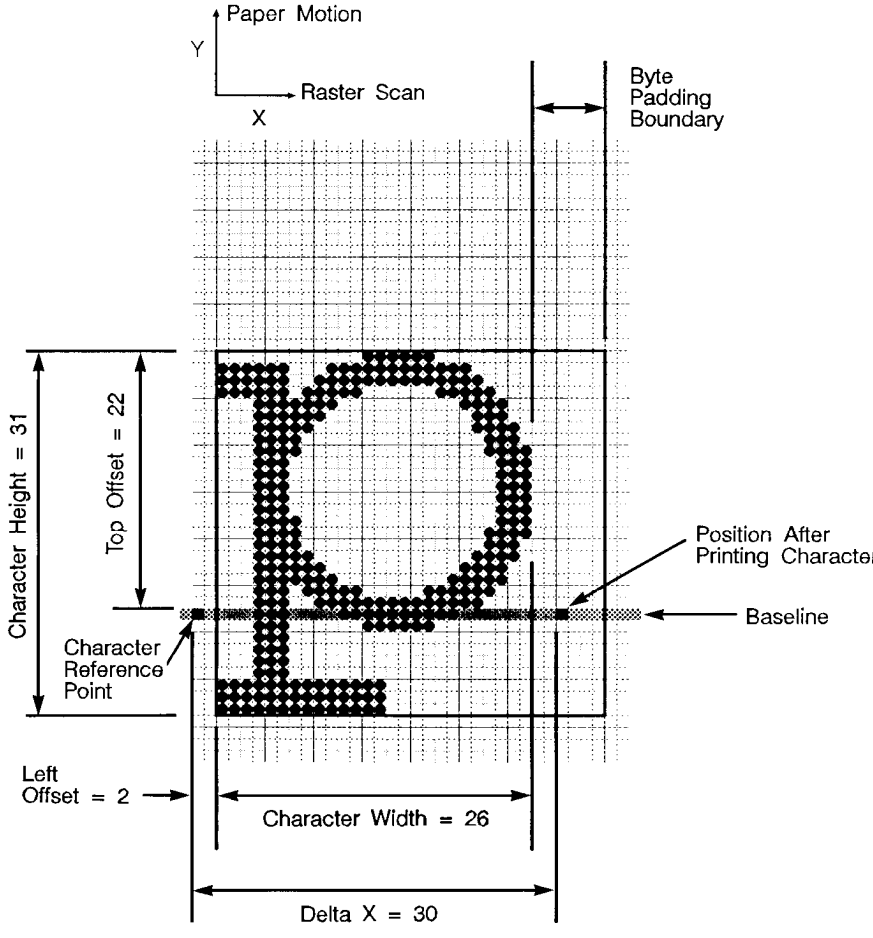
Delta X (SI)

Delta X specifies the number of quarter dots (radix dots) by which the horizontal position within the logical page coordinate system is incremented after printing the character. This value is only used by the printer when the font is proportionally spaced.

The legal range for delta X is –32768 to 32767 quarter units.

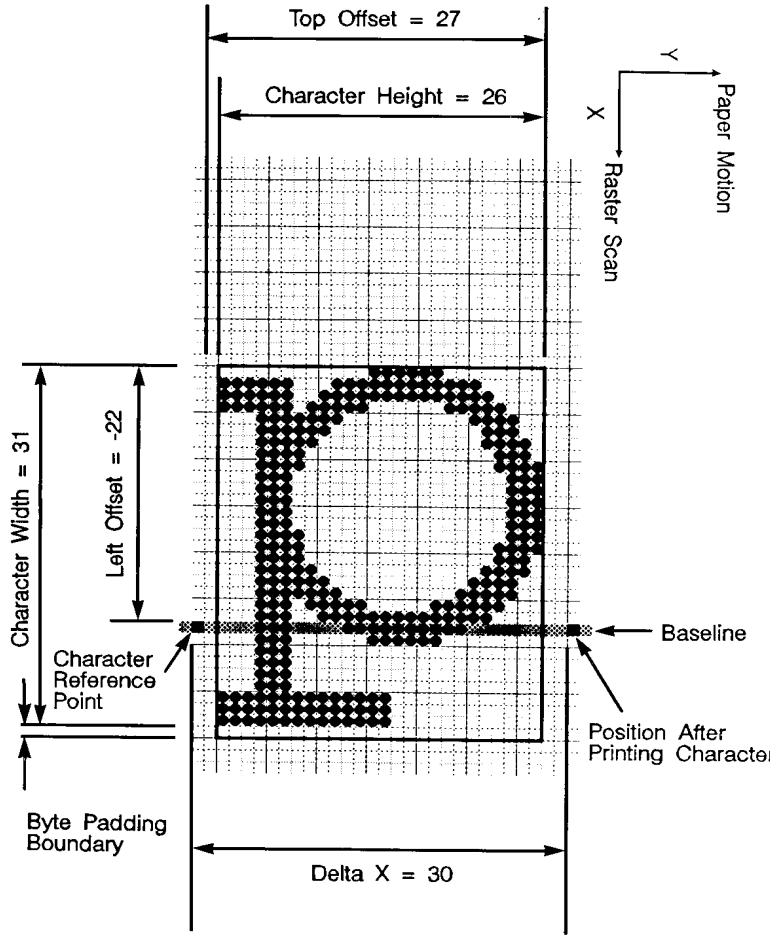
Character Data

Character data is a string of bytes containing the dot-per-bit image of the character or a run-length encoding with line repetition compressed format.



All values are in dots.

Figure 11-6 Portrait Character Example



All values are in dots.

Figure 11-7 Landscape Character Example

Character Descriptor and Data Format for Intellifont Scalable Fonts

The character header contains a block of bytes that identify character outline data. Table 11-40 and Table 11-41 show the format of the Intellifont scalable character descriptor and data

.

Table 11-40 Intellifont Scalable Character Descriptor and Data Format

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (10)	Continuation (0)[1]
2	Descriptor Size	Class (3)
4	Contour Character Data: (in bytes) ⋮ see Table 11-41 for Contour Character Data	
#–2	Reserved (0)	Checksum[2]

1. Continuation is supported for classes 1, 2, 3 and 15 only.
2. These bytes appear only on the last continuation.

Table 11-41 Intellifont Scalable Contour Data Format

Byte	15 (MSB) 8	7 (LSB) 0
4	Contour Data Size	
6	Metric Data Offset	
8	Character Intellifont Data Offset	
10	Contour Tree Offset	
12	XY Data Offset	
14		
	Metric Data ⋮	
	Character Intellifont Data ⋮	

Table 11-41 Intellifont Scalable Contour Data Format

	Contour Tree Data ⋮
	XY Coordinate Data ⋮

Table 11-42 Intellifont Scalable Character Descriptors/Data Continuation Block

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (10)	Continuation (1)[1]
2	Contour Character Data, resumed: (in bytes, see Table 11-41. ⋮	
#–2	Reserved	Checksum[2]

1. Continuation is supported for Intellifont scalable fonts for class 3 only.
2. This byte appears only on the last continuation.

Table 11-43 Intellifont Scalable Compound Character Descriptor and Data Format

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (10)	Continuation (0)
2	Descriptor Size	Class (4)
4	Compound Character Escapement	
6	Number of Components	
8	Component List ⋮ see Table 11-46 for Component List Data	
–2	Reserved	Checksum

Format (UB)

This is the first byte of every character data block header. It specifies the format of the character descriptor and data. The format number for Intellifont scalable fonts is 10.

Table 11-44

Value	Format
4*	LaserJet Family (Raster)
10	Intellifont Scalable
15*	TrueType Scalable
* These are described elsewhere in this chapter.	

If the format number is different from that expected by the device, the character is discarded.

Continuation (B)

This is the second (and last) byte of every character data block header. It specifies whether the following data is the first (0) data block of a new character definition, or a continuation (1) block for a character definition which has already been received by the printer. Because the value field in a Character Definition command is limited to 32767 bytes, characters whose byte count exceed this must be sent in two or more blocks. Table 11-42 shows the continuation block for an Intellifont Scalable font.

Descriptor Size (UB)

This is the first byte of the character descriptor. It specifies the size of the character descriptor in bytes. The typical descriptor size for Intellifont scalable fonts is 2.

Class (UB)

Specifies the format of the character data. For Intellifont scalable fonts values 3 and 4 are used, as described below.

Table 11-45

Value	Class
1*	Bitmap
2*	Compressed Bitmap
3	Contour (Intellifont Scalable)
4	Compound Contour (Intellifont Scalable)
15*	TrueType Scalable
* These are described elsewhere in this chapter.	

Class 3 -Intellifont Scalable Character Contour Data

Class 3 is for Intellifont scalable contour character data. The contour character data is organized as described in Table 11-41. Bytes 0-3 contain the character descriptor.

Class 4 - Intellifont Scalable Compound Character Data

A class 4 character is a compound character and composition data follows. The composition data is organized as described in Table 11-43. The compound descriptor allows combining two different characters to produce a single compound character.

Contour Data Size (UI)

The size of the contour data including the size of this field. For a detailed description of this field, refer to *Intellifont Scalable Typeface Format*.

Metric Data Offset (SI)

The offset to the Metric Data relative to the address of the Contour Data Size field.

Character Intellifont Data Offset (SI)

The offset to the Character Intellifont Data relative to the address of the Contour Data Size field.

Contour Tree Offset (SI)

The offset to the contour Tree Data relative to the address of the Contour Data Size field.

XY Data Offset (SI)

The offset to the XY data relative to the address of the Contour Data Size field.

Metric Data

For information about Metric Data refer to the *Intellifont Scalable Typeface Format* document.

Character Intellifont Scalable Data

For information about Character Intellifont Scalable Data, refer to *Intellifont Scalable Typeface Format*.

Contour Tree Data

For information about Contour Tree Data, refer to *Intellifont Scalable Typeface Format*.

XY Coordinate Data

For information about XY Coordinate Data, refer to *Intellifont Scalable Typeface Format*.

Note

For information on obtaining the *Intellifont Scalable Typeface Format* document, refer to *Related Documents* in the front of this manual.

Checksum

This is a checksum of all the contour character data. The checksum value is contained only in the last character data block.

Compound Character Escapement (SI)

The escapement in design units of a compound character.

Number of Components (UB)

The number of components of a compound character.

Component List

This is a list of component descriptions. The list contains Number of Components elements. Each component descriptor consists of 6 bytes as described in Table 11-46 below.

Table 11-46 Component Descriptor

Byte	15 (MSB) 8	7 (LSB) 0
0	Character Code	
2	X Offset	
4	Y Offset	

The Character Code is the character code number of a component of a compound character. X-offset is the offset of that component from the reference point (origin) in the x direction in design units. Y-offset is the offset in the y direction of a component from the reference point (origin) in design units.

Note The character code may be greater than the last code of the symbol set that is implied by the font type since a compound character can include components that are not part of the symbol set.

Character Descriptor and Data Format for TrueType Fonts

Table 11-47 shows the format of the TrueType character descriptor and data when a **continuation block is not required**. Table 11-49 (next page) shows the format of the TrueType character descriptor and data with **multiple character data blocks**.

Table 11-47 TrueType Character Descriptor (no continuation block required)

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (15)	Continuation (0)
2	Descriptor Size	Class (15)
4	*[additional descriptor data may be inserted here]*	
2 + Desc Size	Character Data Size	
4 + Desc Size	Glyph ID	
6 + Desc Size	TrueType Glyph Data ⋮	
# – 2	Reserved	Checksum
# = Character data block size as defined in Character Definition command.		

Table 11-48

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (15)	Continuation (0)
2	Descriptor Size	Class (15)
4	*[additional descriptor data may be inserted here]*	
2 + Desc Size	Character Data Size	
4 + Desc Size	Glyph ID	

Table 11-48 (continued)

6 + Desc Size	beginning of TrueType Glyph Data ⋮

Table 11-49 TrueType Character Descriptor (multiple character data blocks)

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (15)	Continuation (1)
2	conclusion of TrueType Glyph Data ⋮	
# – 2	Reserved	Checksum
# = Character data block size as defined in Character Definition command.		

Format (UB)

This is the first byte of every character data block header. It specifies the format for character downloading. The number 15 designates the TrueType character format.

Table 11-50

Value	Format
4*	LaserJet Family (Raster)
10*	Intellifont Scalable
15	TrueType Scalable
* These are described elsewhere in this chapter.	

Continuation (B)

This is the second (and last) byte of every character data block header. It specifies whether the following data is the first (0) data block of a new character definition, or a continuation (1) block for a character definition which has already been received by the printer. Because the value field in a Character Definition command is limited to 32767 bytes, characters whose byte count exceed this must be sent in two or more blocks.

Descriptor Size (UB)

This is the first byte of the character descriptor. It specifies the size of the character descriptor in bytes. The character descriptor includes everything that is both after the continuation byte and prior to the Character Data Size field. The TrueType character descriptor includes this Descriptor Size byte plus the following Class byte. The minimum value for Descriptor Size is therefore 2. (Additional descriptor information, if any, can be added immediately after the Class byte.)

Class (UB)

This field is used to distinguish different character data types within a given character format. (For instance, among Intellifont characters of format 10, the Class byte is used to distinguish compound characters from simple Intellifont contour characters.) All TrueType scalable characters are handed to the TrueType font scaler in the same format, consequently, the Class byte does not provide vital new information. For TrueType, set the Class value to 15.

Table 11-51

Value	Class
1*	Bitmap
2*	Compressed Bitmap
3*	Contour (Intellifont Scalable)
4*	Compound Contour (Intellifont Scalable)
15	TrueType Scalable
* These are described elsewhere in this chapter.	

Character Data Size (UI)

The value of the Character Data Size should equal the sum of the sizes of the Character Data Size, Glyph ID, and TrueType Glyph Data fields. This value alerts the PCL interpreter when a continuation block is needed. The minimum possible value is 4. The value of Character Data Size plus Descriptor Size plus 4 (for the Format, Continuation, Reserved and Checksum bytes) will never be less than the value # given in the character download command. If the sum is exactly equal to #, then no continuation block is to be expected for the given character. However, if the sum exceeds #, then a continuation block is needed. A condition for the validity of a downloaded scalable TrueType character is that the sum of the # values for all of that character's data blocks equals the sum of the Descriptor Size and Character Data Size and 2 (for Reserved and Checksum), plus 2 times the number of character data blocks (for Format and Continuation bytes).

Glyph ID (UI)

This field is used by the TrueType font scaler as an ID number for the glyph data associated with the given character.

TrueType Glyph Data

This field contains the data segment associated with the given character as found in the **glyf** table of the original TrueType font file. See the description in *True Type Font Files*.

Checksum (UB)

The value of this byte, when added to the sum of all of the bytes in the Character Data Size, Glyph ID, and TrueType Glyph Data fields, should equal 0 in modulo 256 arithmetic. The Checksum is found only in the last character data block associated with a given character.

Character Definition Examples

Bitmap Portrait Character Example

To download a bitmap character descriptor and data for a portrait, 10 Pitch, 12 point, upright medium, Courier lower-case “p”, send:

$^{E}_{C}$***c112E** (112 is the decimal character code for an ASCII lower-case “p”)

$^{E}_{C}$**(s140W** [character descriptor and data]

Note

Notice that the **140** appearing in the Character Definition Command accounts for 2 bytes of the Character Data Block Header, 14 bytes of Character Descriptor, and 124 bytes of Character Data. Since the Character Width is 26 dots, 4 bytes are needed per raster row. Also, since the Character Height is 31 dots, 124 bytes of Character Data for a Class 1 character is needed (4 × 31 = 124). No continuation block is to be expected.

Table 11-52 Character Format, Continuation, and Descriptor

FIELD NAME	VALUE	DESCRIPTION
Format	4	LaserJet Printer Family
Continuation	0	Not A Continuation Record
Descriptor Size:	14	Bitmap
Class:	1	Normal Raster
Orientation:	0	Portrait
Left Offset:	2	dots
Top Offset:	22	dots
Character Width:	26	dots
Character Height:	31	dots
Delta X:	120	Quarter Dots (30 Dots)

Table 11-53 Portrait Character Data Example

Dot Row	Bit Map	Decimal Equivalent			
01	00000000 00001111 11000000 00000000	0	15	192	0
02	11111100 01111111 11111000 00000000	252	127	249	0
03	11111100 11111111 11111100 00000000	252	255	252	0
04	11111101 11110000 00111110 00000000	253	240	62	0
05	00011111 11000000 00001111 00000000	31	192	15	0
06	00011111 10000000 00000111 00000000	31	128	7	0
07	00011111 00000000 00000111 10000000	31	0	7	128
08	00011110 00000000 00000011 10000000	30	0	3	128
09	00011110 00000000 00000011 11000000	30	0	3	192
10	00011100 00000000 00000001 11000000	28	0	1	192
11	00011100 00000000 00000001 11000000	28	0	1	192
12	00011100 00000000 00000001 11000000	28	0	1	192
13	00011100 00000000 00000001 11000000	28	0	1	192
14	00011100 00000000 00000001 11000000	28	0	1	192
15	00011110 00000000 00000001 11000000	30	0	1	192
16	00011110 00000000 00000011 11000000	30	0	3	192
17	00011110 00000000 00000011 10000000	30	0	3	128
18	00011111 00000000 00000111 10000000	31	0	7	128
19	00011111 10000000 00001111 00000000	31	128	15	0
20	00011111 11000000 00011111 00000000	31	192	31	0
21	00011101 11110000 01111110 00000000	29	240	126	0
22	00011100 11111111 11111100 00000000	28	255	252	0
23	00011100 00111111 11110000 00000000	28	63	240	0
24	00011100 00001111 11000000 00000000	28	15	192	0

Table 11-53 Portrait Character Data Example (continued)

25	00011100 00000000 00000000 00000000	28	0	0	0
26	00011100 00000000 00000000 00000000	28	0	0	0
27	00011100 00000000 00000000 00000000	28	0	0	0
28	00011100 00000000 00000000 00000000	28	0	0	0
29	11111111 11111100 00000000 00000000	255	252	0	0
30	11111111 11111100 00000000 00000000	255	252	0	0
31	11111111 11111100 00000000 00000000	255	252	0	0

Bitmap Landscape Character Example

To download the character descriptor and data for a landscape, 10 pitch, 12 point, upright, medium, Courier lower-case “p”, send:

$^{E}_{C}$***c112E** (112 is the decimal character code for an ASCII lower-case “p”)

$^{E}_{C}$**(s120W** [character descriptor and data]

Table 11-54 Character Format, Continuation and Descriptor

FIELD NAME	VALUE	DESCRIPTION
Format	4	LaserJet Printer Family
Continuation	0	Not A Continuation Record
Descriptor Size:	14	Bitmap
Class:	1	Normal Raster
Orientation:	1	Landscape
Left Offset:	-22	dots
Top Offset:	27	dots
Character Width:	31	dots
Character Height:	26	dots
Delta X:	120	Quarter Dots (30 dots)

Table 11-55 Landscape Character Data Example

Dot Row	Bit Map	Decimal Equivalent			
01	00000000 01111110 00000000 00000000	0	126	0	0
02	00000011 11111111 11000000 00000000	3	255	192	0
03	00001111 11111111 11110000 00000000	15	255	240	0
04	00011111 10000001 11111000 00000000	31	129	248	0
05	00111110 00000000 01111100 00000000	62	0	124	0
06	00111000 00000000 00111100 00000000	56	0	124	0
07	01111000 00000000 00011110 00000000	120	0	30	0
08	01110000 00000000 00001110 00000000	112	0	14	0
09	11100000 00000000 00001111 00000000	224	0	15	0
10	11100000 00000000 00000111 00000000	224	0	7	0
11	11100000 00000000 00000111 00000000	224	0	7	0
12	11100000 00000000 00000111 00000000	224	0	7	0
13	11100000 00000000 00000111 00001110	224	0	7	14
14	11100000 00000000 00000111 00001110	224	0	7	14
15	01110000 00000000 00001110 00001110	112	0	14	14
16	01110000 00000000 00001110 00001110	112	0	14	14
17	00111000 00000000 00011100 00001110	56	0	28	14
18	00111100 00000000 00111100 00001110	60	0	60	14
19	00011110 00000000 01111000 00001110	30	0	120	14
20	00001111 10000001 11110000 00001110	15	129	250	14
21	01111111 11111111 11111111 11111110	127	255	255	240
22	01111111 11111111 11111111 11111110	127	255	255	240
23	01111111 11111111 11111111 11111110	127	255	255	240
24	01110000 00000000 00000000 00001110	112	0	0	14

Table 11-55 Landscape Character Data Example (continued)

25	01110000 00000000 00000000 00001110	112	0	0	14
26	01110000 00000000 00000000 00001110	112	0	0	14