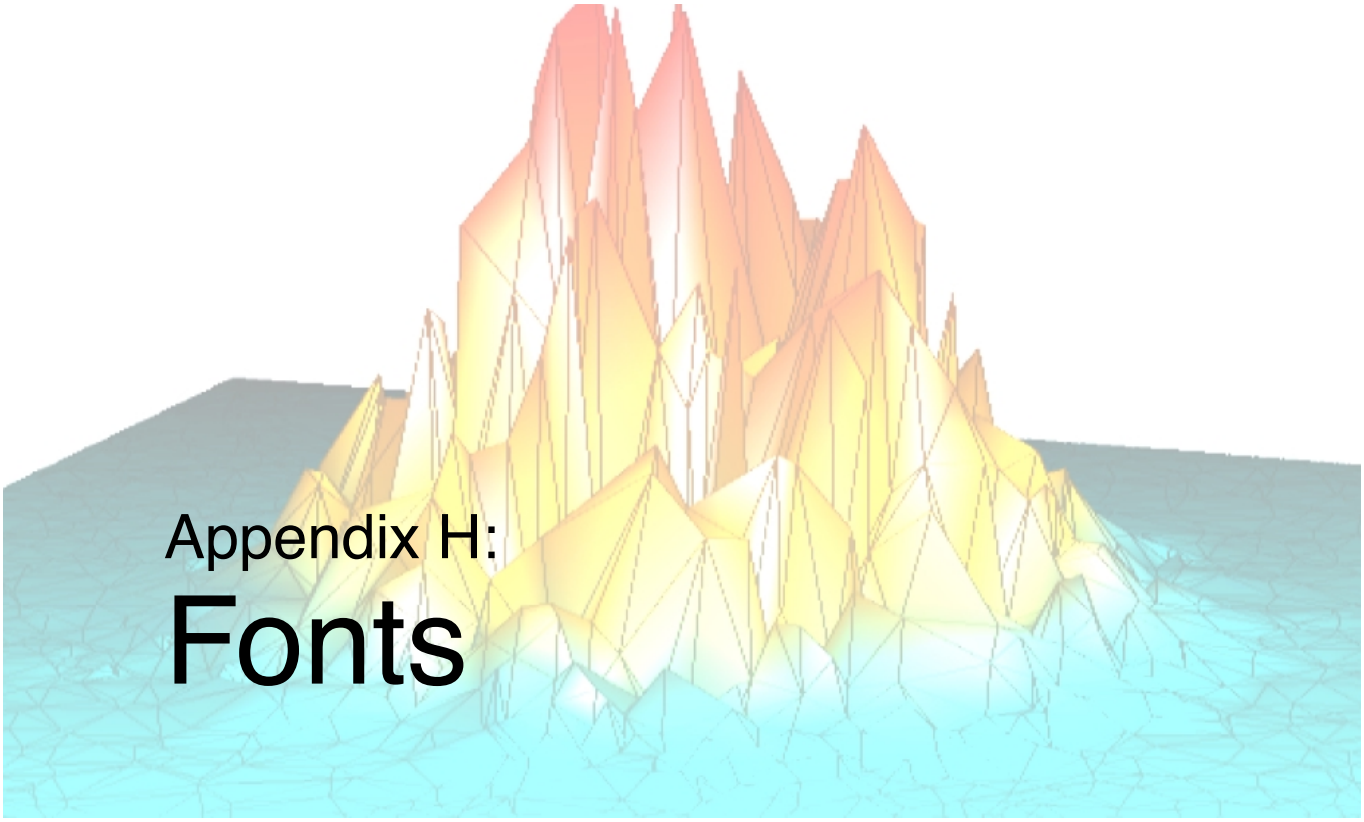


Appendix H: Fonts

The following topics are covered in this appendix:

Overview

IDL uses three font systems for writing characters on the graphics device: Hershey (vector) fonts, TrueType (outline) fonts, and device (hardware) fonts. This chapter describes each of the three types of fonts, discusses when to use each type, and explains how to use fonts when creating graphical output in IDL.

Vector-drawn fonts, also referred to as *Hershey fonts*, are drawn as lines. They are device-independent (within the limits of device resolution). All vector fonts included with IDL are guaranteed to be available in any IDL installation. See "About Vector Fonts" on page 2474 for additional details.

TrueType fonts, also referred to here as *outline fonts*, are drawn as character outlines, which are filled when displayed. They are largely device-independent, but do have some device-dependent characteristics. Four TrueType font families are included with IDL; these fonts should display in a similar way on any IDL platform. TrueType font support for IDL Object Graphics was introduced in IDL version 5.0 and support in IDL Direct Graphics was introduced in IDL version 5.1. See "About TrueType Fonts" on page 2477 for additional details.

Device fonts, also referred to as *hardware fonts*, rely on character-display hardware or software built in to a specific display device. Device fonts, necessarily, are device-dependent and differ from platform to platform and display device to display device. See "About Device Fonts" on page 2482 for additional details.

Fonts in IDL Direct vs. Object Graphics

This volume deals almost exclusively with IDL Direct Graphics. However, the vector and TrueType font systems described here are also available in the IDL Object Graphics system, described in *Using IDL*.

IDL Direct Graphics

When generating characters for Direct Graphics plots, IDL uses the font system specified by the value of the system variable !P.FONT. The normal default for this variable is -1, which specifies that the built-in, vector-drawn (Hershey) fonts should be used. Setting !P.FONT equal to 1 specifies that TrueType fonts should be used. Setting !P.FONT equal to zero specifies that fonts supplied by the graphics device should be used.

The setting of the IDL system variable !P.FONT can be overridden for a single IDL Direct Graphics routine (AXIS, CONTOUR, PLOT, SHADE_SURF, SURFACE, or XYOUTS) by setting the FONT keyword equal to -1, 0, or 1.

Once a font system has been selected, an individual font can be chosen either via a formatting command embedded in a text string as described in "Embedded Formatting Commands" on page 2491, or by setting the value of the FONT keyword to the DEVICE routine (see "FONT" on page 2326).

IDL Object Graphics

IDL Object Graphics can use the vector and TrueType font systems. See *Using IDL* for more information on using fonts with Object Graphics. Any TrueType fonts you add to your IDL installation as described in "About TrueType Fonts" on page 2477 will also be available to the Object Graphics system.

About Vector Fonts

The vector fonts used by IDL were digitized by Dr. A.V. Hershey of the Naval Weapons Laboratory. Characters in the vector fonts are stored as equations, and can be scaled and rotated in three dimensions. They are drawn as lines on the current graphics device, and are displayed quickly and efficiently by IDL. The vector fonts are built into IDL itself, and are always available.

All the available fonts are illustrated in "Vector Font Samples" on page 2501. The default vector font (Font 3, Simplex Roman) is in effect if no font changes have been made.

Using Vector Fonts

To use the vector font system with IDL Direct Graphics, either set the value of the IDL system variable !P.FONT equal to -1 (negative one), or set the FONT keyword of one of the Direct Graphics routines equal to -1. The vector font system is the default font system for IDL.

Once the vector font system is selected, use an embedded formatting command to select a vector font (or fonts) for each string. (See "Embedded Formatting Commands" on page 2491 for details on embedded formatting commands.) The font selected "sticks" from string to string; that is, if you change fonts in one string, future strings will use the new font until you change it again or exit IDL.

For example, to use the Duplex Roman vector font for the title of a plot, you would use a command that looks like this:

```
PLOT, mydata, TITLE="!5Title of my plot"
```

Consult *Using IDL* for details on using the vector font system with IDL Object Graphics.

Specifying Font Size

To specify the size of a vector font, use the SET_CHARACTER_SIZE keyword to the DEVICE procedure. The SET_CHARACTER_SIZE keyword takes a two-element vector as its argument. The first element specifies the width of the "average"

character in the font (in pixels) and calculates a scaling factor that determines the height of the characters. (It is not important what the “average” character is; it is used only to calculate a scaling factor that will be applied to all of the characters in the font.) The second element of the vector specifies the number of pixels between baselines of lines of text.

The ratio of the “average” character’s height to its width differs from font to font, so specifying the same value [*x*, *y*] to the SET_CHARACTER_SIZE keyword may produce characters of different sizes in different fonts.

Note

While the first element of the vector specified to SET_CHARACTER_SIZE is technically a width, it is important to note that the width value has no effect on the widths of individual characters in the font. The width value is used only to calculate the appropriate scaling factor for the font.

For example, the following IDL commands display the word “Hello There” on the screen, in letters based on an “average” character that is 70 pixels wide, with 90 pixels between lines:

```
DEVICE, SET_CHARACTER_SIZE=[70,90]
XYOUTS, 0.1, 0.5, 'Hello!CThere'
```

You can also use the CHARSIZE keyword to the graphics routines or the CHARSIZE field of the !P System Variable to change the size of characters to a multiple of the size of the currently-selected character size. For example, to create characters one half the size of the current character size, you could use the following command:

```
XYOUTS, 0.1, 0.5, 'Hello!CThere', CHARSIZE=0.5
```

Note

Changing CHARSIZE adjusts both the character size and the space between lines.

ISO Latin 1 Encoding

The default font (Font 3, Simplex Roman) follows the ISO Latin 1 Encoding scheme and contains many international characters. The illustration of this font under “Vector Font Samples” on page 2501 can be used to find the octal codes for the special characters.

For example, suppose you want to display some text with an Angstrom symbol in it. Looking at the chart of font 3, we see that the Angstrom symbol has octal code 305.

Non-printable characters can be represented in IDL using octal or hexadecimal notation and the STRING function (see "Representing Non-Printable Characters" in Chapter 3 of *Building IDL Applications* for details). So the Angstrom can be printed by inserting a `STRING("305B)` character in our text string as follows:

```
XYOUTS,.1, .5, 'Here is an Angstrom symbol: ' + STRING("305B), $
    /NORM, CHARSIZE=3
```

Customizing the Vector Fonts

The EFONT procedure is a widget application that allows you to edit the Hershey fonts and save the results. Use this routine to add special characters or completely new, custom fonts to the Hershey fonts.

About TrueType Fonts

A TrueType Font

Beginning with version 5.2, five TrueType font families are included with IDL. The fonts included are:

Font Family	Italic	Bold	BoldItalic
Courier	Courier Italic	Courier Bold	Courier Bold Italic
Helvetica	Helvetica Italic	Helvetica Bold	Helvetica Bold Italic
Monospace Symbol			
Times	Times Italic	Times Bold	Times Bold Italic
Symbol			

Table H-1: TrueType font names

When TrueType fonts are rendered on an IDL graphics device or destination object, the font outlines are first scaled to the proper size. After scaling, IDL converts the character outline information to a set of polygons using a triangulation algorithm. When text in a TrueType font is displayed, IDL is actually drawing a set of polygons calculated from the font information. This process has two side effects:

1. Computation time is used to triangulate and create the polygons. This means that you may notice a slight delay the first time you use text in a particular font and size. Once the polygons have been created, the information is cached by IDL and there is no need to re-triangulate each time text is displayed. Subsequent uses of the same font and size happen quickly.
2. Because the TrueType font outlines are converted into polygons, you may notice some chunkiness in the displayed characters, especially at small point sizes. The smoothness of the characters will vary with the quality of the TrueType font you are using, the point size, and the general smoothness of the font outlines.

Using TrueType Fonts

To use the TrueType font system with IDL Direct Graphics, either set the value of the IDL system variable !P.FONT equal to 1 (one), or set the FONT keyword to on one of the Direct Graphics routines equal to 1.

Once the TrueType font system is selected, use the SET_FONT keyword to the DEVICE routine to select the font to use. The value of the SET_FONT keyword is a *font name string*. The font name is the name by which IDL knows the font; the names of the TrueType fonts included with IDL are listed under "About TrueType Fonts" on page 2477. Finally, specify the TT_FONT keyword in the call to the DEVICE procedure. For example, to use Helvetica Bold Italic, use the following statement:

```
DEVICE, SET_FONT='Helvetica Bold Italic', /TT_FONT
```

To use Times Roman Regular:

```
DEVICE, SET_FONT='Times', /TT_FONT
```

IDL's default TrueType font is 12 point Helvetica regular.

Specifying Font Size

To specify the size of a TrueType font, use the SET_CHARACTER_SIZE keyword to the DEVICE procedure. The SET_CHARACTER_SIZE keyword takes a two-element vector as its argument. The first element specifies the width of the "average" character in the font (in pixels) and calculates a scaling factor that determines the height of the characters. (It is not important what the "average" character is; it is used only to calculate a scaling factor that will be applied to all of the characters in the font.) The second element of the vector specifies the number of pixels between baselines of lines of text.

The ratio of the "average" character's height to its width differs from font to font, so specifying the same value [*x, y*] to the SET_CHARACTER_SIZE keyword may produce characters of different sizes in different fonts.

Note

While the first element of the vector specified to SET_CHARACTER_SIZE is technically a width, it is important to note that the width value has no effect on the widths of individual characters in the font. The width value is used only to calculate the appropriate scaling factor for the font.

For example, the following IDL commands display the word “Hello There” on the screen in Helvetica Bold, in letters based on an “average” character that is 70 pixels wide, with 90 pixels between lines:

```
DEVICE, FONT='Helvetica Bold', /TT_FONT,
SET_CHARACTER_SIZE=[70,90]
XYOUTS, 0.1, 0.5, 'Hello!CThere'
```

You can also use the CHARSIZE keyword to the graphics routines or the CHARSIZE field of the !P System Variable to change the size of characters to a multiple of the size of the currently-selected character size. For example, to create characters one half the size of the current character size, you could use the following command:

```
XYOUTS, 0.1, 0.5, 'Hello!CThere', CHARSIZE=0.5
```

Note that changing the CHARSIZE adjusts both the character size and the space between lines.

Using Embedded Formatting Commands

Embedded formatting commands allow you to position text and change fonts within a single line of text. A subset of the embedded formatting commands available for use with the vector fonts are also available when using the TrueType font system. See “Embedded Formatting Commands” on page 2491 for a list of in-line formatting commands.

IDL TrueType Font Resource Files

The TrueType font system relies on a resource file named `ttfont.map`, located in the `resource/fonts/tt` subdirectory of the IDL directory. The format of the ttfont.map file is:

```
FontName    FileName    DirectGraphicsScale    ObjectGraphicsScale
```

where the fields in each row must be separated by white space (spaces and/or tabs). The fields contain the following information

The *Fontname* field contains the name that would be used for the SET_FONT keywords to the DEVICE routine.

The *Filename* field contains the name of the TrueType font file. On UNIX and VMS platforms, IDL will search for the file specified in the *FileName* field in the current directory (that is, in the `resource/fonts/tt` subdirectory of the IDL directory) if a bare filename is provided, or it will look for the file in the location specified by the fully-qualified file name if a complete path is provided. Because different platforms

use different path-specification syntax, we recommend that you place any TrueType font files you wish to add to the `ttfont.map` file in the `resource/fonts/tt` subdirectory of the IDL directory. On Macintosh and Windows platforms, this entry may be '*', in which case the font will be loaded from the operating system font list, but that the following two scale entries will be honored.

The *DirectGraphicsScale* field contains a correction factor that will be applied when choosing a scale factor for the glyphs prior to being rendered on a Direct Graphics device. If you want the tallest character in the font to fit exactly within the vertical dimension of the device's current character size (as set via the SET_CHARACTER_SIZE keyword to the DEVICE procedure), set the scale factor equal to 1.0. Change the scale factor to a smaller number to scale a smaller portion of the tallest character into the character size.

For example, suppose the tallest character in your font is "Å". Setting the scale factor to 1.0 will scale this character to fit the current character size, and then apply the same scaling to all characters in the font. As a result, the letter "M" will fill only approximately 85% of the full height of the character size. To scale the font such that the height of the "M" fills the vertical dimension of the character size, you would include the value 0.85 in the scale field of the `ttfont.map` file.

The *ObjectGraphicsScale* field contains a correction factor that will be applied when choosing a scale factor for the glyphs prior to being rendered on a Direct Graphics device. (This field works just like the *DirectGraphicsScale* field.) This scale factor should be set to 1.0 if the maximum ascent among all glyphs within a given font is to fit exactly within the font size (as set via the SIZE property to the IDLgrFont object).

Adding Your Own Fonts

To add a your own font to the list of fonts known to IDL, use a text editor to edit the `ttfont.map` file, adding the *FontName*, *FileName*, *DirectGraphicsScale*, and *ObjectGraphicsScale* fields for your font. You will need to restart IDL for the changes to the `ttfont.map` file to take effect. On Windows and Macintosh systems, you can use fonts that are not mentioned in the `ttfont.map` file, as long as they are installed in the Fonts control panel or Font folder, as described below.

Warning

If you choose to modify the `ttfont.map` file, be sure to keep a backup copy of the original file so you can restore the defaults if necessary. Note also that applications that use text may appear different on different platforms if the scale entries in the `ttfont.map` file have been altered.

Where IDL Searches for Fonts

The TrueType font files included with IDL are located in the `resource/fonts/tt` subdirectory of the IDL directory. When attempting to resolve a font name (specified via the FONT keyword to the DEVICE procedure), IDL will look in the `ttfont.map` file first. If it fails to find the specified font file in the `ttfont.map` file, it will search for the font file in the following locations:

UNIX and VMS

No further search will be performed. If the specified font is not included in the `ttfont.map` file, IDL will substitute Helvetica.

Microsoft Windows

If the specified font is not included in the `ttfont.map` file, IDL will search the list of fonts installed in the system (the fonts installed in the Font control panel). If the specified font is not found, IDL will substitute Helvetica.

Macintosh

If the specified font is not included in the `ttfont.map` file, IDL will search the list of fonts installed in the system (the fonts installed 1in the Fonts subfolder of the System folder). If the specified font is not found, IDL will substitute Helvetica.

About Device Fonts

Device, or hardware, fonts are fonts that are provided directly by your system's hardware or by software other than IDL. In past releases of IDL, we have used the term "*hardware fonts*" extensively to discuss these types of fonts. This is because in the early days of IDL, computer displays were either text-only terminals or dedicated graphics display devices such as plotters or Tektronix graphics terminals. These graphics displays generally came with a set of fonts built-in; when IDL asked the device to display characters in a built-in font, it was making a request to the hardware to display those characters.

As computer displays have become more sophisticated, the concept of fonts provided "by the hardware" has expanded to include fonts provided by the computer's operating system, or by font-management software. For example, many computers now use font management software like Adobe Type Manager to manage the fonts made available by the operating system to all applications. We use the term "device font" to refer to a font that is available to one of IDL's graphics devices from a source *other than IDL itself*. (In this case, a "graphics device" can be either a Direct Graphics device as specified by the DEVICE routine or an Object Graphics "destination" such as a window or a printer.) While device fonts may include fonts only available because a particular piece of hardware knows how to draw characters in that font (a pen plotter is an example of a device that may still have its own special fonts), in most cases device fonts are fonts supplied by the operating system to any application that may want to use them.

Which Device Fonts Are Available?

To determine which device fonts are available on your system and the exact font strings to specify for each, use the GET_FONTNAMES keyword to the DEVICE procedure. You can also use an operating system specific method to determine which fonts are available:

UNIX and VMS

On most systems, the `xlsfonts` utility displays a list of fonts available to the operating system.

Microsoft Windows

Fonts available to the system are displayed in the Fonts control panel. You may also have other fonts available if you use font-management software such as Adobe Type Manager.

Macintosh

Fonts available to the system are displayed in the Fonts folder in the System folder. You may also have other fonts available if you use font-management software such as Adobe Type Manager.

Using Device Fonts

To use the Device font system with IDL Direct Graphics, either set the value of the IDL system variable !P.FONT equal to 0 (zero), or set the FONT keyword to on one of the Direct Graphics routines equal to 0.

Once the Device font system is selected, use the SET_FONT keyword to the DEVICE routine to select the font to use. Because device fonts are specified differently on different platforms, the syntax of the *fontname* string depends on which platform you are using.

UNIX and VMS

Usually, the window system provides a directory of font files that can be used by all applications. List the contents of that directory to find the fonts available on your system. The size of the font selected also affects the size of vector drawn text. On some machines, fonts are kept in subdirectories of `/usr/lib/X11/fonts`. You can use the `xlsfonts` command to list available X Windows fonts.

For example, to select the font 8X13:

```
!P.FONT = 0
DEVICE, SET_FONT = '8X13'
```

Microsoft Windows

The SET_FONT keyword should be set to a string with the following form:

```
DEVICE, SET_FONT="font*modifier1*modifier2*...modifiern"
```

where the asterisk (*) acts as a delimiter between the font's name (*font*) and any modifiers. The string is *not* case sensitive. Modifiers are simply "keywords" that change aspects of the selected font. Valid modifiers are:

- For font weight: THIN, LIGHT, BOLD, HEAVY

- For font quality: DRAFT, PROOF
- For font pitch: FIXED, VARIABLE
- For font angle: ITALIC
- For strikeout text: STRIKEOUT
- For underlined text: UNDERLINE
- For font size: Any number is interpreted as the font height in pixels.

For example, if you have Garamond installed as one of your Windows fonts, you could select 24-pixel cell height Garamond italic as the font to use in plotting. The following commands tell IDL to use hardware fonts, change the font, and then make a simple plot:

```
!P.FONT = 0
DEVICE, SET_FONT = "GARAMOND*ITALIC*24"
PLOT, FINDGEN(10), TITLE = "IDL Plot"
```

This feature is compatible with TrueType and Adobe Type Manager (and, possibly, other type scaling programs for Windows). If you have TrueType or ATM installed, the TrueType or PostScript outline fonts are used so that text looks good at any size.

Macintosh

The SET_FONT keyword should be set to a string with the following form:

```
DEVICE, SET_FONT="font*modifier1*modifier2*...modifiern"
```

where the asterisk (*) acts as a delimiter between the font's name (*font*) and any modifiers. The string is *not* case sensitive. Modifiers are simply "keywords" that change aspects of the selected font. Valid modifiers are:

- For font weight: BOLD
- For font angle: ITALIC
- For font width: CONDENSED, EXTENDED
- For outlined text: OUTLINE, SHADOW
- For underlined text: UNDERLINE
- For font size: Any number is interpreted as the font size, in points.

For example, if you have Garamond installed, you could select 24-point Garamond italic as the font to use in plotting. The following commands tell IDL to use hardware fonts, change the font, and then make a simple plot:

```
IDL> !P.FONT = 0
IDL> DEVICE, SET_FONT = "GARAMOND*ITALIC*24"
IDL> PLOT, FINDGEN(10), TITLE = "IDL Plot"
```

Fonts and the PostScript Device

A special set of device fonts are available when the current Direct Graphics device is `PS` (PostScript). IDL includes font metric information for 35 standard PostScript fonts, and can create PostScript language files that include text in these fonts. (The 35 fonts known to IDL are listed in the following table; they the standard fonts included in memory in the vast majority of modern PostScript printers.) The PostScript font metric files (`*.afm` files) are located in the `resource/fonts/ps` subdirectory of the IDL directory.

AvantGarde-Book	Helvetica-Narrow-Oblique
AvantGarde-BookOblique	Helvetica-Oblique
AvantGarde-Demi	NewCenturySchlbk-Bold
AvantGarde-DemiOblique	NewCenturySchlbk-BoldItalic
Bookman-Demi	NewCenturySchlbk-Italic
Bookman-DemiItalic	NewCenturySchlbk-Roman
Bookman-Light	Palatino-Bold
Bookman-LightItalic	Palatino-BoldItalic
Courier	Palatino-Italic
Courier-Bold	Palatino-Roman
Courier-BoldOblique	Symbol
Courier-Oblique	Times-Bold
Helvetica	Times-BoldItalic
Helvetica-Bold	Times-Italic
Helvetica-BoldOblique	Times-Roman
Helvetica-Narrow	ZapfChancery-MediumItalic
Helvetica-Narrow-Bold	ZapfDingats
Helvetica-Narrow-BoldOblique	

Table H-2: Names of Supported PostScript Fonts

Using PostScript Fonts

To use a PostScript font in your Direct Graphics output, you must first specify that IDL use the device font system, they switch to the PS device, then choose a font using the SET_FONT keyword to the DEVICE procedure.

The following IDL commands choose the correct font system, set the graphics device, select the font Palatino Roman, open a PostScript file to print to, plot a simple data set, and close the PostScript file:

```
!P.FONT = 0
SET_PLOT, 'PS'
DEVICE, SET_FONT = 'Palatino-Roman', FILE = 'testfile.ps'
PLOT, INDGEN(10), TITLE = 'My Palatino Title'
DEVICE, /CLOSE
```

Note

Subsequent PostScript output will continue to use the font Palatino Roman until you explicitly change the font again, or exit IDL.

You can also specify PostScript fonts using a set of keywords to the DEVICE procedure. The keyword combinations for the fonts included with IDL are listed in the following table.

PostScript Font	DEVICE Keywords
Courier	/COURIER
Courier Bold	/COURIER, /BOLD
Courier Oblique	/COURIER, /OBLIQUE
Courier Bold Oblique	/COURIER, /BOLD, /OBLIQUE
Helvetica	/HELVETICA
Helvetica Bold	/HELVETICA, /BOLD
Helvetica Oblique	/HELVETICA, /OBLIQUE
Helvetica Bold Oblique	/HELVETICA, /BOLD, /OBLIQUE
Helvetica Narrow	/HELVETICA, /NARROW
Helvetica Narrow Bold	/HELVETICA, /NARROW, /BOLD

Table H-3: The Standard 35 PostScript Fonts

PostScript Font	DEVICE Keywords
Helvetica Narrow Oblique	/HELVETICA, /NARROW, /OBLIQUE
Helvetica Narrow Bold Oblique	/HELVETICA, /NARROW, /BOLD, /OBLIQUE
ITC Avant Garde Gothic Book	/AVANTGARDE, /BOOK
ITC Avant Garde Gothic Book Oblique	/AVANTGARDE, /BOOK, /OBLIQUE
ITC Avant Garde Gothic Demi	/AVANTGARDE, /DEMI
ITC Avant Garde Gothic Demi Oblique	/AVANTGARDE, /DEMI, /OBLIQUE
ITC Bookman Demi	/BKMAN, /DEMI
ITC Bookman Demi Italic	/BKMAN, /DEMI, /ITALIC
ITC Bookman Light	/BKMAN, /LIGHT
ITC Bookman Light Italic	/BKMAN, /LIGHT, /ITALIC
ITC Zapf Chancery Medium Italic	/ZAPFCHANCERY, /MEDIUM, /ITALIC
ITC Zapf Dingbats	/ZAPFDINGBATS
New Century Schoolbook	/SCHOOLBOOK
New Century Schoolbook Bold	/SCHOOLBOOK, /BOLD
New Century Schoolbook Italic	/SCHOOLBOOK, /ITALIC
New Century Schoolbook Bold Italic	/SCHOOLBOOK, /BOLD, /ITALIC
Palatino	/PALATINO
Palatino Bold	/PALATINO, /BOLD
Palatino Italic	/PALATINO, /ITALIC
Palatino Bold Italic	/PALATINO, /BOLD, /ITALIC
Symbol	/SYMBOL
Times	/TIMES
Times Bold	/TIMES, /BOLD

Table H-3: The Standard 35 PostScript Fonts

PostScript Font	DEVICE Keywords
Times Italic	/TIMES, /ITALIC
Times Bold Italic	/TIMES, /ITALIC, /BOLD

Table H-3: The Standard 35 PostScript Fonts

For example to use the PostScript font Palatino Bold Italic, you could use either of the following DEVICE commands:

```
DEVICE, SET_FONT = 'Palatino*Bold*Italic'
DEVICE, /PALATINO, /BOLD, /ITALIC
```

Changing the PostScript Font Assigned to an Index

You can change the PostScript font assigned to a given font index using the FONT_INDEX keyword to the DEVICE procedure. Font indices and their use are discussed in "Embedded Formatting Commands" on page 2491.

Changing the font index assigned to a font can be useful when changing PostScript fonts in the middle of a text string. For example, the following statements map Palatino Bold Italic to font index 4, and then output text using that font and the Symbol font:

```
; Map the font selected by !4 to be PalatinoBoldItalic:
DEVICE, /PALATINO, /BOLD, /ITALIC, FONT_INDEX=4
; Output "Alpha :" in PalatinoBoldItalic followed by an
; Alpha character:
XYOUTS, .3, .5, /NORMAL, "!4Alpha: !9a", FONT=0, SIZE=5.0
```

Adding Your Own PostScript Fonts

Because the 35 PostScript fonts included with IDL are built in to a PostScript printer's memory, the IDL distribution includes only the font metric files, which provide positioning information. In addition, the `.afm` files used by IDL are specially processed to provide the information in a format IDL expects.

You can add your own PostScript fonts to the list of fonts known to IDL if you have access to the PostScript font file (usually named *font*`.pfb`) to load into your printer and to the *font*`.afm` file supplied by Adobe. You can convert the standard `.afm` file into a file IDL understands using the IDL routine PSAFM. Consult the file `README.TXT` in the `resource/fonts/ps` subdirectory of the IDL directory for details on adding PostScript fonts to your system.

Choosing a Font Type

Some of the issues involved in choosing between vector, TrueType, and device fonts are explained below.

Appearance

Vector-drawn characters are of medium quality, suitable for most uses. TrueType characters are of relatively high quality, although at some point sizes the triangulation process (described in "About TrueType Fonts" on page 2477) may cause characters to appear slightly jagged. The appearance of device characters varies from mediocre (characters found in many graphics terminals) to publication quality (PostScript).

Three-Dimensional Transformations

Vector or TrueType fonts should always be used with three-dimensional transformations. Both vector and TrueType characters pass through the same transformations as the rest of the plot, yielding a better looking plot. See "Three-Dimensional Graphics" in Chapter 12 of *Using IDL* for an example of vector-drawn characters with three-dimensional graphics. Device characters are not subject to the three-dimensional transforms.

Portability

The vector-drawn fonts work using any graphics device and look the same on every device (within the limitations of device resolution) on any system supported by IDL.

TrueType fonts are available only on the `X`, `WIN`, `MAC`, `PRINTER`, `PS`, and `Z` Direct Graphics devices, and in IDL's Object Graphics system. If you use only the fonts supplied with IDL, TrueType fonts also look the same on every supported device (again within the limits of the device resolution). If you use TrueType fonts other than those supplied with IDL, your font may not be installed on the system which runs your program. In this case, IDL will substitute a known font for the missing font.

The appearance, size, and availability of device fonts varies greatly from device to device. Many, if not most, of the positioning and font changing commands recognized by the vector-drawing routines are ignored when using device fonts. The exception to this rule is the Direct Graphics `PS` device; if you use one of the PostScript fonts supported by IDL, the PostScript output from the `PS` device will be identical between platforms.

Computational Time

Device fonts are generally rendered the most quickly, since the hardware device or operating system handles all computations and caching.

It takes more computer time to draw characters with line vectors and generally results in more input/output. However, this is not an important issue unless the plot contains a large number of characters or the transmission link to the device is very slow.

The initial triangulation step used when displaying TrueType fonts for the first time can be computationally expensive. However, since the font shapes are cached, subsequent uses of the same font are relatively speedy.

Flexibility

Vector-drawn fonts provide a great deal of flexibility. There are many different typefaces available, as shown in the tables at the end of this chapter. In addition, such fonts can be arbitrarily scaled, rotated, and transformed.

TrueType fonts support fewer embedded formatting commands than do the vector fonts, and cannot be scaled, rotated, or transformed.

The abilities of hardware-generated characters differ greatly between devices so it is not possible to make a blanket statement on when they should be used—the best font to use depends on the available hardware. In general, however, the vector or TrueType fonts are easier to use and often provide superior results to what is available from the hardware. See the discussion of the device you are using in Appendix B, "IDL Graphics Devices" for details on the hardware-generated characters provided by that device.

Print Quality

For producing publication-quality output, we recommend using either the TrueType font system or the Direct Graphics PS device and one of the PostScript fonts supported by IDL.

Embedded Formatting Commands

When you use vector, TrueType, and some device fonts, text strings can include embedded formatting commands that facilitate subscripting, superscripting, and equation formatting. The method used is similar to that developed by Grandle and Nystrom (1980). Embedded formatting commands are always introduced by the exclamation mark, (**!**). (The string "**!!**" is used to produce a literal exclamation mark.)

Note

Embedded formatting commands prefaced by the exclamation mark have no special significance for hardware-generated characters unless the ability is provided by the particular device in use. The IDL PostScript device driver accepts many of the standard embedded formatting commands, and is described here. If you wish to use hardware fonts with IDL Direct Graphics devices other than the PostScript device, consult the description of the device in Appendix B, "IDL Graphics Devices" before trying to use these commands with hardware characters.

You can determine whether embedded formatting commands are available for use with device fonts on your current graphics device by inspecting bit 12 of the *Flags* field of the !D System Variable. Use the IDL statement:

```
IF (!D.FLAGS AND 4096) NE 0 THEN PRINT, 'Bit is set.'
```

to determine whether bit 12 of the *Flags* field is set for the current graphics device.

Changing Fonts within a String

You can change fonts one or more times within a text string using the embedded font commands shown in the table below. The character following the exclamation mark can be either upper or lower case.

Examples of commands used to change fonts in mid-string are included in "Formatting Command Examples" on page 2494.

Command	Vector Font	TrueType Font	PostScript Font
!3	Simplex Roman (default)	Helvetica	Helvetica
!4	Simplex Greek	Helvetica Bold	Helvetica Bold
!5	Duplex Roman	Helvetica Italic	Helvetica Narrow
!6	Complex Roman	Helvetica Bold Italic	Helvetica Narrow Bold Oblique
!7	Complex Greek	Times	Times Roman
!8	Complex Italic	Times Italic	Times Bold Italic
!9	Math/special characters	Symbol	Symbol
!M	Math/special characters (change effective for one character only)	Symbol	Symbol
!10	Special characters	Symbol *	Zapf Dingbats
!11(!G)	Gothic English	Courier	Courier
!12(!W)	Simplex Script	Courier Italic	Courier Oblique
!13	Complex Script	Courier Bold	Palatino
!14	Gothic Italian	Courier Bold Italic	Palatino Italic
!15	Gothic German	Times Bold	Palatino Bold
!16	Cyrillic	Times Bold Italic	Palatino Bold Italic
!17	Triplex Roman	Helvetica *	Avant Garde Book
!18	Triplex Italic	Helvetica *	New Century Schoolbook
!19		Helvetica *	New Century Schoolbook Bold
!20	Miscellaneous	Helvetica *	Undefined User Font
!X	Revert to the entry font	Revert to the entry font	Revert to the entry font
* The font assigned to this index may be replaced in a future release of IDL.			

Table H-4: Embedded Font Selection Commands

Positioning Commands

The positioning and other font-manipulation commands are described in the following table. Examples of commands used to position text are included in "Formatting Command Examples" on page 2494.

Command	Action
!A	Shift above the division line .
!B	Shift below the division line .
!C	"Carriage return," begins a new line of text. Shift back to the starting position and down one line.
!D	Shift down to the first level subscript and decrease the character size by a factor of 0.62.
!E	Shift up to the exponent level and decrease the character size by a factor of 0.44.
!I	Shift down to the index level and decrease the character size by a factor of 0.44.
!L	Shift down to the second level subscript. Decrease the character size by a factor of 0.62.
!N	Shift back to the normal level and original character size.
!R	Restore position. The current position is set from the top of the saved positions stack.
!S	Save position. The current position is saved on the top of the saved positions stack.
!U	Shift to upper subscript level. Decrease the character size by a factor of 0.62.
!X	Return to the entry font.
!Z(u_0,u_1,...,u_n)	Display one or more character glyphs according to their unicode value. Each u_i within the parentheses will be interpreted as a 16-bit hexadecimal unicode value. If more than one unicode value is to be included, the values should be separated by commas.
!!	Display the ! symbol.

Table H-5: Vector-Drawn Positioning and Miscellaneous Commands

Formatting Command Examples

The figure below illustrates the relative positions and effects on character size of the level commands. In this figure, the letters "!N" are normal level and size characters.

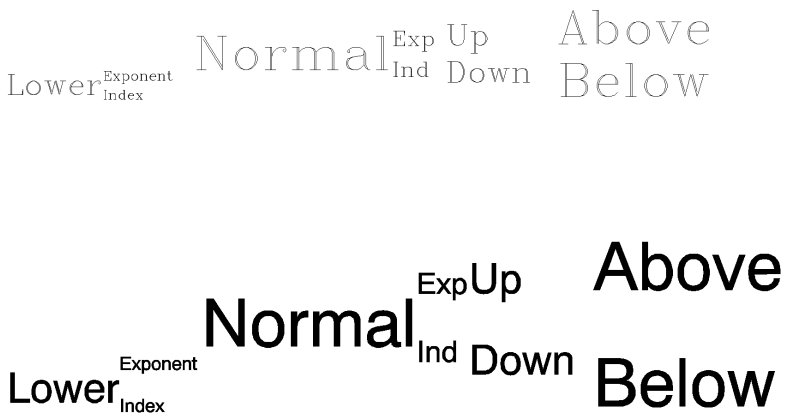


Figure H-1: Positioning commands with vector fonts (top) and TrueType fonts (bottom).

The positioning shown was created with the following command:

```
XYOUTS, 0.1, 0.3, $
'!LLower!S!EExponent!R!IIndex!N  Normal!S!EExp!R!IInd!N!S!U Up
!R!D Down!N!S!A Above!R!B Below'
```

Note that the string argument to the XYOUTS procedure must be entered on a single line rather than the two lines shown above.

A Complex Equation

Embedded positioning commands and the vector font system can be used to create the integral shown below:

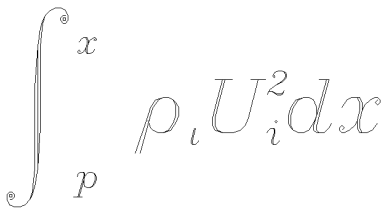


Figure H-2: An integral created with the vector fonts.

The command string used to produce the integral is:

```
XYOUTS, 0, .2, $
    '!MI!S!A!E!8x!R!B!Ip!N !7q!Ii!N!8U!S!E2!R!Ii!Ndx', $
        SIZE = 3, /NORMAL
```

Remember that the case of the letter in an embedded command is not important. The string may be broken down into the following components:

!MI

Changes to the math set and draws the integral sign, uppercase I.

!S

Saves the current position on the position stack.

!A!E!8x

Shifts above the division line and to the exponent level, switches to the Complex Italic font (Font 8), and draws the "*x*."

!R!B!Ip

Restores the position to the position immediately after the integral sign, shifts below the division line to the index level, and draws the "*p*."

!N !7q

Returns to the normal level, advances one space, shifts to the Complex Greek font (Font 7), and draws the Greek letter rho, which is designated by "*q*" in this set.

!Ii!N

Shifts to the index level and draws the "*i*" at the index level. Returns to the normal level.

!8U

Shifts to the Complex Italic font (Font 8) and outputs the upper case "*U*."

!S!E2

Saves the position and draws the exponent "2."

!R!Ii

Restores the position and draws the index "*i*."

!N dx

Returns to the normal level and outputs "*dx*."

Note

The equation shown in the figure above could not be created so simply using the TrueType font system, because the large integral symbol is broken into two or more characters in the TrueType fonts.

Vector-Drawn Font Example

IDL uses vector-drawn font when the value of the system variable !P.FONT is -1. This is the default condition. Initially, all characters are drawn using the Simplex Roman font (Font 3). When plotting, font changing commands may be embedded in the title strings keyword arguments (XTITLE, YTITLE, and TITLE) to select other fonts. For example, the following statement uses the Complex Roman font (Font 6) for the *x*-axis title:

```
PLOT, X, XTITLE = '!6X Axis Title'
```

This font remains in effect until explicitly changed. The order in which the annotations are drawn is main title, *x*-axis numbers, *x*-axis title, *y*-axis numbers, and *y*-axis title. Strings to be output also may contain embedded information

selecting subscripting, superscripting, plus other features that facilitate equation formatting.

The following statements were used to produce the figure below. They serve as an example of a plot using vector-drawn characters and of equation formatting.

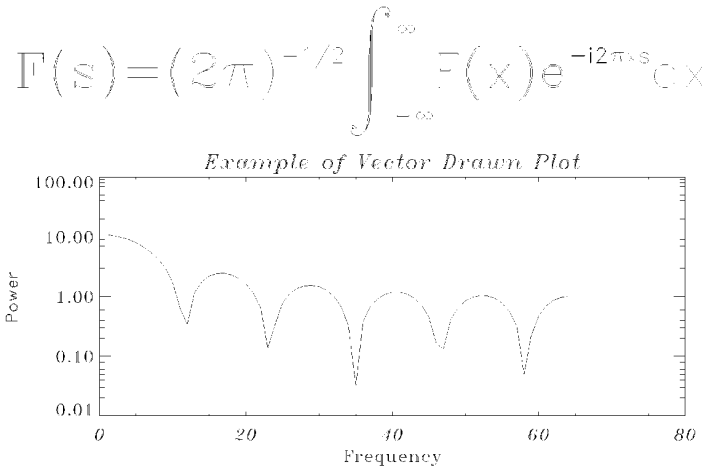


Figure H-3: Example of a Vector-drawn Plot.

```
; Define an array:
X = FLTARR(128)
; Make a step function:
X[30:40] = 1.0
;Take FFT and magnitude:
X = ABS(FFT(X, 1))
; Produce a log-linear plot. Use the Triplex Roman font for the
; x title (!17), Duplex Roman for the y title (!5), and Triplex
; Italic for the main title (!18). The position keyword is used to
; shrink the plotting window:
PLOT, X[0:64], /YLOG, XTITLE = '!17Frequency', $
   YTITLE = '!5Power', $
   TITLE = '!18Example of Vector Drawn Plot', $
   POSITION = [.2, .2, .9, .6]
SS = '!6F(s) = (2!4p)!e-1/2!n !mi!s!a!e!m $
     !r!b!i ' + '-!m $
; String to produce equation:
     !nF(x)e !e-i2!4p!3xs!ndx'
XYOUTS, 0.1, .75, SS, SIZE = 3, $
; Output string over plot. The NOCLIP keyword is needed because
; the previous plot caused the clipping region to shrink:
     /NORMAL, /NOCLIP
```

TrueType Font Samples

The following figures show roman versions of the four TrueType font families included with IDL. The character sets for the bold, italic, and bold italic versions of these fonts are the same as the roman versions.

The SHOWFONT command was used to create these figures. For example, to display the following figure on the screen, you would the command:

```
SHOWFONT, 'Helvetica', 'Helvetica', /TT_FONT
```

For more information, see "SHOWFONT" on page 1248.

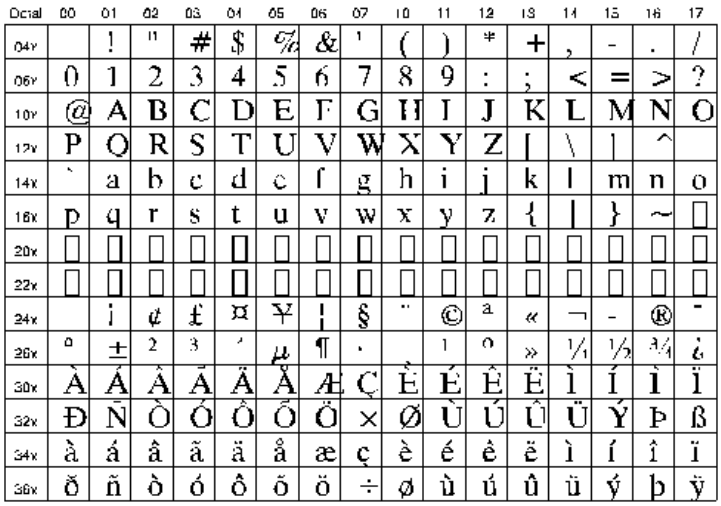
Note

The following font charts are numbered in octal notation. To read the octal number of a character, add the column index (along the top) to ten times the row index. For example, the capital letter "A" is octal 101, and the copyright symbol is octal 251.

Font Helvetica

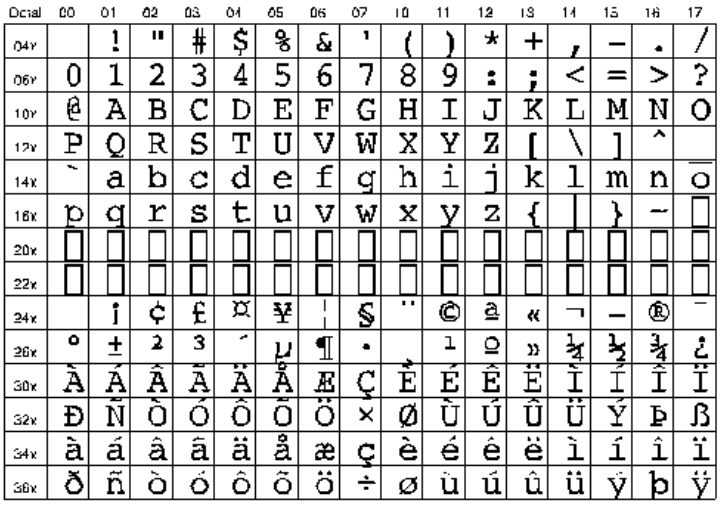
Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04x		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
06x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
10x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
12x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
14x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
16x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	□
20x	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
22x	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
24x		¡	ç	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
26x	°	±	2	3	´	µ	¶	·	¸	1	º	»	¼	½	¾	¿
30x	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
32x	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
34x	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
36x	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Font Times



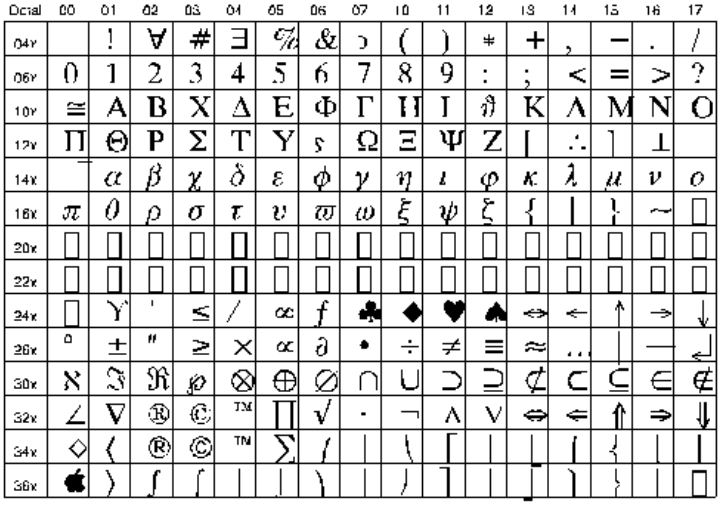
Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04x		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
06x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
10x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
12x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	
14x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
16x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	□
20x	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
22x	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
24x		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
26x	°	±	2	3	´	µ	¶	·	¸	1	º	»	¼	½	¾	¿
30x	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
32x	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
34x	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
36x	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Font Courier



Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04x		!	"	#	$	%	&	'	(	)	*	+	,	–	.	/
06x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
10x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
12x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
14x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
16x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	□
20x	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
22x	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□	□
24x		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	–	®	¯
26x	°	±	2	3	´	µ	¶	·	¸	1	º	»	¼	½	¾	¿
30x	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
32x	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
34x	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
36x	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Font Symbol


Octal 00 01 02 03 04 05 06 07 10 11 12 13 14 15 16 17
04x
06x
10x
12x
14x
16x
20x
22x
24x
26x
30x
32x
34x
36x

Vector Font Samples

The following figures show samples of various vector-drawn fonts. The SHOWFONT command was used to create these figures. For example, to display the following figure on the screen, you would the command:

```
SHOWFONT, 3, 'Simplex Roman'
```

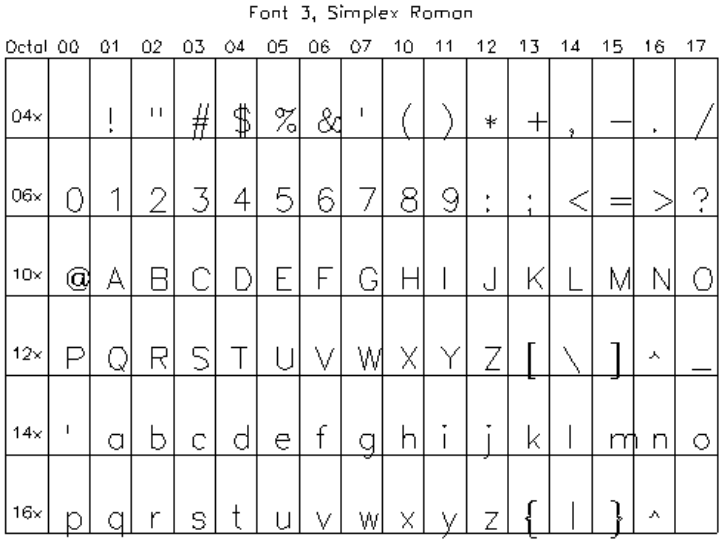
To output this figure to a postscript file, you would use the following commands:

```
SET_PLOT, 'PS'
SHOWFONT, 3, 'Simplex Roman'
DEVICE, /CLOSE
```

For more information, see “SHOWFONT” on page 1248.

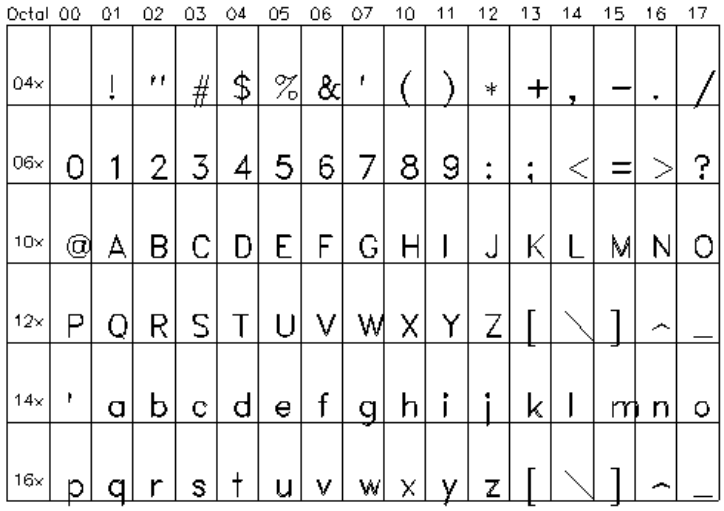
Note

The following font charts are numbered in octal notation. To read the octal number of a character, add the column index (along the top) to ten times the row index. For example, the capital letter “A” is octal 101, and the “$” symbol is octal 44.

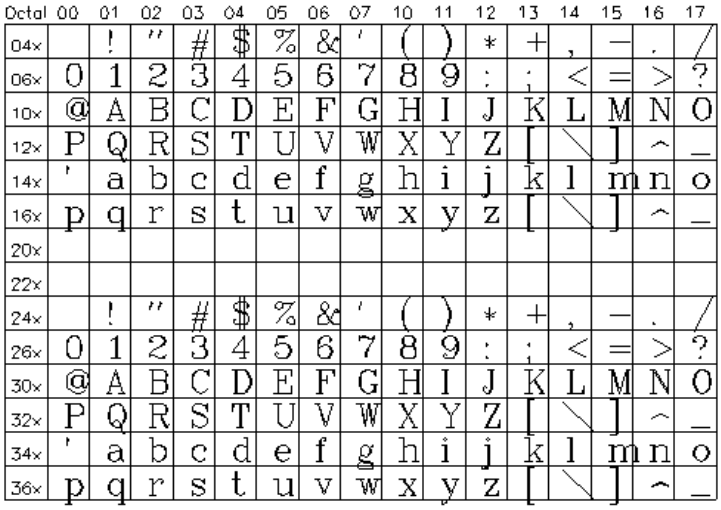


Font 4, Simplex Greek

Font 5, Duplex Roman



Font 6, Complex Roman

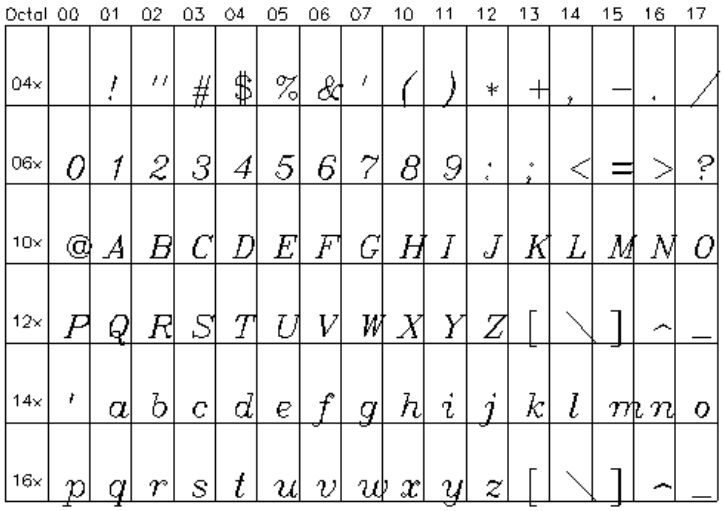


Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04x		!	"	#	$	%	&	'	(	)	*	+	,	−	.	/
06x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
10x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
12x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
14x	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
16x	p	q	r	s	t	u	v	w	x	y	z	[	\	]	^	_
20x																
22x																
24x		!	"	#	$	%	&	'	(	)	*	+	,	−	.	/
26x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
30x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
32x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
34x	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
36x	p	q	r	s	t	u	v	w	x	y	z	[	\	]	^	_

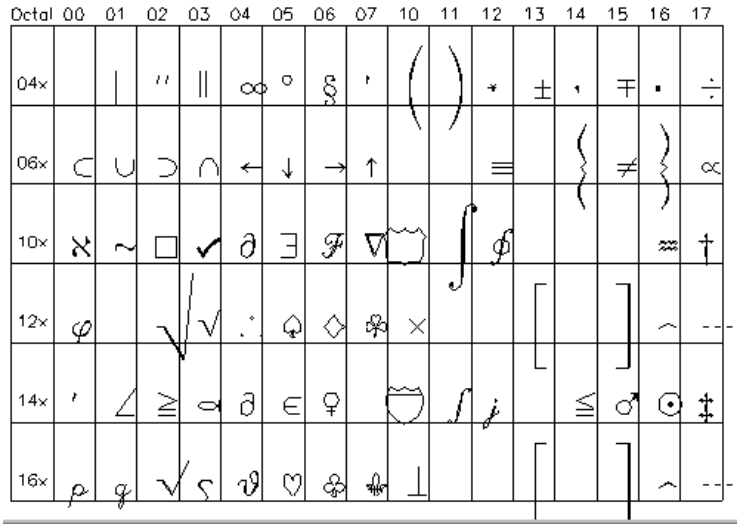
Font 7, Complex Greek

Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04x		!	"	#	$	%	&	'	(	)	*	+	,	−	.	/
06x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
10x	@	A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M	N	Ξ	O
12x	Π	P	Σ	T	ϒ	Φ	X	Ψ	Ω	∞	ℓ	[	\	]	^	_
14x	'	α	β	γ	δ	ϵ	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο
16x	π	ρ	σ	τ	υ	ϕ	χ	ψ	ω	∞	ℓ	[	\	]	^	_

Font 8, Complex Italic

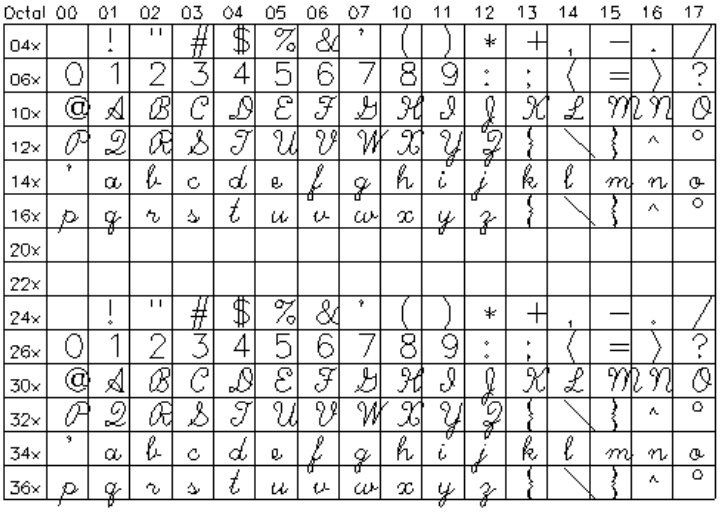


Font 9, Math and Special

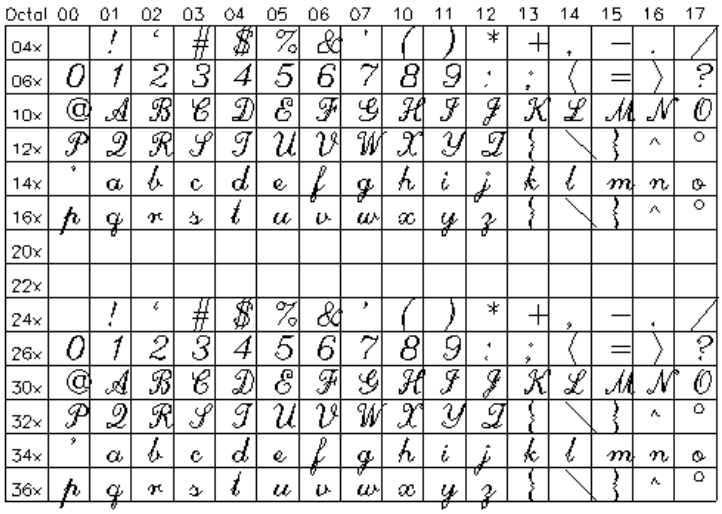


Font 11, Gothic English

Font 12, Simplex Script

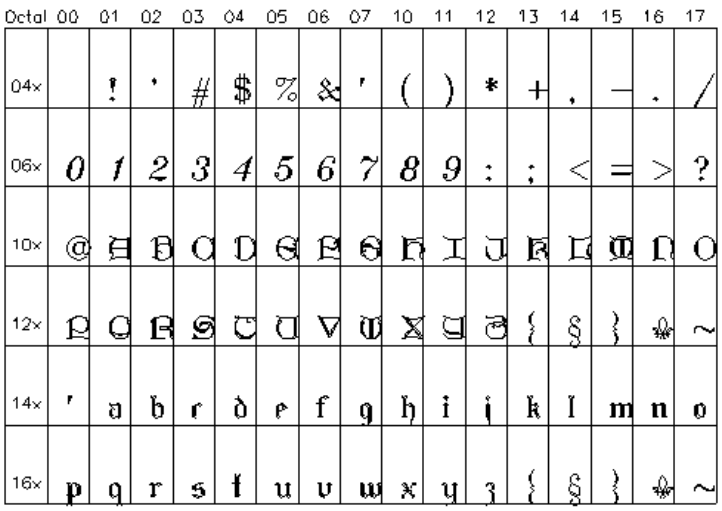


Font 13, Complex Script



Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04x		!	‘	#	$	%	&	’	(	)	*	+	,	–	.	/
06x	0	1	2	3	4	5	6	7	8	9	:	;	〈	=	〉	?
10x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
12x	P	Q	R	S	T	U	V	W	X	Y	Z	{	\	}	^	°
14x	’	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
16x	p	q	r	s	t	u	v	w	x	y	z	{	\	}	^	°
20x																
22x																
24x		!	‘	#	$	%	&	’	(	)	*	+	,	–	.	/
26x	0	1	2	3	4	5	6	7	8	9	:	;	〈	=	〉	?
30x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
32x	P	Q	R	S	T	U	V	W	X	Y	Z	{	\	}	^	°
34x	’	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
36x	p	q	r	s	t	u	v	w	x	y	z	{	\	}	^	°

Font 14, Gothic Italian



Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04x		!	’	#	$	%	&	’	(	)	*	+	,	–	.	/
06x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
10x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
12x	P	Q	R	S	T	U	V	W	X	Y	Z	{	§	}	^	~
14x	’	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
16x	p	q	r	s	t	u	v	w	x	y	z	{	§	}	^	~

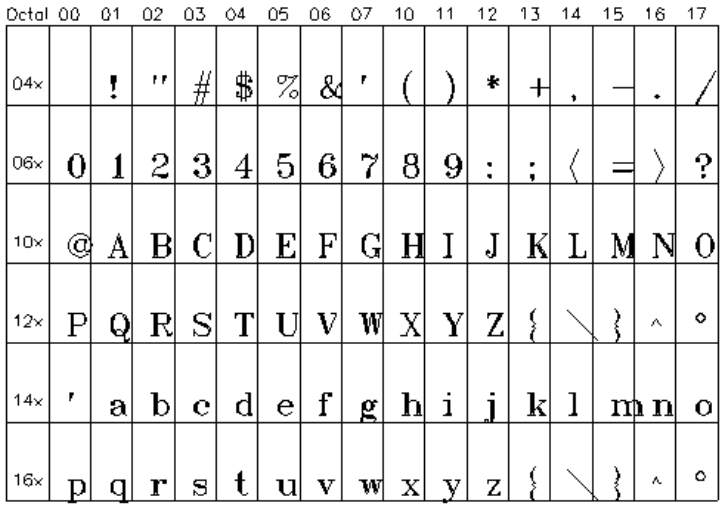
Font 15, Gothic German

Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04x		!	"	#	$	%	&	'	(	)	*	+	,	−	.	/
06x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
10x	𝔰	𝔄	𝔅	𝔈	𝔇	𝔈	𝔉	𝔊	𝔎	𝔍	𝔍	𝔎	𝔏	𝔐	𝔑	𝔒
12x	𝔓	𝔔	𝔎	𝔖	𝔗	𝔘	𝔙	𝔚	𝔛	𝔜	𝔷	{	ß	}	ß	~
14x	'	𝔞	𝔟	𝔠	𝔡	𝔢	𝔣	𝔤	𝔥	𝔦	𝔧	𝔨	𝔩	𝔪	𝔫	𝔬
16x	𝔭	𝔮	𝔯	𝔰	𝔱	𝔲	𝔳	𝔴	𝔵	𝔶	𝔷	{	ß	}	ß	~

Font 16, Cyrillic

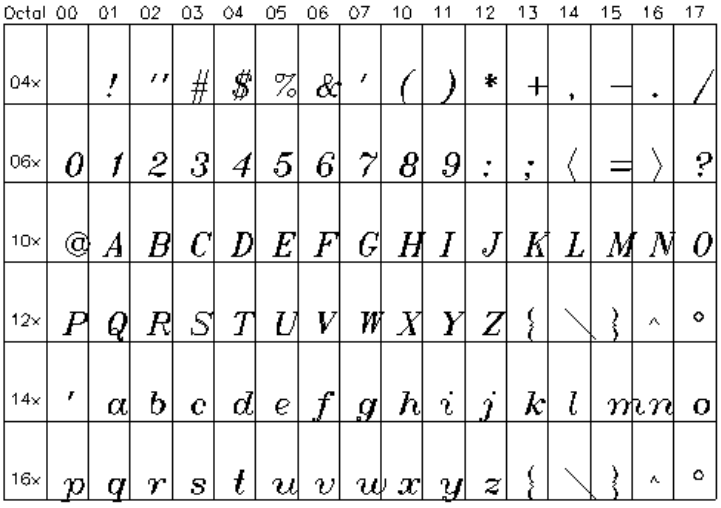
Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04x		!	Ю	Ъ	$	Э	&	'	(	)	*	+	,	−	.	/
06x	0	1	2	3	4	5	6	7	8	9	:	я	ъ	=	ы	?
10x	ь	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О
12x	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ы	э	Ь	ю	Я
14x	'	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о
16x	п	р	с	т	у	ф	х	ц	ч	ш	щ	ы	э	ь	ю	я
20x																
22x																
24x		!	Ю	Ъ	$	Э	&	'	(	)	*	+	,	−	.	/
26x	0	1	2	3	4	5	6	7	8	9	:	я	ъ	=	ы	?
30x	ь	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О
32x	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ы	э	Ь	ю	Я
34x	'	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о
36x	п	р	с	т	у	ф	х	ц	ч	ш	щ	ы	э	ь	ю	я

Font 17, Triplex Roman



Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04×		!	"	#	$	%	&	'	(	)	*	+	,	–	.	/
06×	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
10×	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
12×	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	°
14×	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
16×	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	°

Font 18, Triplex Italic



Octal	00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17
04×		!	"	#	$	%	&	'	(	)	*	+	,	–	.	/
06×	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
10×	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
12×	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	°
14×	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
16×	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	°

Font 20, Miscellaneous