

Canadian kg

The *Fancy Font*[™] System
User's Manual

SoftCraft

8726 S. Sepulveda Blvd. Suite 1641
Los Angeles, CA 90045
(213) 641-3822

This manual was printed by the *Fancy Font* System.

Table of Contents

Part I -- Introduction	Page
I.1 What to Read	I-1
I.2 Program Interaction	I-1
I.3 Font Terms -- Size, Style, Face	I-2
Part P -- Printing with Fonts	
P.1 The Text File	P-2
P.2 Parameter Specification	P-21
P.2.1 Interactive Mode	P-21
P.2.2 Command Line Mode	P-22
P.2.3 Parameter File Input	P-23
P.2.4 Which Disk Drives to Use	P-23
P.2.5 Special Characters	P-24
P.3 Parameters	P-26
P.3.1 Parameters Description Format	P-27
P.3.2 The Parameters	P-28
P.4 Commands While Printing	P-56
P.5 Error Interaction	P-56
Part E -- Editing Fonts	
E.1 Commands	E-1
E.2 Printing Mechanics	E-4
E.2.1 Interline Spacing Limitation	E-5
E.3 Maximum Character Size	E-6
Part C -- Creating Fonts	
C.1 Commands	C-1
Appendices	
A1 Glossary	A1-1
A2 Font Descriptions and Samples	A2-1
A3 Using <i>Fancy Font</i> with WordStar	A3-1
A4 Minimum/Maximum Values	A4-1
A5 Distribution File Descriptions	A5-1
A6 Hershey Database	A6-1
A7 Data File Formats	A7-1
A8 ASCII Conversions	A8-1
A9 Parameter and Command Summary	A9-1

Part I -- Introduction

The *Fancy Font*TM system provides the tools necessary to fully exploit the superb resolution offered by the Epson printer. The system is embodied by three separate programs, the main one being the printing program (Pfont). The other two programs allow editing of existing fonts (Efont) and creation of new fonts from a master character table (Cfont).

The printing program provides many features normally found in word processors. However, *Fancy Font* is not a general purpose word processor, and instead, it should be viewed as a very sophisticated printing system combined with moderately sophisticated text formatting capabilities. No personal computer word processor exists today that can fully exploit the capabilities of *Fancy Font*. But, such word processors are available on larger machines and will eventually be available on personal computers. Until then, *Fancy Font* provides the ability to achieve high quality print and a high degree of format flexibility for medium-sized documents.

Pfont achieves high printing quality by passing the print head over each character many times -- six times for standard characters, more for tall characters -- moving the paper as little as a 216th of an inch between each pass. Faster, rough draft modes are obtained by using fewer passes.

I.1 WHAT TO READ

The remainder of the manual is composed of three parts, describing Pfont (part P), Efont (part E) and Cfont (part C). If you are only interested in printing with the distributed fonts then only part P needs to be read. Parts E and C should be read by those desiring to edit or create fonts. Part E also contains interesting information on the detailed printing method and vertical line spacing limitations. This latter information will be critical to anyone trying to modify a word processor for use with *Fancy Font*, but should not be necessary for casual users of Efont.

There are numerous useful appendices. The glossary defines terms that may be foreign to casual computer users and also defines terms specific to the world of typesetting and character design. The last appendix (A9) gives a summary of the commands and parameters accepted by all three programs of *Fancy Font* and can be consulted during reading of the manual to get a better feeling for the whole of the system structure. The *Fancy Font* package comes with a predefined terminal and printer profile. If your terminal uses non-standard backspace codes or has a non-standard screen size, or if you have an MX100 printer, then appendix 5 should be consulted for directions on profile reconfiguration.

I.2 PROGRAM INTERACTION

The three programs in *Fancy Font* perform different functions and, of necessity, have slightly different user interfaces. Pfont asks the user to specify some parameters and then it prints for an extended period of time. In computer science terms we would say it has a declarative nature. Efont and Cfont have more of an interactive command interface and thus have a more procedural nature. The

major difference then is that Pfont allows specification of its parameters either on the CP/M command line or interactively, whereas Efont and Cfont are only interactive and have no command line parameters. All of the programs provide online help, but Pfont in particular provides extensive help (on demand) and has very robust error recovery mechanisms.

I.3 FONT TERMS -- SIZE, STYLE, FACE

A number of data files are included along with the *Fancy Font* programs and most of these are font sets. A font set is generally a collection of characters of like style and size. The sizes of the characters are given in printer's points (a point is a 72nd of an inch). For comparison, normal typewriter size is generally 10 to 12 points (unrelated to 10 and 12 pitch typewriter mechanisms). Two common font styles are Roman and Sans Serif and appendix 2 presents a variety of other font styles. Also, different faces may be defined for a style, such as italic or bold. The glossary contains more detailed definitions of these terms but this short introduction should suffice for reading of subsequent parts of the manual.

Part P -- Printing with Fonts

As mentioned in the introduction, Pfont is the primary program in the *Fancy Font* trio of programs. The basic function of Pfont is to format and print text, either from a file on disk or interactively from the keyboard, on an Epson printer. A text file can be printed by typing

```
A><u>Pfont <u>text.ff</u>
```

The underlined characters are typed by the user, 'A' is the CP/M prompt, and '>' represents the carriage return key (Pfont may be executed from any drive). In this case 'text.ff' is the name of the file to be printed. Executing Pfont in this way will cause the contents of 'text.ff' to be printed using the built-in settings for unspecified parameters such as line spacing, line width, font size and style, etc. These settings can easily be changed. For example,

```
A><u>Pfont <u>+sp .5</u>
```

will cause the file to be printed as before but with a half inch of space between each line of text. In general, many different parameters can be specified on the command line. Alternatively, parameters may be specified interactively to take advantage of the extensive online help within Pfont. To specify parameters interactively, simply execute Pfont without any command line arguments, e.g.

```
A><u>Pfont</u>
```

Parameters can then be specified in response to the Pfont prompt, which is composed of two angle brackets. For example,

```
A><u>Pfont</u>
>> <u>text.ff <u>+sp .5</u>
>> <u></u>
```

is equivalent to the previous example where the interline spacing was set to a half inch. The extra carriage return ends the interaction and tells Pfont to begin printing.

The user can intermix specification of parameters on the command line and interactive specification and help can be obtained about the parameters at any time.

This completes the introduction to the Pfont *printing parameters*. The other aspect of Pfont to be explained is the types of special commands that can appear *in the text file*. That is, different words on a line of text can be printed in different fonts, words can be underlined, and lines can be centered or flushed against the right margin. These actions and others are accomplished by placing commands in the text file at the location where the action is desired. All commands are of the form: backslash (\) followed by a single letter. For example '...\uquality software\u...' appearing in the text file would cause the words 'quality' and 'software' to be underlined, as in quality software. The '\u' command is called an underlining

"toggle"; i.e., it turns underlining on if it was off and off if it was on. The most commonly used command is probably the '\f' or "change font" command. The '\f' is followed by a single digit specifying which font number to change to. Thus, '...\f2Fancy Font\f0...' would cause 'Fancy' and 'Font' to be printed in font number 2, as in *Fancy Font*, and the remainder of the text to be printed in font zero; the correspondence between font numbers and the actual font is specified by the Pfont '+fo' parameter and can be changed with each printing of the text file if desired. (In the case above font 2 was mapped to the font FF12.fon). The '+fo' parameter is described in detail in section P.3 and the complete list of allowable in-text commands is described in the following section.

How to Read About Pfont

This introduction provides you with the basic ideas behind Pfont -- the capability to select and specify printing parameters and the embedded in-text commands to control the appearance of sections of text.

The remaining sections of part P detail the in-text commands, the modes of interaction with Pfont, and the parameters. They also provide additional information on controlling printing and on understanding and recovering from errors.

All commands and parameters are described in a consistent format to aid understanding and to provide convenient reference.

P.1 THE TEXT FILE

The most common use of Pfont is to print a file containing text. The text file can be created with any standard CP/M text editor or with most word processing software. The use of Pfont was introduced in the previous sections and is described in detail in sections P.2 and P.3.

There is a collection of in-text (embedded) commands which indicate appearance and formatting characteristics of portions of the text to be printed. Each of the commands is preceded by a command indicator to distinguish it from normal text. The built-in command indicator is the backslash character (\), however if this is inconvenient for a particular file, you can use the Command Indicator (CI) parameter (see section P.3) to specify a different command indicator for the entire text file. The command indicator is followed by 1 or a few characters to specify the details of the requested appearance. The characters following the command indicator may be either upper or lower case.

Thus all embedded commands follow the format:

<Command Indicator><Command Character><Value>

For example,

\f5

indicates that remaining text is to be printed using font 5. The '\ ' is the <Command Indicator>, 'f' is the <Command Character> and 5 is the <Value>.

There can be no spaces within the embedded command (see example above). Some commands require a multi-digit <Value>. For example, vertical motion \v0072 advances the paper 1 inch. In such specifications exactly the right number of digits must be specified (in the \v example, 4 digits). The distance is measured in points or dots — not in inches. For vertical motion, there are 72 points to an inch, thus 0072 specifies one inch. For horizontal motion, there are 120 dots per inch, thus 0240 specifies 2 inches. This somewhat cryptic method has been used so that additional features can be made available without sacrificing printing speed. The description of each command which requires a multi-digit <Value> indicates the exact requirements and meaning of the <Value>.

The embedded commands are described in the following pages. Each command is described in terms of its purpose, its syntax (as in the example above), how to use the command (usage), and at least one example of use. Additionally, the possible values and errors are detailed. Finally, a set of notes and suggestions for use are included.

\b -- Break (temporary no justify)

Break (temporary no justify) -- \b

Signify a line that is not to be right-justified.

<i>Syntax</i>	<code>\b</code>
---------------	-----------------

Usage:

Place the `\b` command anywhere in your text file. The print line in which the `\b` is placed will not be justified. The `\b` can appear anywhere in the line. See the `\j` and `\k` commands.

Values:

none

Examples: `\jThis line is not justified, notice the right margin.\b`

Produces: This line is not justified, notice the right margin.

Errors:

none

Notes:

none

Suggestions:

Use `\b` to mark the end of a paragraph so that the last line will not be justified. `\b` can also be used to selectively turn off justification for special lines of text such as examples and nested paragraphs. If you notice that a printed line has very large gaps in it, you probably should have a `\b` at the end of that line.

\c -- Center

Center -- \c

Adjust text so that it is centered on the print line.

<i>Syntax</i>	\c
---------------	----

Usage:

The \c embedded command specifies text to be centered and the region of the line in which it is to be centered. Any text following the \c command up to the first tab (\t or \a), right alignment command (\r) or end of line, will be centered. If the end of line or the right alignment command are encountered, the text will be centered over the entire line (i.e. from the left margin for the width of the line - see LM and LW parameters). If a tab (\t or \a) is encountered, the centering region will start at the position of the \c command and end at the horizontal position that the tab specifies.

Values:

none

Examples: \cCentered Title

Produces:

Centered Title

\a0240\ccentered between tabs\a0480\ccolumn 2\a0720

Produces:

centered between tabs

column 2

Left\cCentered\rRight

Produces:

Left

Centered

Right

In the first example the centered text is ended by a λ (end of line). Thus it is centered over the entire line width (offset by the left margin). The second example shows two centered strings, one between 2 and 4 inches, the other between 4 and 6 inches. If the final absolute tab (\a0720) were not specified, "column 2" would be centered over the entire line (causing overprinting). The final example divides the line into 3 sections: left, centered and right aligned. The centered text is centered over the entire line since it is ended by a right align command (\r).

Errors:

A center command can not be specified within another center command; an error message will be displayed and the second center command will be ignored. If the length of the text to be centered exceeds the region in which the text is to be centered, the region is adjusted to be the length of the text without an error message being displayed.

Notes:

Centering can result in overprinting.

Suggestions:

Use in conjunction with HL and FL parameters to create centered header and footer lines. Use a combination of tabs, absolute tabs and center commands to create labelled tables.

\d -- Ascii Character

Ascii Character -- \d

Specify an ASCII character by its decimal equivalent.

Syntax \d<integer between 000 and 127>

Usage:

The \d command will be replaced by a single character which is the ASCII equivalent of the indicated decimal number. See appendix A8 for a table of decimal-ASCII equivalents.

Values: <integer between 000 and 127>

The value must be exactly 3 digits long (e.g. 000, 122). The value indicates an ASCII character by its decimal equivalent. Thus the value must be between 0 and 127.

Examples: The A character is: \d065.

Produces: The A character is: A.

Errors:

If the 3 characters following the \d command are not all digits, or if the digits represent a number larger than 127, then an error message is displayed and the command and next 3 characters are ignored.

Notes:

none

Suggestions:

Use \d to include characters (such as control characters) in a text file that are otherwise difficult to enter using your text editor or on the command line.

\f -- Font Selection

Font Selection -- \f

Specify the font to be used for all printing until the next \f command.

Syntax \f<integer between 0 and 9>

Usage:

Place the \f command anywhere in your text file. All subsequent text until the next \f will be printed in the font represented by the integer between 0 and 9.

Values: <integer between 0 and 9>

For *Fancy Font* printing, the integer between 0 and 9 corresponds to a font specified using the +FO parameter. Font 0 is the first font specified by +FO, font 1 the second, etc. Pfont initially uses font 0 until changed by a \f command.

For printing using Epson fonts (i.e. +EP parameter), the font numbers select from a different set of fonts. The correspondence of numbers to Epson fonts is: (this correspondence can *not* be changed)

<u>Font Number</u>	<u>Epson Font</u>
0	Normal
1	Compressed
2	Expanded
3	Compressed/Expanded
4	Normal Italics
5	Compressed Italics
6	Expanded Italics
7	Compressed/Expanded Italics
8	Normal Emphasized
9	Expanded Emphasized

Examples: (using Pfont parameters: +FO Romn12 Romnb12)

\f0This is \f1wonderful weather\f0.

Produces: This is **wonderful weather**.

Where \f0 refers to Romn12 and \f1 refers to Romnb12.

Errors:

The integer following \f must be between 0 and 9 and must correspond to a font specified with the +FO parameter (if not using Epson fonts). If there is no correspondence, a message will be displayed and no font change will take place. Most fonts define a similar set of characters (alphabet, numbers, punctuation, etc.). If you select a character for printing that is not defined in the current font, that character will not be printed (see Appendix A2). An error message is not displayed in this event.

Notes:

1. When you change fonts, you change the definition of all characters including spaces. Thus, the width of a space may vary depending on the current font. For example,

\f0This is\f1 a test.\f0

may be spaced differently than:

\f0This is \f1a test.\f0

2. The amount of vertical space occupied by a printed line depends upon the tallest and deepest fonts used on the line. Pfont attempts to maintain uniform spacing between lines, but can not always do so (see section E.2). When this is not possible, a message indicating the number of extra *points* is displayed. The number of points indicates the extra white space between lines (1 point is 1/72 inch).

3. The height of a blank line is determined by the current font. As in note 1, the placement of the \f command determines the spacing. See \v for finer control of vertical white space.

4. Epson Fonts and *Fancy Fonts* can not be used within the same document.

Suggestions:

Use \f commands throughout text to print with multiple fonts for emphasis, super and sub-scripting, section headings, etc.

\h -- Horizontal Increment

Horizontal Increment -- \h

Leave 1 dot horizontal space (1/120 inch).



Usage:

The `\h` command will be replaced by exactly 1 dot of horizontal white space (1/120 inch).

Values:

none

Examples: There is a barely perceptible `g\ha\hp` between g, a and p.

Produces: There is a barely perceptible gap between g, a and p.

Errors:

none

Notes:

This does not work for Epson font printing (see EP) due to lack of horizontal resolution.

Suggestions:

If you wish to squeeze slightly more characters on a line but do not want to adjust the line width, you may replace a space character with a sequence of `\h`'s to reduce the size of the space character. Otherwise, this command can be used by programmers to produce justified lines without using the `\j` command. See also `\i` command.

\k -- Justification Off

Justification Off -- \k

Turn off justification.

<i>Syntax</i>	<code>\k</code>
---------------	-----------------

Usage:

Place the `\k` command anywhere in your text file. No subsequent text will be justified until a `\j` command. The line in which the `\k` appears will *not* be justified.

Values:

none

Examples: `\j\k`This line is not justified, notice the right margin.

Produces: This line is not justified, notice the right margin.

Errors:

none

Notes:

none

Suggestions:

Use a combination of `\j`, `\b` and `\k` embedded commands to control justification.

\r -- Right Alignment

Right Alignment -- \r

Align a string of text so that it is flush with the right margin.

```
Syntax _____ \r _____
```

Usage:

All characters to the right of the \r command in a text line will be printed so that the rightmost character is flush with the right margin.

Values:

none

Examples: Left of page\rright aligned.

Produces: Left of page

right aligned.

Errors:

An error message will be displayed if two or more \r commands appear within the same line; all but the first \r will be ignored.

Notes:

none

Suggestions:

\r is very useful in formatting header lines (HL) and footer lines (FL).

\s -- Substitute

Substitute -- \s

Include current page number, file name or an arbitrary string in the text before formatting and printing.

Syntax \s# or \sf or \s<integer between 0 and 9>

Usage:

The \s command will be replaced in the text by a string of characters determined by the character following \s. The substitution takes place before justification or any other processing of embedded commands.

Values: # or f or <integer between 0 and 9>

If the character following the \s command is '#', then the entire command (\s#) will be replaced by the number of the current page.

If the character is 'f', then the command (\sf) will be replaced by the full name of the file currently being printed.

If the character is an integer between 0 and 9 and there is a corresponding substitution string specified (see Substitution +SU), then the entire command will be replaced by the substitution string.

Examples: This is page \s#, file \sf.

Produces: This is page 16, file ffmanp2.man.

(using Pfont parameters: +SU "String 1" "page is \s#")

Substitution occurs here: \s1 and here: \s0.

Produces: Substitution occurs here: page is 16 and here: String 1

Errors:

1. Substitutions can be indicated within substitution strings (see second example above). It is possible that a substitution string attempts to substitute itself (recursive substitution). A line length exceeded error message will be displayed and subsequent program behavior will be unpredictable. Immediately exit the program using Control-C following recursive substitution.

2. If the number of a nonexistent substitution string is used, an error message is displayed and the substitution request is ignored.

Notes:

1. Be careful that lines do not exceed the line width *after* substitution occurs.

2. The 'f' in '\sf' may appear as either an upper or lower case character.

3. Any embedded commands may appear in a substitution string and will be processed after substitution.

Suggestions:

The embedded command \s# is essential in printing the page number in the header or footer line. See HL and FL for details. Substitution strings can be used to fill in variable fields in form letters (one line at a time), to include frequently typed words or to represent a sequence of other embedded commands.

\t -- Tab

Tab -- \t

Horizontal motion to the next tab stop.

Syntax	\t
--------	----

Usage:

Whenever the \t command (or its Control-I synonym) is recognized in the text, an attempt will be made to find the next available tab stop (see Tabs parameter). If found, the print head will be positioned at the tab stop, otherwise the tab will be ignored.

Values:

none

Examples: (using Pfont parameter: +TB 4)

Tab coming\tfirst stop.

Produces: Tab coming first stop.

The word first appears 4 inches from the left margin.*Errors:*

If there is no remaining tab to the right of the current print position, an error message is displayed and the tab command is ignored.

Notes:

1. \t and Control-I (the normal "Tab" key on your keyboard) are synonyms and can be used interchangeably.
2. Tabs can only move the print head forward (towards the right). See \a for backwards horizontal motion.
3. Tab stops can not be changed within a document.
4. All tab stops are relative to the left margin.

Suggestions:

Use a combination of tabs and absolute tabs to help line up text in columns and tables.

\u -- Underlining

Underlining -- \u

Turn on and off underlining of text.

<i>Syntax</i>	\u
---------------	----

Usage:

Place the \u command anywhere in your text file. All subsequent text will be underlined until another \u. Thus, \u is a toggle. Each time it appears in text underlining will be turned off if it was on, or on if it was off.

Values:

none

Examples: Some of this is \uunderlined text\u, some is not.

Produces: Some of this is underlined text, some is not.

Errors:

none

Notes:

White space is not underlined, only characters which print are underlined (see example above). Continuous underlining, including spaces, can be achieved by using a space character other than ASCII code 32. For convenience all of the distributed fonts have the Control-X character (ASCII 24) defined as a space. Use Control-X (or \d024) instead of the normal space to have underlined white space.

Suggestions:

Don't forget to turn off underlining (by using another \u). If you forget, the remainder of your text will be underlined.

\v -- Relative Vertical Motion**Relative Vertical Motion -- \v**

Specify the amount of white space to precede the current line.

<i>Syntax</i>	<code>\v<integer between 0000 and 9999></code>
---------------	--

Usage:

Insert the `\v` command in text followed immediately by 4 digits specifying the amount of white space to precede the current line (measured in points = $1/72^{\text{nd}}$ inch). Regardless of the position of the `\v` command in a text line, the white space will precede that line.

Values: <integer between 0000 and 9999>

The value for the `\v` command must be exactly 4 digits long (e.g. 0000, 0045, 1203) and must immediately follow the `\v` with no intermediate spaces. The value indicates the amount of white space (measured in points) to precede the current line of text when it is printed. A value of 0000 will leave the minimum amount of space before the print line (see section E.2.1 for details).

Examples: There is 1/2 inch of white space between this line

`\v0036`and this line.

Produces: There is 1/2 inch of white space between this line

and this line.

Errors:

If the 4 characters following the `\v` command are not digits, an error message will be displayed; the `\v` command and next 4 characters will be ignored.

Notes:

1. If `\v` appears on the first line of a printed page it will be ignored unless the new page was explicitly requested by a `\p` command, a Control-L character, or if it is the first page of a document. `\v` can cause a new page to be printed. In this case any unused portion of the distance specified with `\v` will be discarded.
2. Remember that interline spacing refers to the distance between the bottom of one line (including descenders) and the top of the next.

Suggestions:

Use the `\v` command to temporarily override the value of the Interline Spacing parameter (SP). The command is useful for specifying the exact amount of white space to be left before paragraphs, sections, examples and in tables. You may leave a blank line in your text to achieve vertical white space. `\v` provides this same capability, but with greater control over the amount of white space.

\w -- Absolute Vertical Motion**Absolute Vertical Motion -- \w**

Specify that the next print line is to appear at a specific distance from the top of the page.

Syntax `\w<integer between 0000 and 9999>`

Usage:

Insert the `\w` command in text followed immediately by 4 digits specifying the vertical location of the current line (measured in points = 1/72nd inch). Regardless of the position of the `\w` command in a text line, the text line in which the `\w` appears will be placed at the indicated distance from the top of the page.

Values: <integer between 0000 and 9999>

The value for the `\w` command must be exactly 4 digits long (e.g. 1011, 0045, 1203) and must immediately follow the `\w` with no intermediate spaces. The value indicates the distance (measured in points) from the top of the page to the top of the current line of text. The value must result in forward vertical motion. That is, the distance specified must be greater than the distance of the previous line from the top of the page. You can not "back up" the paper in the printer.

Examples: `\w0396`This line will be 5 1/2 inches from the top of the page.

Produces: This line will be 5 1/2 inches from the top of the page.

Errors:

If the 4 characters following the `\w` command are not digits an error message will be displayed and the `\w` command and next 4 characters will be ignored.

Notes:

If `\w` appears on the first line of a printed page it will be ignored unless the new page was explicitly requested by a `\p` command, a Control-L character, or if it is the first page of a document. The distance is measured from the top of the page to the top of the current print line. `\w` can cause a new page to be printed. In this case any unused portion of the distance specified with `\w` will be discarded.

Suggestions:

Use the `\w` command to place footnotes at the bottom of selected pages or to otherwise place text at a specific vertical location on a page.

P.2 PARAMETER SPECIFICATION

Different users have different needs at different times and many users have different approaches to using computers. Therefore, we have designed Pfont to provide 3 methods of interaction. You may select to use any one or a combination of these methods each time you use Pfont. The methods are: 1) **Interactive**— parameters are specified one or more at a time, help may be requested at any time; 2) **Command Line**— all parameters are specified on the CP/M command line; and 3) **Parameter Input File**— input is obtained from a prepared file containing parameter specifications (it is intended that this file be reused for multiple "runs" of Pfont).

New users should be most interested in the interactive method, frequent and experienced users should learn how to use the parameter input files and command line methods.

Regardless of the method of interaction, you can specify any number of parameters using upper or lower case or any combination thereof. If you specify the same parameter multiple times, only the last specification of the parameter will be used. If the parameter specifies a list (such as +FO Romn12 Romnb12 ...), each time the parameter is respecified the old list is discarded.

When learning or uncertain about the use of Pfont, it is best to experiment. Make frequent use of the help facilities (? , &) and feel secure in that Pfont performs no destructive operations.

P.2.1 Interactive Mode

Pfont is designed to assist in its use; this is particularly true in the interactive mode. To use Pfont interactively, start by typing the following:

B> Pfont

Pfont will respond with its usual greeting message as well as reminding you that '?' may be typed *at any time* for assistance. Pfont prompts with ">>" and awaits input. You may enter as many or as few parameters as you like and may also make use of the special characters '?', '&', '>' and 'Control-V' (see section P.2.5).

In the following transcript of Pfont use, user input is underlined, ↵ is 'carriage return' and commentary is in small print.

B> Pfont

Pfont: Print with Fonts (Version x.x)

Copyright (C) 1982 SoftCraft

Type '?' for assistance at any time

>> ?

Expecting a Parameter Specification - a '+' or '-' followed by

One of:

FO Fonts

FI Filenames

...

Start Pfont.

Pfont identification and version. This message overwrites a temporary "loading" message.

Request for assistance.

Pfont is expecting any of the following

Pfont is expecting any of the following

Pfont lists many other options

Pfont will indicate "-- more --" after the

Part P -- Printing with Fonts

>> +rd?

Expecting value for RD Rough Draft Parameter - a Integer Between 0 and 2

Current Value: <Off>

>> RD Rough Draft> 1

>> +fi Sample.FF

FI Filenames> 3

>> +fo Romn12 Romnb12 Romni12

FO Fonts> 3

>> &

...

>> 3

Loading Romn12.Fon as font number 0 ... in memory

Loading Romnb12.Fon as font number 1 ... in memory

Loading Romni12.Fon as font number 1 ... in memory

Make sure the printer is 'On Line'

Printing Sample.FF

Page 1, 42 Lines

B>

number of lines which fit on your screen are displayed, press any key to continue.

Select the 'rd' parameter, then ask for help.

Pfont indicates the current setting for

Rough Draft (Off) and possible settings (0,1,2).

User selects rough draft mode 1.

User indicates that file Sample.FF

is to be printed (+F1 is optional here).

3 indicates no more file names.

User selects 3 fonts.

3 indicates no more fonts.

Request summary of current settings.

Pfont lists the settings of all parameters.

A final 3 ends interaction and starts printing.

The 3 fonts are now ready for use.

Reminder to get the printer ready.

Pfont will now start printing.

Pfont has finished printing the first page.

All done.

Notice that carriage return (↵) is used to end a parameter specification and also to end interaction and begin prompting. The *ending* function of ↵ only applies when it is the only input on a line. When you type ↵ at the end of a sequence of characters, it allows you to continue specifying the same parameter. If this seems confusing or if you ever feel *stuck* while using Pfont, simply type a couple of ↵'s and Pfont will either start printing or display a message indicating additional parameters that must be specified.

P.2.2 Command Line Mode

Pfont can be used without any interaction by specifying all of its parameters on the CP/M command line. For example,

B> Pfont +fi Sample.FF +fo Romn12 RomnB12 Romni12

will print the contents of the file Sample.FF using 3 Roman fonts. The maximum length of the parameter specification list is 128 characters. CP/M converts all command line input to upper case, so parameters that have case sensitive values should be entered either interactively or using a parameter input file. Pfont modifies the CP/M convention somewhat; it converts the first character of each string to upper case and converts all others to lower case. For example:

input: Pfont test +hl "CHAPTER TWO" +fl "end of page"

result: Pfont Test +Hl "Chapter two" +Fl "End of page"

Any of the special characters '?', ',', '&', '<' etc. can be used on the command line,

',' '>' and '?' are of particular use (see section P.2.5).

If you make an error during command line interaction, the portion of the command line following the error is retained, you are automatically placed in interactive mode and are presented with an error message. See section P.5 for details.

P.2.3 Parameter File Input

As you become an experienced Pfont user, you will find that you often wish to print a letter or document multiple times or to print similar letters or documents using the same settings for Pfont parameters. Pfont's parameter file input capability makes this possible. A parameter file is simply a file containing parameter specifications exactly as you would type them on the command line or interactively. To request parameter specifications that have been stored in a parameter file, you need only type:

<file name

Where *file name* is the name of the parameter input file (names with the extension 'ffi' are used by convention, e.g. Sample.ffi). The *<file name* can be included anywhere on the command line or be used at any time in the interactive mode. To create your own parameter files, use your editor and simply type parameter specifications; the parameter file may contain one or more lines. One or more parameter input files can be used for any "run" of Pfont. Parameter input files can contain additional requests for parameter input (i.e. *<file name*). However, any parameters specified in the original parameter input file *after* the "<" will be ignored. For most uses, if a parameter input request is used within a parameter input file, the request should be the last item in the file.

It is recommended that you create a separate parameter input file for each type of printing you normally do (e.g. letters, mailing labels, disk labels, etc.).

P.2.4 Which Disk Drives To Use

Pfont requires that each of the files Pfont.com, Pfont00.ovl, Pfont01.ovl, Pfont02.ovl, Pfont03.ovl, Pfont.hlp, Pfont.hle and FancFont.pro be resident on one of the system disk drives. It is simplest to put all of these files on one disk and use Pfont while "logged" to the drive in which that disk is mounted. However, Pfont will find its overlays (Pfont00, Pfont01, Pfont02, Pfont03, FancFont.pro) if they are either on the logged disk or are on a disk mounted in disk drive 'A'.

For CP/M systems with a small disk storage capacity, such as the Osborne I, it may be necessary to switch diskettes in order to print long documents with multiple fonts. The following procedure is recommended:

1. Put all fonts to be used on one diskette (the font diskette).
2. Place the font diskette in drive B.
3. Place the file or files containing text on one diskette (the text diskette).
4. Insert the Pfont diskette in drive A.
5. Invoke Pfont using interactive mode.

6. Remove Pfont diskette and insert text diskette in drive A.
7. Interactively (or with parameter input file) specify Pfont printing parameters.
8. Remove text diskette and insert Pfont diskette in drive A.
9. Type final ↵ (carriage return) to start printing.
10. Pfont will prompt for disk change; following prompt, remove Pfont diskette and insert text diskette in drive A.
11. Printing should start, no further disk changes are required.

P.25 Special Characters

These special characters are recognized and acted upon by Pfont in any interaction mode, they are primarily used in the interactive mode, however.

'?' *Always* provides assistance. '?' may be used at any time during command line or interactive modes of Pfont use. The response to a '?' assistance request is information pertinent to the current usage of Pfont. In interactive mode, pressing '?' will produce a list of all available parameters if issued following the '>>' prompt. If pressed following a parameter name (e.g. +Rd ?), a description of acceptable parameter values as well as the current value for the parameter will be displayed.

Example: >> +Rd ?
 Expecting value for RD Rough Draft feature - a Integer
 Between 0 and 2
 Current Value: <Off>
 RD Rough Draft>

>> +Fo Romn12 Romn12?
 Expecting FO Fonts list element - a Font File Name
 Current Value:
 0: Romn12.Fon
 1: Romn12.Fon
 FO Fonts>

'?' may appear anywhere on the command line following Pfont. It will cause the interactive mode to be entered and Pfont will provide assistance. The most common command line use, however, is:

B> Pfont ??

This provides general assistance on using Pfont and displays a list of all parameters and their default settings.

'&' Display the current settings for all parameters. This is a good way to review any parameters you have specified and to determine the built-in settings for all other parameters. & may be issued interactively or on the command line.

Example: >> &
 FI Filenames - List of up to 15 File Names
 Current Value:

0:Sample.ff
FO Fonts - List of up to 10 Font File Names

...

';' Explicitly begins the interactive mode. Use this at the end of the command line if you have additional features you wish to specify interactively.

Example: B> Pfont <Test.ffi ,
Pfont: Print with Fonts (Version x.x)
Copyright (C) 1982 SoftCraft
>>

'<' Obtain parameter specifications from the parameter input file. Continue processing any remaining command or interactive input line parameters after input from the parameter input file. '<' may be used multiple times on the command line and/or interactively. See section P.2.3 for more information.

Example: B> Pfont <Control.ffi ,

will read the file Control.ffi as though its contents were typed on the command line and then Pfont will enter interactive mode (because of the comma) to allow further specification of parameters or override parameters specified in Control.ffi.

'Carriage Return' (↵) Ends input - it can be used to exit the interactive parameter specification, to end a list or to end a parameter if you decide it is not the one you wanted (useful after using '?', for example: '+rd ?' followed by ↵ leaves 'rd' as it was). 'Carriage Return' is ignored if it is not the first key typed on an input line and thus can be used to help in typing long lines.

'Control-V' Display on the terminal a description of the special characters (this can only be used interactively). The file 'Pfont.hlp' must be on the logged disk.

'Control-C' Exit Pfont, ignore all parameter settings and do not print. Pfont will ask for a confirmation of a Control-C request. Control-C may be pressed whenever you are using Pfont in the interactive mode.

'Back Space' (normally Control-H) Backs up one character at a time to correct typographical errors. This is analogous to the CP/M character delete function. The back space character and its displayed action can be set in the profile file (FancFont.Pro -- see appendix A5).

'Control-U' Delete an entire input line. This line delete character is analogous to

the CP/M line delete function and is operational in interactive mode.

P.3 PARAMETERS

Pfont offers a variety of parameters (features) to control the appearance, formatting and printing of a document. Most of the parameters have preset (default) values and hence need not be specified during each use of Pfont. However, you will want to use these parameters at different times to fully exploit the capabilities of the *Sansy Font* system. Each parameter controls some aspect of the formatting and printing of text. A parameter may be turned *off* or *on* or left at its normal (default) setting.

This section describes each of the parameters in detail, discussing the common aspects of using the parameters and then the method by which the parameters are described in the manual. Each parameter specification fits the format:

<On/Off switch> <Parameter Name> <Parameter Value>

The On/Off switch is used to indicate whether the parameter is to be used or not. '+' indicates On and '-' indicates Off. The name indicates which parameter is being specified and the values detail the use of the parameter (some parameters do not require values). For example,

+TM 1.5

specifies that the top margin parameter be used and that the top margin is 1.5 inches.

-TM

specifies no top margin (this is exactly the same as +TM 0). Some parameters have no values, they are either on or off. For example:

+PP

specifies that single pieces of paper are being used (the page pause parameter is on), and

-PP

indicates continuous form paper is in use (do not use the page pause parameter). Finally, some parameters can have a value that is a list of items:

+FO SanS10 SanS12 Romn12

turns on the *Sansy Font* parameter (+FO) and indicates that each of the 3 fonts listed are to be used. As before,

-FO

indicates that no Sansy Fonts are to be used (the parameter is *off*).

When specifying parameter values, all distances are specified in inches (e.g. 3, 6.0125). Parameter names may be specified in either upper or lower case.

P.3.1 Parameter Description Format

The rest of section P.3 describes each of the parameters, one parameter to a page. The top line on the page indicates the descriptive name of the parameter. The second line gives a very general description of the meaning of the parameter. Then the *syntax* of the parameter is presented. The syntax indicates how to type the parameter, either turning it on or off, whether it requires any values and what those values are. At the right edge of the syntax line, the default use of the parameter is indicated. If you do not specify the parameter, the default for that parameter will be used.

The *usage* indicates how to use the parameter; first how to turn the parameter on, then how to turn the parameter off and the meaning of doing each of these. The *values* section provides details about the parameter's values, describing acceptable and unacceptable values and the meaning of each. The final 3 sections provide *examples*, *notes* and *suggestions*. Each of these sections will help clarify the use and meaning of the parameter. Notes provide the most detailed information and suggestions provide ideas as to how to best use the parameter.

There are a few words and symbols used in describing the parameters that may be unfamiliar to you. A quick summary:

<...>

Brackets are used to describe a value. You do not actually type any words contained in the brackets nor should you type the brackets. (e.g. <list of real numbers between 0 and 150 inches>, indicates a list of numbers - you fill in the numbers).

list

One or more values; the values are separated by at least 1 space (e.g. 5 10.5 12 36 or File1 File2 File3).

string

A sequence of characters. Any characters may appear in a string, but if a space character is in a string, the entire string must be surrounded by double quotes to mark its beginning and end (and to distinguish it from a list). For example, "This is 1 string", is a string. If you wish to have the " character appear within a string, you must type the quote twice. Example, "This ""string"" contains quotes". The paired quotes are undoubled when the string is used, thus the example yields:

This "string" contains quotes

integer

A whole number, no fractional part (e.g. 1, 3, 12, 0, 5). There are no negative numbers in Pfont.

real number

A number which may include a fractional part (e.g. 1.5, .7, 0.2, 6, 12.8).

File Name

The name of a file using standard CP/M conventions (e.g. B:Name.Ext). If the file name contains Pfont special characters (see below), it must be enclosed in quotes.

Font File Name

The name of a file containing the characters of a font. The 'Fon' is optional, it will be provided if it is missing, otherwise this is a standard CP/M file name (see File Name).

Pfont Special Characters

Several characters have a special meaning when parameters are indicated to Pfont. If you want to use one of these characters, but not its special meaning, the character must appear in a quoted string (e.g. "?" or "A & B"). The special characters are:

? & " < + - ,

See section P.2.5 for a description of each.

P.3.2 The Parameters

The parameter descriptions follow in alphabetical order.

BM -- Bottom Margin**Bottom Margin -- BM**

Specify the amount of white space to appear after the bottom of the main body of text on a page.

Syntax: +BM <real number between 0 and 150 inches> or -BM [default: +BM .75]

Usage: +BM <real number between 0 and 150 inches>

The amount of white space to place at the bottom of each page. This is the distance from the actual bottom of the page to the bottom of the last line of text on the page (other than the footer line).

-BM

Do not leave any white space at the bottom of each page, no bottom margin (same as +BM 0).

Values: <real number between 0 and 150 inches>

A number between 0 and 150 inches. This is the amount of white space to be left at the bottom of each page.

Examples: +BM .75

Leave a .75 inch bottom margin on each page.

-BM

No bottom margin.

Notes:

1. The bottom margin is actually measured in printer points ($1/72^{\text{nd}}$ inches). The number of inches is converted to points; any fractional points are ignored.
2. If there is no bottom margin (-BM), then there can be no Footer Line (FL).
3. If FL is specified, the bottom margin must be large enough to accommodate the height of the footer line plus the footer margin (FM).

Suggestions:

If you change the Page Length (PL) you will probably want to also change the top and bottom margins (TM and BM).

CF -- Concatenate Files

Concatenate Files -- CF

Indicate whether files are to be individually printed or concatenated during printing.

<i>Syntax</i> +CF or -CF [default: +CF]
--

Usage: +CF

The text from all files specified with the Filenames (FI) parameter is to be printed as if it came from one file. No page breaks are automatically included between files. However, each file must end with a carriage return or one will be provided.

-CF

The text from all files specified with the Filenames (FI) parameter is not to be concatenated. A page break is automatically inserted before printing the contents of the next file and all embedded printing controls are reset (e.g. font, underlining). Page numbering will restart at 1 or the value specified with the Initial Page Number (PN) parameter for each file.

Values:

none

Examples: +CF

Concatenate the contents of all files in the Filenames (FI) list without page breaks.

-CF

Force a page break between printing the contents of each file in the Filenames (FI) list.

Notes:

1. Each concatenated file begins on a new print line.
2. See Initialization String (IS) parameter for related information.

Suggestions:

Use +CF for printing a document which has several sections where each section is contained in a different file. Use -CF for printing several unrelated documents in the same "run" of Pfont.

CI -- Command Indicator

Command Indicator -- CI

Specify a character to be used to indicate commands embedded in text files.

<i>Syntax</i>	+CI <1 character> or -CI	[default: +CI \]
---------------	--------------------------	------------------

Usage: +CI <1 character>

A single character to be used as a flag to indicate embedded commands within the text files. Normally this character is "\".

-CI

Use the default command indicator ("\"). Same as +CI \.

Values: <1 character>

The command indicator must be 1 character and should not be a character used for an embedded command. The upper and lower case characters f, t, a, u, -, v, w, s, #, p, c, r, h, j, k, b, d and i are specifically not recommended. It is best to select an infrequently used character such as "" or "%".

Examples: +CI %

Change the command indicator from "\" to "%".

-CI

Use the default command indicator.

Notes:

1. Command indicators are case sensitive; that is, they only work in the case (upper or lower) that you specify. Warning - if the command indicator parameter is specified on the CP/M command line, the CI will automatically be converted to upper case.

2. If you need to print the command indicator, type it twice in the text file (e.g. "\\\" will print "\").

Suggestions:

Change the command indicator if the default command indicator "\" is frequently used in a document or is difficult to generate with your keyboard. Be careful to change the command indicator each time you reprint a file containing other than the default command indicator.

EP -- Epson Font

Epson Font -- EP

Enable/Disable printing with normal Epson fonts.

Syntax: +EP or -EP [default: -EP]

Usage: +EP

Printing will use the normal Epson fonts - *Fancy Fonts* can *not* be used simultaneously. See description of the \f embedded command for how to select fonts. The font number mappings for Epson fonts are as follows:

Font Number	Epson Font
0	Normal
1	Compressed
2	Expanded
3	Compressed/Expanded
4	Normal Italics
5	Compressed Italics
6	Expanded Italics
7	Compressed/Expanded Italics
8	Normal Emphasized
9	Expanded Emphasized

-EP

Do not use Epson fonts, allows the +FO specification of *Fancy Fonts* to be used.

Values:

none

Examples: +EP

The embedded \f commands will refer to normal Epson fonts.

-EP +FO Romn12

The embedded \f commands will refer to *Fancy Fonts* specified (Romn12) using the +FO parameter.

Notes:

1. The horizontal formatting commands are designed primarily for use with *Fancy Fonts* and thus perform all measurements in dots (1/120th inch). The character placement of normal Epson font characters does not have this precision. Therefore tabs, justification and other horizontal formatting parameters do not work with fine accuracy when printing using Epson fonts (i.e., +EP).

2. If neither +FO nor +EP is specified, a warning message will be displayed and +EP will be assumed.

Suggestions:

The +EP parameter is useful for printing program listings and documents where printing quality is not as important as speed of printing, but where the formatting parameters of Pfont are desired. We print our program listings using +EP and also use pagination, header line and footer line parameters to enhance readability. See IS parameter for more help in printing existing non-*Fancy Font* files.

FI -- Filenames

Filenames -- FI

Specify the files containing the text to be printed by Pfont.

Syntax: +FI <list of 1 to 15 CP/M file names> [default: none]

Usage: +FI <list of 1 to 15 CP/M file names>

Print the contents of the listed files. When more than 1 file is specified the files can either be concatenated or printed with intervening form feeds (\p) as specified by the Concatenate Files (CF) parameter. The default is to concatenate files. Note that -FI is meaningless.

Values: <list of 1 to 15 CP/M file names>

Each file named must be a valid CP/M file. File names containing special Pfont characters ("+", "-", "?", "&", ...) must be enclosed in double quotes (""). Two special names may be used in the file list. A file name of "tty:" indicates that input is to be obtained from the user's terminal as if it were input from a file. With this parameter, Pfont allows your computer to be used as an electronic typewriter. ('Control-Z' indicates the end of terminal input). "tty:" may appear anywhere in the file name list and may appear more than once. The file name "loop:" indicates that the list of file names is to be used repeatedly until the user explicitly causes printing to stop. "loop:" should appear as the last name in the list and should not be the only name in the list.

Examples: +fi manual.txt B:appendix

Print the contents of files "manual.txt" and "appendix" ("appendix" from disk 'B').

+FI envelope.ff tty: letter.ff loop:

Print contents of "envelope.ff", then enter text to be printed using the keyboard, then print contents of "letter.ff". Continue this sequence indefinitely until user enters Control-C to force it to stop.

Notes:

1. At most 15 items may appear in the list of file names.
2. A sequence of file names may be entered without the "+FI" control for convenience, but the file names may not immediately follow any other list. This allows a command line such as:

Pfont manual.txt appendix

FI is special in this way, it is the *default parameter*.

3. The files are checked to determine that they can be opened for use by the Pfont program.
4. See the Concatenate Files (CF) parameter for controlling the printing of multiple files.

Suggestions:

Use "tty:" for filling variable sections of letters and forms or for using your computer as an electronic typewriter. Use "loop:" for printing multiple copies of the same file or sequence of files. When using loop:, it is usually desirable to specify -CF so that each copy begins on a new page.

FL -- Footer Line**Footer Line -- FL**

Control printing of a footer line in the bottom margin of every page.

Syntax: `+FL <string> or -FL` [default: `-FL`]

Usage: `+FL <string>`

Specify 1 line to appear at the bottom of every page at a distance from the bottom of the page indicated by Footer Margin (FM).

`-FL`

No footer line will appear in the bottom margin.

Values: `<string>`

The string is a line to appear in the bottom margin of every page. The string can contain any embedded commands to control fonts, underlining etc. The length of the string must not exceed the Line Width (LW). If the string contains any spaces or special Pfont characters ("&", "?", etc.), surround the string with quotes ("").

Examples: `+FL "A Day at the Lake -- Section 3"`

The string A Day at the Lake -- Section 3 will appear in the bottom margin of every page. It will start at the left margin.

`+FL "\f2The Biker\f1\rpage \s#"`

The bottom margin of each page will contain a line with The Biker left justified and in font 2, and the word page followed by the page number right justified and in font 1.

Notes:

1. The footer line will always be printed using font 0, no underlining etc. unless otherwise specified.
2. See the Not On First Page (NF) parameter to disable the printing of the footer line on the first page of a document.
3. See FM and BM for determining the exact location of the footer line on a page.
4. The current font, underlining, etc. for normal text printing will not be affected by any embedded commands in the footer line.
5. CP/M converts all characters on the command line to upper case. Pfont will adjust so that the first word of a string is capitalized. For best results, avoid this problem by using interactive mode or a parameter input file to input the footer line.

Suggestions:

Use FL with the `\s#` embedded command to print the page number at the bottom of each page. Use FL with the `\c` and `\r` embedded commands to produce a left, centered and right justified footer line.

FM -- Footer Margin**Footer Margin -- FM**

Specify the amount of white space between the bottom of the footer line and the bottom of a page.

Syntax: `+FM <real number between 0 and 150 inches> or -FM [default: +FM .25]`

Usage: `+FM <real number between 0 and 150 inches>`

Leave the specified white space between the bottom of the footer line and the bottom of the page.

`-FM`

Place the footer line at bottom of page, no white space following the footer line.

Values: `<real number between 0 and 150 inches>`

A number between 0 and 150 inches. This is the amount of white space between the bottom of the footer line and the bottom of the page.

Examples: `+FM .5`

Leave .5 inches of white space between the bottom of the footer line and bottom of page.

`-FM`

No footer margin.

Notes:

The footer margin is actually measured in printer points (1/72nd inches). The number of inches is converted to points; any fractional points are truncated. The footer margin plus the height of the footer line must be less than the bottom margin. If no footer line (FL) is specified, the footer margin is not used.

Suggestions:

Consider changing the header margin and footer margin (HM and FM) when changing PL, TM or BM.

FO -- Fonts

Fonts -- FO

Indicate the fonts to be used for printing.

Syntax +FO <list of 1 to 10 *Fancy Font* file names> or -FO [default: -FO]

Usage: +FO <list of 1 to 10 *Fancy Font* file names>

Print using *Fancy Fonts*. The names and order specified in the file name list specifies the fonts to be used. See the \f embedded command.

-FO

Do not use *Fancy Fonts*. Use normal Epson fonts - see Epson Fonts (EP).

Values: <list of 1 to 10 *Fancy Font* file names>

Each name should be that of a font file containing the definitions of characters. The conventional ".fon" extension need not be given; if it is left off, Pfont will append it. The first font in the list will be known as font 0, the second as font 1, etc.

Examples: +FO Romn12 Romnb12 Sans10

Use font file "Romn12.fon" for font 0, "Romnb12.fon" for font 1 and "Sans10.fon" for font 2.

+FO b:Sans8.fon Sans10.fon

Use font file "Sans8.fon" (from disk B) for font 0 and "Sans10.fon" for font 1.

Notes:

1. At most 10 fonts (numbered 0 through 9) may be specified for any run of Pfont.
2. The files are tested to see if they are valid font files.
3. The order of the files is important. It specifies the mapping to the embedded \f commands, but also controls printing speed by partially determining which fonts fit in memory. For fastest printing, put your most frequently used fonts first in the list.
4. Before printing, Pfont will indicate which fonts fit in memory and which will be used directly from the disk.
5. See appendix A.2 for details on fonts (height, width, characters etc.).

Suggestions:

When creating a document and determining which fonts to use, write down the names and numbers of each font and reorder the list so that frequently used fonts appear first.

FP -- First Page

First Page -- FP

Select a portion of the document to be printed.

Syntax +FP <integer between 1 and 9999> or -FP [default: -FP]

Usage: +FP <integer between 1 and 9999>

Suppress all printing until the specified page number is to be printed. Resume normal printing from the specified page to end of document or to +LP specification.

-FP

Normal printing, begin printing with the first page of document.

Values: <integer between 1 and 9999>

The page number that is the first page to be printed. Any integer between 1 and 9999 may be specified, however if the number is greater than the page number of the last page, no pages will be printed.

Examples: +FP 4

Page number 4 will be the first page printed.

-FP

Normal printing, begin printing with first page.

Notes:

1. FP works with the same number that would be printed on a page; i.e., the one specified with the Initial Page Number (PN) parameter.
2. Pages which are not printed are still processed by Pfont and thus all embedded commands are checked and processed and all warning and error messages are displayed.

Suggestions:

1. Use this parameter in conjunction with the Last Page (LP) parameter to select one, or a sequence of pages to be printed. This is particularly useful if you need to reprint a portion of a document, but not the entire document.
2. Since error and warning messages will be displayed for all pages of a document (whether printed or not), the +FP parameter may be used to check a document for errors (such as line width exceeded) without any printing or screen display (see Screen Display, SD). This is accomplished by using the FP parameter and specifying a large page number (e.g. +FP 9999). To speed font loading, always use +SD in conjunction with this technique.

HM -- Header Margin

Header Margin -- HM

Specify the amount of white space to precede the Header Line (HL).

Syntax `+HM <real number between 0 and 150 inches> or -HM[default: +HM .25]`

Usage: `+HM <real number between 0 and 150 inches>`

The amount of white space between the top of a page and the top of the Header Line (HL).

`-HM`

Do not leave any white space between the top of the page and the top of the header line.

Values: `<real number between 0 and 150 inches>`

A number between 0 and 150 inches. This is the amount of white space between the top of the page and the top of the header line.

Examples: `+HM .5`

Leave .5 inches of white space before the top of the header line.

`-HM`

No header margin.

Notes:

The header margin is actually measured in printer points (1/72nd inch). The number of inches is converted to points; any fractional points are truncated. The header margin plus the height of the header line must be less than the top margin. If no header line (HL) is specified, the header margin is ignored.

Suggestions:

Consider changing the header and footer margins (HM and FM) when changing PL, TM or BM.

IS -- Initialization String**Initialization String -- IS**

Specify a string to be printed at the beginning of each file.

Syntax: +IS <string> or -IS [default: -IS]

Usage: +IS <string>

Insert a string of characters, possibly containing embedded commands at the beginning of the file to be printed. If files are not concatenated (see CF parameter) this string will be inserted before *each* file.

-IS

No special initialization, use default formatting controls.

Values: <string>

A string of characters, usually embedded commands, to specify the starting font, underlining mode and other printing characteristics. If the string contains any blanks or special Pfont characters, surround the entire string with quotes ("").

Examples: +IS "\f3\j"

Begin printing with font 3 and justified text.

-IS

No initialization.

Notes:

1. The initialization string will be applied before printing each file which is not concatenated. Thus, it will always apply to the first file, and will apply to all others if -CF (Concatenate Files) is specified.
2. The default initialization is font 0, no justify, no underline (see section P.1 for additional information).

Suggestions:

This parameter is convenient for printing files that you don't want to modify (e.g. listings, data files).

LM -- Left Margin

Left Margin -- LM

Control the Left Margin of the text to be printed.

Syntax +LM <real number between 0 and 8 inches> or -LM [default: +LM 1]

Usage: +LM <real number between 0 and 8 inches>

Specify a left margin for all text to be printed.

 -LM

Turn off the left margin parameter; there will be no left margin. (Same as +LM 0).

Values: <real number between 0 and 8 inches>

The number of inches to be used as a left margin for all text to be printed. The left margin must be between 0 and 8 inches (13.6 inches for MX100 owners). A left margin of 0 inches indicates no left margin and is equivalent to -LM.

Examples: +LM 1.5

Set a left margin of 1.5 inches.

 -LM

No left margin - text will be printed flush with left edge of page.

Notes:

All horizontal formatting parameters and commands (Tab, Line Width, \a, etc.) are specified relative to the left margin. Thus, if the left margin is changed, all the other parameters will reflect that change. If you specify a Line Width (LW) of 6.5 inches and a left margin of 3 inches then you have actually requested a right margin of 9.5 inches which is of course too large for MX80 printers. The left margin can *not* be changed within a text file.

Suggestions:

Use the left margin to horizontally reposition text on a page.

LP -- Last Page

Last Page -- LP

Select a portion of the document to be printed.

Syntax +LP <integer between 1 and 9999> or -LP [default: -LP]

Usage: +LP <integer between 1 and 9999>

Stop printing after the indicated page number is printed. The Pfont program will terminate following this page.

-LP

Normal printing, do not stop printing until the end of all specified files.

Values: <integer between 1 and 9999>

The page number which is the last page to be printed. Any integer between 1 and 9999 may be specified.

Examples: +LP 4

Page number 4 will be the last page printed.

-LP

Normal printing, end printing after the last page.

Notes:

LP works with the same number that would be printed on a page; i.e., the one specified with the Initial Page Number (PN) parameter.

Suggestions:

Use this parameter in conjunction with the First Page (FP) parameter to select one, or a sequence of pages to be printed. This is particularly useful if you need to reprint a portion of a document, but not the entire document.

LW -- Line Width

Line Width -- LW

Specify the maximum width of a printed line.

<i>Syntax</i>	+LW <real number between 0 and 8 inches>	[default: +LW 6.5]
---------------	--	--------------------

Usage: +LW <real number between 0 and 8 inches>

Indicate the width of printed lines of text. The width is measured from the left margin. Thus a left margin of 1 inch (+LM 1) and a line width of 7 inches (+LW 7) yields a line end of 8 inches. The line width is of particular importance for \j, \c and \r embedded commands. Turning off the line width is not allowed; neither -LW or +LW 0 would allow any characters to be printed.

Values: <real number between 0 and 8 inches>

The width of the line in inches. The sum of the left margin (LM) and the line width (LW) should not exceed 8 inches for MX80 or 13.6 inches for MX100 printers. If it does, an error message will be displayed and the margin set to the minimum.

Examples: +LW 5 +LM 1.5

Line width of 5 inches, left margin 1.5 inches, implied line end at 6.5 inches from left edge of paper.

+LW 8 -LM

Line width of 8 inches, no left margin (0 inches), implied line end at 8 inches from left edge of paper.

Notes:

If a printed line exceeds the line width, a message will be displayed on the user's terminal and the line will be truncated. No justification will occur for such a line (see \j embedded command).

Suggestions:

Use Screen Display (SD) to quickly check line widths.

NF -- Not On First Page

Not On First Page -- NF

Control the printing of Headers and Footers on the first page.

<i>Syntax</i> +NF or -NF	[default: -NF]
--------------------------	----------------

Usage: +NF

Disable printing of the Header Line (HL) and Footer Line (FL) on the first page of text only. Any header and footer specified by HL and FL parameters will be printed on the second and subsequent pages.

 -NF

Retain the normal operation of the Header Line (HL) and Footer Line (FL) parameters.

Values:

none

Examples: +NF

Disable printing of header and footer lines on first page only.

 -NF

Normal printing of header and footer lines on all pages.

Notes:

TM and BM are not affected by NF.

Suggestions:

Use the +NF parameter to print letters, and documents with a title page where the first page should not have a header or footer (e.g. page number), but where subsequent pages should have this information.

PF -- Process FormFeeds

Process FormFeeds -- PF

Enable/Disable recognition of Control-L characters as page feeds.

<i>Syntax</i>	+PF or -PF	[default: +PF]
---------------	------------	----------------

Usage: +PF

Any Control-L characters (ASCII decimal 12) in the text will be treated as if they are \p commands. That is, they cause the current page to be ejected and a new page to be printed.

-PF

Any Control-L characters (decimal 12) in the text are ignored. They do not force a new page and are not printed.

Values:

none

Examples: +PF

Control-L in text will force a new page.

-PF

Control-L in text will be ignored.

Notes:

The PF parameter does *not* impact the \p embedded command.

Suggestions:

-PF is useful for printing assembler listings and other text which may include embedded form feed characters (Control-L).

PG -- Paginate

Paginate -- PG

Shorthand for enabling or disabling all top and bottom margin parameters.

<i>Syntax</i> +PG or -PG	[default: +PG]
--------------------------	----------------

Usage: +PG

Enable normal function of the Top Margin (TM), Bottom Margin (BM), Header Margin (HM), Header Line (HL), Footer Margin (FM), Footer Line (FL) and Page Length (PL) parameters.

 -PG

Turn off (disable) all the top and bottom margin parameters (-TM, -BM, -HM, -HL, -FM, -FL, -PL). This allows Pfont to follow pagination and margin directives from other word processing software. Normal text files will print as one continuous stream of text with no top or bottom margins.

Values:

none

Examples: +PG

Normal pagination.

 -PG

Text will be printed continuously with no top or bottom margins.

Notes:

none

Suggestions:

Normally use the default, +PG. Use -PG if your text already includes page breaks, top and bottom margins, headers and footers.

PL -- Page Length

Page Length -- PL

Specify the length of a printed page including top and bottom margins.

Syntax +PL <real number between 0 and 150 inches> or -PL [default: +PL 11]

Usage: +PL <real number between 0 and 150 inches>

Indicate the total length of a page from the top of the paper to the bottom of the paper. A new page will be printed after the specified number of vertical inches has been printed (including top and bottom margins). The page length must exceed the sum of the top and bottom margins.

-PL

Perform no pagination (same as +PL 0). There will be no page breaks and no bottom margin. The top margin on the first page will be printed however.

Values: <real number between 0 and 150 inches>

The height of a page in inches. +PL 0 is equivalent to -PL.

Examples: +PL 4

Print using 4 inch pages.

-PL

No pagination.

Notes:

Maximum page length is 150 inches, however -PL (+PL 0) specifies no paging and thus an infinite page length.

Suggestions:

Use short pages for printing mailing or diskette labels. -PL (+PL 0) is useful for continuous printing without any page breaks.

PN -- Initial Page Number

Initial Page Number -- PN

Specify the number to be associated with the first page.

<i>Syntax</i>	+PN <integer between 0 and 9999> or -PN	[default: +PN 1]
---------------	---	------------------

Usage: +PN <integer between 0 and 9999>

Normally, if pages are numbered, the numbering sequence will begin with page 1 (-PN or +PN 1). If you require a different starting page, specify that page number by using +PN.

-PN

Use the default initial page number (1). Same as +PN 1.

Values: <integer between 0 and 9999>

An integer which is the page number to be associated with the first page. The page number must not exceed 4 digits.

Examples: +PN 5

Begin numbering the first page as page number 5.

+PN 50

Begin numbering the first page as number 50.

Notes:

The page number is only printed if the embedded command `\s#` is used to indicate where the page number is to be printed on each page. Page numbers are generally printed in either the header line or footer line, but can be printed anywhere in the text. First Page (FP), Last Page (LP) etc. will refer to the numbering of pages specified by the PN parameter.

Suggestions:

The initial page number parameter is convenient for printing different sections of a document using different "runs" of Pfont. For example, if the first section covers pages 1 to 12 then specify +PN 13 when printing the next section. Page numbers must be integers, however a compound page number such as P-48 can be printed by a string such as P-\s#.

PP -- Page Pause

Page Pause -- PP

Control automatic and manual loading of paper.

<i>Syntax</i>	+PP or -PP	[default: -PP]
---------------	------------	----------------

Usage: +PP

Manual paper feed. Printing will stop at the end of each page and the printer will *beep* to indicate end of page. When any key on the keyboard is depressed, printing will resume.

-PP

Normal, continuous form paper feed. The printer will continuously feed paper and print each page until the end of the document.

Values:

none

Examples: +PP

Printer will pause after each printed page.

-PP

Normal, continuous form printing.

Notes:

1. +PP attempts to disable the printer's paper out detection circuitry; however, the Epson printers do not allow the disabling of this parameter in graphics mode (which is used for *Fancy Font* printing). If you wish to supply paper manually (special paper etc.) then you must manually disable the paper out detection. This is done by either setting the internal printer switch which disables paper out detection or by placing a piece of paper over the paper out detector on the left side of the printer platen.

2. The printer may be turned off to allow you to more easily change paper following a page pause, make sure to turn it back on.

3. You should probably use -TM when using +PP because it is difficult to position single sheets of paper in the printer to properly print at the top of the paper.

Suggestions:

Use this parameter when feeding one sheet of paper at a time. For example when printing on bond paper, letterhead, or preprinted forms.

RD -- Rough Draft

Rough Draft -- RD

Specify the quality and speed of printing.

Syntax: +RD <integer between 0 and 2> or -RD [default: -RD]

Usage: +RD <integer between 0 and 2>

Select a rough draft printing mode.

-RD

Do not use any rough draft mode, use the normal, high quality printing (same as +RD 0).

Values: <integer between 0 and 2>

0 Normal printing (high quality, low speed) same as -RD.

1 Draft printing (medium quality, faster speed).

2 *Sandpaper* printing (low quality, fast speed).

Examples: +RD 2

Cause all printing to be at high speed.

-RD

Highest quality, final copy printing.

Notes:

-RD 216 DPI (Dots Per Inch) vertical, 120 DPI horizontal

+RD 1 72 DPI vertical, 120 DPI horizontal

+RD 2 72 DPI vertical, 60 DPI horizontal

More fonts fit into memory during rough draft printing, further increasing printing speed.

Suggestions:

+RD 2 is very fast and should be used to preview pagination, line widths and general page layout. +RD 1 produces reasonable quality and can be used for drafts and some finished documents. Use -RD (+RD 0) for final, high quality printing. (See Screen Display (SD) for other drafting possibilities).

SD -- Screen Display

Screen Display -- SD

Preview, on the terminal, the text to be printed.

Syntax +SD or -SD

[default: -SD]

Usage: +SD

Display output on the user's terminal in a manner similar to that in which it would be printed. Do not actually print.

-SD

Normal printing -- do not preview the document.

Values:

none

Examples: +SD

Preview document on terminal, no printing.

-SD

Normal printing, no preview.

Notes:

1. When using +SD, output is displayed at the user's terminal rather than being printed. The output is displayed with the arguments for embedded commands visible. For example, if `\f3` appears in the text file, the `3` will appear in the processed text during screen display.
2. If +SD is indicated, no fonts will be loaded in memory in order to increase the speed of previewing (the information normally loaded in memory is not used in this mode).
3. The Screen Display parameter will preview line width problems and page breaks, but does not display on your console an exact view of printed output (i.e. no visual font changes or proportional spacing).
4. +SD causes display output paging (the `--more--` message) to remain in effect. Paging is normally turned off during printing.

Suggestions:

+SD is a quick way to determine proper line widths and preview a document. See the First Page (FP) parameter for an even quicker way to check line widths.

SP -- Interline Space

Interline Space -- SP

Control the white space between printed lines.

Syntax +SP <real number between 0 and 150 inches> or -SP [default: +SP .045]

Usage: +SP <real number between 0 and 150 inches>

Indicate the amount of white space to appear between printed lines. The distance being measured is from the bottom of one line to the top of the next.

-SP

Indicate that no white space is to appear between printed lines (same as +SP 0).

Values: <real number between 0 and 150 inches>

The amount of white space to appear between printed lines, measured in inches. A value of 0 (or -SP) will cause lines to be printed as close to each other as possible (see section E.2 for details).

Examples: +SP .5

Leave .5 inches of white space between printed lines.

-SP

Leave no white space between printed lines.

Notes:

1. The actual white space is measured in printers points ($1/72^{\text{nd}}$ inch). The specified inches are converted to points by truncating fractional points (e.g. .8 inches converts to 57 points).
2. Depending upon the heights and depths of fonts on 2 successive print lines, more white space than requested may appear. In this case, a message indicating the number of extra points of white space is displayed (this may happen even for blank lines). See section E.2 for a description of interline spacing.
3. The default spacing (.045 inches) is 3 points.

Suggestions:

Use the \v command in text to temporarily override spacing.

SU -- Substitute

Substitute -- SU

Specify substitution strings for \s commands embedded in text files.

<i>Syntax</i>	+SU <list of 1 to 10 strings> or -SU	[default: -SU]
---------------	--------------------------------------	----------------

Usage: +SU <list of 1 to 10 strings>

Specify the contents and order of strings to be substituted for \s commands. The first string corresponds to \s0 in the text, the second to \s1, etc.

-SU

Specify no substitution strings, any \s strings in the text will be flagged as errors.

Values: <list of 1 to 10 strings>

1 to 10 strings may be listed. Each string may include any embedded command including the \s command. If a string includes spaces or special Pfont characters ("&", "?", etc.) surround the string with double quotes (""). Each string must be no longer than one line of text. The strings are numbered 0 to 9 to allow easy specification of a maximum of 10 strings.

Examples: +SU "The \f2Fancy FontTM\f0 System" "October 1, 1901"

Substitute the string The Fancy Font™ System for each occurrence of \s0 in the text file. Notice the change of font. Also substitute the date October 1, 1901 for each occurrence of \s1 in the text file.

+SU Xyzzx Xylophone "Zebra?"

Substitute Xyzzx for \s0, Xylophone for \s1 and Zebra? for \s2.

Notes:

If the +SU parameter is specified on the CP/M command line, the first word of each substitution string will be capitalized; all other words in the string will be all lower case. To retain your desired capitalization use interactive mode or parameter input file.

Suggestions:

+SU and \s commands can be combined to associate *styles* with paragraphs or other sections of text. Define a string containing all the embedded commands (fonts, underlining etc.) which define up to 10 *styles* to be used in your documents. Then use the appropriate strings at the start of each paragraph, section, etc. Other uses include specification of frequently used phrases and information that you may wish to change each time a document is printed (e.g. date, addressee).

TB -- Tabs

Tabs -- TB

To clear or set evenly spaced or explicit tab stops.

Syntax +TB <list of 1 to 15 real numbers (inches)> or -TB [default: +TB .8]

Usage: +TB <list of 1 to 15 real numbers (inches)>

Set up to 15 tab stops.

-TB

Do not process any tabs; if a tab is detected in the text, an error message will be displayed on the terminal.

Values: <list of 1 to 15 real numbers (inches)>

The real numbers specify tab stops in inches. If only 1 number is given, then up to 15 evenly spaced tab stops will be set (the spacing will be the indicated number of inches). If more than one number is indicated, then a tab stop will be set at each of the indicated locations. The numbers must be in increasing order and must be less than the line width. All tab stops are set relative to the left margin.

Examples: +TB 1.5 4.25 6.75

Set 3 tab stops, the first 1.5 inches from the left margin, the second 4.25 inches from left margin, etc.

+TB .8

Set 15 tab stops, .8 inches apart (i.e. .8, 1.6, 2.4 ... inches from the left margin).

Notes:

Remember that the tabs are measured *from the left margin*.

Suggestions:

See \t, \a and \i commands for use of tabs and for tab alternatives.

TM -- Top Margin

Top Margin -- TM

Specify the amount of white space to appear before the top of the main body of text on a page.

Syntax +TM <real number between 0 and 150 inches> or -TM [default: +TM .75]

Usage: +TM <real number between 0 and 150 inches>

The amount of white space to place at the top of each page. This is the distance from the actual top of the page to the top of the first line of text on the page (other than a header line).

-TM

Do not leave any white space at the top of each page, no top margin (same as +TM 0).

Values: <real number between 0 and 150 inches>

A number between 0 and 150 inches. This is the amount of white space to be left at the top of each page.

Examples: +TM 3.5

Leave a 3.5 inch top margin on each page.

-TM

No top margin.

Notes:

1. The top margin is actually measured in printer points (1/72nd inches). The number of inches is converted to points; any fractional points are ignored.
2. If there is no top margin (-TM), then there can be no Header Line (HL).
3. If HL is specified, the top margin must be large enough to accommodate the height of the header line plus the header margin (HM).

Suggestions:

If you change the Page Length (PL) you will probably want to also change the top and bottom margins (TM and BM).

P.4 COMMANDS WHILE PRINTING

These commands are only available when printing is in progress. It takes a little while to get used to these since there is not immediate response when one of these commands is entered. Type any of these commands *once printing has started*, however the command *will not be acted upon until the start of printing of a new print line*. Furthermore, the print line must be in the main body of text, i.e. not the header or footer line. These commands require a bit of patience but nevertheless can be quite useful.

'Control-P' Stop printing current page and skip to the next page. Use this command to advance to the next page when the current page has an error or is otherwise not worth printing. If you know in advance that you want to use Control-P, try using the FP and/or LP parameters for better control.

'Control-F' Stop printing the current file and skip to the next file. Use this if you notice errors in the current file and wish to continue printing the contents of any remaining files.

'Control-C' Stop all printing and exit Pfont. Use this if you want to immediately quit printing. Control-C can be used to temporarily stop printing to allow paper to be repositioned. Make sure to answer 'n' when asked if you really want to quit. Press one additional key to continue printing.

P.5 ERROR INTERACTION

Since Pfont is quite flexible in allowing input to be specified in a variety of ways, it is difficult for Pfont to predict the cause of errors. Therefore, Pfont provides clear indication of where the error occurred, the problem detected and a choice of options as to how to proceed. The options deal with any input which follows the error and generally allow insertion of new input, selective deletion of existing input, continuing as if there were no error, or restarting from the beginning. Dealing with errors is simple if one parameter and one value is typed on each line. It becomes more complicated when the command line interaction, parameter input files and multiple parameters on a line are used.

When an error is encountered the unexpected input, as indicated in the error message, is discarded. Any remaining input can still be processed if you so desire. You can insert, delete, continue with the remainder of input or backup to leave a parameter (such as a font list). <E> is added to the normal prompt to indicate that you are currently recovering from an error and therefore have some special options.

The error recovery options are:

Normal Typing: Insert and continue normal processing (type the input to be inserted).

Control-B: Leave the current parameter and continue with other error correction options.

Control-D: Delete the next input word and continue with other error correction options. A final Control-D will stop input from the current parameter input file (if any).

Carriage Return (↵): Process remaining input and return to normal interaction.

Control-X: Discard all remaining input on the line containing an error (also stop input from parameter input file) and return to normal interaction (a combination of multiple Control-D's and a Carriage Return).

In the following transcript of Pfont error interaction, user input is underlined, ↵ is 'carriage return' and commentary is in small print.

```

B> Pfont Sample.ff +fo French12 RomnB12 +rd X Y +pl 6
Invoke Pfont to print the contents of file
'Sample.ff' using 2 fonts, rough draft printing
and a page length of 6 inches.

Pfont: Print with Fonts (Version x.x)
Copyright (C) 1982 SoftCraft      Pfont introductory messages.
Sample.ff +fo French12 RomnB12 +Rd X Y +Pl 6
      ↑
Font file 'French12' not found.    Font not on disk.
'French12' has been discarded.    Erroneous input discarded.
Remaining input:                  The unprocessed input is displayed
RomnB12 +Rd X Y +Pl 6             and some help is provided indicating
                                  error recovery options.

Insert input or press 'Return' to continue
Control-D to delete next input word, Control-B to backup
Control-X to ignore all remaining input and continue
FO Fonts<E> ↵                     Prompt indicates error recovery in progress,
                                  font file names expected. User types ↵ to
                                  continue processing input.

RomnB12 +Rd X Y +Pl 6             Another error, X is wrong.
      ↑                           An integer is required for RD parameter.

Integer was expected.
'X' has been discarded.
Remaining input:
Y +Pl 6

Insert input or press 'Return' to continue
Control-D to delete next input word, Control-B to backup
Control-X to ignore all remaining input and continue
RD Rough Draft<E> Control-D     The user decides to give up with
                                  rough draft and to clean up by deleting
                                  the next input, 'Y'.

Remaining input:                  Now only Pl parameter left.
+Pl 6

Insert input or press 'Return' to continue
Control-D to delete next input word, Control-B to backup
    
```

Part P -- Printing with Fonts

Control-X to ignore all remaining input and continue

RD Rough Draft<E> Control-B

Since user decided not to use rough draft, Control-B 'backs out' of the rough draft parameter.

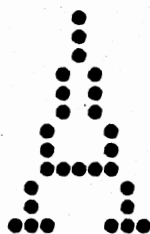
<E> 2

User presses 2 to continue by processing remaining input (i.e. the Page Length parameter.)

Part E -- Editing Fonts

Any character in any font set can be edited using any standard text editor. Characters or logos can also be created using a text editor and then installed in a new or existing font set using Efont. The basic operating sequence for editing a character is 1) create a text file for a character using the Efont *edit* command, 2) edit the character using a standard text editor, and finally 3) replace the original character by using the Efont *replace* command.

A character is defined by a pattern of dots (sometimes referred to as pixels); the Epson allows printing of as many as 216 dots per inch vertically and 120 per inch horizontally. For example, a magnified 'A' might look like this:



If the *edit* command were applied to this character it would write a standard text file with asterisks in place of the dots. The text file can then be edited with a text editor and asterisks can be moved, added, or removed. The *replace* command is able to read the modified file and convert the asterisks to the proper dot pattern. Besides the asterisks, there are three other items of information written into the text file by the *edit* command. The first two items are left and right margins for the character, i.e., the number of dot positions to be left blank around the character. Currently, these numbers must be positive or zero and normally, on small font sets, the left and right margins are each one. The third item is the relative height of the top dot in the character. We call this parameter *ytop* in the remainder of the manual. The absolute height is irrelevant, but when two fonts appear on the same line there must be some way of making the baselines of the characters line up horizontally. By convention (and pragmatics; see section E.2.1, the section on minimum line spacing) the baselines of all of the distributed fonts are at a height of seven, so the height of the top dot in the 'A' above would be 18. One use of the height information is in the construction of the superscript and subscript fonts. These fonts are identical to other fonts except that they have been moved up or down by varying the height parameter (this is conveniently done for an entire font using the Efont *updown* command).

E.1 COMMANDS

All command names are single letters and do not need to be followed by a carriage return. Immediately upon typing of the command letter the program will prompt the terminal for any required parameters. The list of available commands can be obtained by typing a question mark in response to the "Command" prompt. Question mark generally does not provide help when parameters are being requested by the

program. Unlike the Pfont program, Efont uses the standard CP/M terminal interface, so Control-C will exit immediately (when in type-in mode) and the normal CP/M line editing characters prevail. The command descriptions follow.

l -- Load Load a font set for editing or inspection. The terminal will be prompted for a font file name. The complete font name, including the 'fon' extension must be typed. Upon successful loading, font parameters such as top and bottom row, top and bottom blank points, and commentary will be displayed on the terminal (section E.2 describes these parameters). Any font previously loaded will be erased from memory. If there is not enough memory for the font then nothing will be loaded, but the previous font will still be erased. If the previously loaded font had been modified and not saved, then the terminal will be prompted for confirmation. Note, there is slightly more space available for loading a font at the first *load* command than for subsequent loads; if there is not enough room for a font try starting Efont afresh.

s -- Save Save the modified font set. Any replacements or other modifications made to a loaded font can be made permanent by saving the modified font on disk. The terminal will be prompted for a font file name; type the entire name including the 'fon' extension. If you try to overwrite an existing file the terminal will be prompted for confirmation.

p -- Print Print a set of expanded characters to the printer. A sequence of one or more characters can be expanded into their asterisk representation and printed on the printer. The terminal will be prompted for a range of characters to print (type their decimal ASCII codes).

e -- Edit Edit a set of characters by creating a text file for each. One or more characters can be expanded into their asterisk representation and saved into a text file. The file can subsequently be edited with any text editor and then the modified character can be reinstalled in the same font or a different font with the *replace* command. The terminal will be prompted for a range of characters to edit (type their decimal ASCII codes). If more than one character is specified then text files with names of "A<nnn>.edf", where "<nnn>" is the decimal ASCII code for the edited character, will be created, one file for each character, e.g., 'a65.edf'. If only one character is to be edited the terminal will be prompted for a file name for the edit output.

r -- Replace Replace a set of characters with their edit file specifications. This is generally used in conjunction with the *edit* command, but can also be used to read in characters or logos created from scratch with a text editor. The terminal will be prompted for a range of characters to be replaced and the replacement file names follow the same convention as do the *edit* command file names. The text file specification of the character to be replaced should begin with three integers, left margin, right margin, and *ytop* values, respectively (see section E.2 for a detailed discussion of the role of *ytop*). The rest of the file should be asterisks and spaces indicating the dot pattern of the new character. Look at a file created by the *edit* command for an example of the required file format. See section E.3 for a discussion of the maximum character size.

If WordStar is used to edit these files be sure to use non-document mode to ensure creation of a CP/M compatible file.

d -- Display Display an expanded character on the terminal screen. This is similar to the *print* command, but output is to the terminal and only a single character is displayed.

m -- Margins Modify the margins of a range of characters. The left and right margins can be set to different values for a range of characters. The terminal will be prompted for a range of characters and for new left and right margin values. (The "box width" of a character is the horizontal distance taken up by the character on a line. It is the "ink width" of the character plus the left and right margins. There is currently no way to set the overall box width of a set of characters to a uniform number other than by individually changing the margins of characters according to the ink width of the character.) The margins must be numbers in the range 0 to 120. A space character is just a character with a left and/or right margin but no ink. Different sized spaces can be made by changing the margins of character codes that were previously undefined in the font.

t -- Test Test the font by printing a string on the printer. After replacements or other changes have been made it is useful to test the new font. This command prompts for a string to be printed. A carriage return ends the string input. The only Pfont embedded commands that are recognized are '\d', '\u' and '\\'.

z -- Zap Zap (remove) a set of characters from the currently loaded font set.

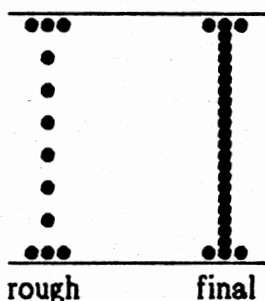
u -- Updown Specify a value to move characters up or down by when the *edit* command is used. The terminal will be prompted for a positive or negative integer. A positive integer indicates that characters should be moved up (by changing the *ytop* value) when they are edited with the *edit* command. A negative integer means move them down. The *edit*, *display*, and *print* commands are the only commands affected by *updown*. To move characters up or down and make the changes permanent, use: 1) *updown*, 2) *edit*, 3) *replace*, 4) *save*. See section E.3 for a discussion of the maximum and minimum allowable vertical locations of characters. Note that this command has no immediate effect on the font, but is only used in conjunction with the *edit* command.

a -- Auxinfo Display and/or modify the auxiliary font information. Auxiliary information includes font descriptive commentary and the top and bottom blank points values (section E.2 describes these parameters). This command prints the current values and then asks if they should be changed. Entering a carriage return will always leave the values unchanged. Use this command also to get a quick listing of defined and undefined characters in a font.

q -- Quit Quit. If any changes have not been made permanent by saving the font, then Efont will prompt for confirmation.

E.2 PRINTING MECHANICS

Because of the parameters of the Epson print head and the minimum allowable paper movement we find it convenient to define a new term called a "print head row" or just "row". A row contains up to 24 vertical dot locations and is 8 points in height (yielding the 3 dots per point, or 216 dots per inch, resolution of the Epson). Therefore, an 8 point font usually fits into one row. In rough draft mode 1, one row is printed with one pass of the print head, thus printing at most 8 dots (only 8 pins of the print head are usable in graphics mode). In final draft mode three passes of the print head are required to fill in all 24 dots. However, in either case the height of the row is 8 points; in final draft mode the dots are closer together. The concept of "row" and the two printing modes are depicted in the following diagram.



The dots between the horizontal lines constitute one row. Top and bottom rows are specified for each font to define the vertical location of the font relative to other fonts. The row number for a given height is the height divided by 24 (for positive numbers), and, again, the number has only relative, not absolute significance. For example, a font consisting of just the letter 'I' above, assuming the top dot was at height 23 (i.e. a *ytop* value of 23), would have *top row* of zero and *bottom row* of 0. If the *ytop* value were 22 instead, the font would have a *top row* value of zero, but a *bottom row* value of -1 because the lowest dot would fall into the next lower row as in the following diagram.



(row -1)

This has implications for both the minimum interline spacing and for the font file storage size as explained below. The top and bottom row values for a font are the maximum top and minimum bottom for all characters in the font. Note that with respect to the example above a font containing the letter 'I' would usually have lower case letters as well as the 'I', and the descenders of the lower case letters would fall into the -1 row anyway.

Two other height related parameters besides top and bottom row are associated with a font. These parameters indicate the relative position of a font within the top and bottom row and are expressed as the number of points of blank space at the top of the top row and at the bottom of the bottom row. For example, if the *ytop* value of the 'I' was 20 then there would be three dots, or 1 point, of blank space in the top row of the 'I'. Therefore, if the 'I' was the highest character in the font then the font would have a *top blank points* value of 1 point. In the same situation the 'I' would extend 3 dots into row -1 (since there are 24 dots total in the 'I') and, if the 'I' was the lowest character in the font then the font would have a *bottom blank points* value of 7 points. These values affect the interline spacing as described in the next section.

E.2.1 INTERLINE SPACING LIMITATION

The *top blank points* and *bottom blank points* values are combined in the Pfont program to determine the interline gap or the amount of white space between two lines of text. Consider two consecutive lines of text where the upper line has a *bottom blank points* of "upperbp" and the lower line has a *top blank points* value of "lowertp". The minimum interline gap between these two lines is "upperbp+lowertp-7". For example, consider the 'I' font where the *ytop* of the 'I' is 20 again. Two consecutive lines of this font would yield a "upperbp" of 7 and a "lowertp" of 1, so the minimum interline gap would be "7+1-7" or one point. That is, it would be impossible to have the bottom of the upper 'I' just touch the top of the lower 'I'; there would be at least one point of space (3 dots) regardless of the value of the '+sp' parameter.

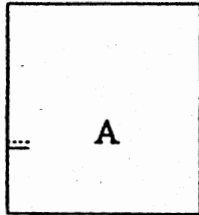
Alternatively, look at the original 'I' font that had a *ytop* value of 23. Here "upperbp" would be zero and "lowertp" would also be zero yielding a minimum spacing of -7 points. That is, the two lines could theoretically overlap by 7 points. (The current version of Pfont does not allow a negative '+sp' parameter, however, so the actual minimum spacing would be zero. Negative values may be allowed in future versions.)

These two examples point up the practical reason for choosing particular *ytop* values (or baseline value in Cfont). A wider range of '+sp' values are possible when a font fits into the minimum number of rows, thus minimizing the sum of *bottom blank points* + *top blank points*. In Cfont, the standard font baseline of 7 was chosen so that the descenders of the lower case letters in the 12 point font just fit into row zero. As a side benefit, minimizing the number of rows for a font minimizes the disk space required to store the font and also maximizes printing speed

by reducing the number of passes required to print the font.

E.3 MAXIMUM CHARACTER SIZE

The maximum character size is one inch wide by a little over one inch high. The figure below consists of two characters, a large box and a 12 point 'A' overprinted inside the box. The box is the largest character that can be handled by Efont.



The 'A' overprinted inside the box indicates the relative vertical position of the baseline of the distributed fonts. The horizontal location of a character is determined by the left and right margin values. The solid tick mark at the left of the box is vertical location zero and the dotted mark is location seven, the default baseline. The maximum height (i.e., the maximum *ytop* value) is 167, the lowest dot is at height -72, and the range of horizontal values is 0 to 119. The *replace* command will ignore any dots lying outside the box. The *generate* command in Cfont also clips characters to fit in the box. See the *baseline* command in Cfont for more information regarding placement of characters inside the vertical range.

Ascender -- The stroke of any lowercase letter which extends above the top of an x, e.g., b, h.

Baseline -- The location of the bottoms of most letters, e.g. A, x. Some characters descend below the baseline, e.g., g, p.

CP/M -- A popular operating system for microcomputers. The full, but never used name is Control Program/Microcomputer.

Descender -- The stroke of any lowercase letter which extends below the bottom of the baseline, e.g., g, p.

Dot -- An element of the dot pattern that creates the type image of a character. Also, often used in this manual as a unit of horizontal measure equal to 1/120th inch.

Em -- A unit of distance equal to the width of an 'M' in a font, or a unit of area equal to that distance squared. (Not used in this manual.)

Face -- Face typically refers to the boldness of a font, i.e., boldface type. We use face to differentiate between regular, bold, and italic forms of a given font style.

Fill -- To place as many words on a line as possible by drawing words from subsequent lines until the end of a paragraph is reached.

Flush left (or right) -- Type set to line up at the left (or right) margin.

Font -- The complete assortment of type of one size, face, and orientation.

Format -- The size, style, margins, printing requirements, etc. of a printed document.

Hairline -- The thinnest part of a letter.

Hershey database -- The Hershey character database is a set of characters created by Alan V. Hershey for the National Bureau of Standards. The Cfont program uses this database to construct new fonts.

Italic -- Normally, a style of type with letters slanting to the right, patterned on a compact manuscript hand. Used in this manual to differentiate any right-slanting font from its related vertical font.

Justify -- To space out lines by increasing the size of the spaces to make the right ends of the lines flush right.

Kerning -- To adjust the spacing between two characters to account for apparent spacing anomalies created by specific character adjacency combinations. For example, 'ox' looks more widely spaced than 'oo' even though the character margins are the same.

Letterspacing -- To place additional space between each letter of a word to

better balance a line or emphasize a word. Pfont does not do letterspacing (look at a newspaper, it usually looks terrible).

Logged disk -- Most microcomputers that run CP/M have more than one disk drive. CP/M has the notion of 'logged disk' to indicate on which drive to search for files. The currently logged disk is displayed as part of the CP/M prompt, e.g., 'A>' indicates that drive 'A' is the logged disk.

Pica -- Unit of measurement equal to $1/6^{\text{th}}$ inch.

Pixel -- Picture element or dot.

Point -- Unit of measurement approximately equal to $1/72^{\text{nd}}$ inch. Usually used for vertical measurement.

Roman -- Letters with thick and thin strokes and serifs.

Scale -- To change the size of an image by reducing or enlarging.

Sans serif -- Letters without serifs and usually having strokes of apparently even thickness.

Serif -- A smaller line used to finish off the main stroke of a letter, usually at the top or bottom of the stroke.

Stem -- The vertical stroke of a letter.

Style -- Fonts come in different styles, for example Roman, which has serifs, and Sans Serif, which does not. Other styles appearing in the manual are Old English and Script.

x-height -- The height of lowercase letters without ascenders or descenders.

All font files adhere to the following naming convention.

`<fontname>.fon`

The 'fon' extension is assumed by Pfont if no extension is given to the arguments to the '+fo' parameter. For example, '+fo romn12' is equivalent to '+fo romn12.fon'. `<fontname>` can be further refined to the following form:

`<style>{<face>}<size>{<supersub>}`

where `{<...>}` indicates an optional element. `<style>` examples are 'romn', 'sans', 'olde', and 'scrp' for Roman, Sans Serif, Old English, and Script respectively. `<face>` is optional and can be 'i' or 'b' for italic and bold. (We are a little loose about the term italic. We use it to modify the Roman style, which is not strictly accurate but is in common usage. Worse, we use it to describe "slanted" Sans Serif fonts. These are characters that are just slanted Sans Serif characters, but do not have the normal italic script-like curves.) `<size>` is the size of the font in printers points (generally the distance from the bottom of the lower case descenders to the top of the upper case letters, but sometimes adjusted for proper overall appearance). `<supersub>` is optional and is either 'p' for superscript or 'b' for subscript.

The currently available fonts are identified in the following table. An asterisk indicates a full, 95 ASCII printing character font and an 'x' indicates a partial font, but containing at least the upper and lower case alphabets.

	8	10	11	12	14	18	20	40	(size)
	rib	rib	rib	rib	rib	rib	rib	rib	(regular, italic, bold)
romn	*	***	***	***		***		x	
sans	*	*	*	**		*			
olde						x	x	x	
scrp				x	x	x	x	x	

The distribution package also includes 8 point subscript and superscript fonts, 'romn8b' and 'romn8p', and some fonts containing special characters, 'ff*.fon'.

Appendix 2 -- Font Descriptions and Samples

The following list gives the font height parameters as described in section E.2. Unless otherwise indicated all fonts contain the 95 ASCII printing characters, i.e., decimal ASCII codes 32 through 126 (see Appendix 8). Font samples appear on the following pages.

<u>filename</u>	<u>description</u>	<u>top row,</u>	<u>bottom row,</u>	<u>top blank,</u>	<u>bottom blank</u>
Romn8.fon	8 pt. Roman regular	0,0,0,1			
Romn10.fon	10 pt. Roman regular	1,0,6,0			
Romn11.fon	11 pt. Roman regular	1,0,6,0			
Romn12.fon	12 pt. Roman regular	1,0,5,0			
Romn18.fon	18 pt. Roman regular	1,-1,1,7			
Romn40.fon	40 pt. Roman regular	3,-1,0,0			alphabetics only
Romni10.fon	10 pt. Roman italic	1,0,6,0			
Romni11.fon	11 pt. Roman italic	1,0,6,0			
Romni12.fon	12 pt. Roman italic	1,0,5,0			
Romni18.fon	18 pt. Roman italic	1,-1,1,7			
Romnb10.fon	10 pt. Roman bold	1,0,6,0			
Romnb11.fon	11 pt. Roman bold	1,0,6,0			
Romnb12.fon	12 pt. Roman bold	1,0,5,0			
Romnb18.fon	18 pt. Roman bold	1,-1,1,7			
Romn8b.fon	8 pt. Roman subscript	0,-1,2,6			
Romn8p.fon	8 pt. Roman superscript	1,0,7,6			false top blank for better fit with standard fonts
Sans8.fon	8 pt. Sans Serif regular	0,0,0,1			
Sans10.fon	10 pt. Sans Serif regular	1,0,6,0			
Sans11.fon	11 pt. Sans Serif regular	1,0,6,0			
Sans12.fon	12 pt. Sans Serif regular	1,0,5,0			
Sans18.fon	18 pt. Sans Serif regular	1,-1,1,7			
Sansi12.fon	12 pt. Sans Serif italic	1,0,5,0			
Olde18.fon	18 pt. Old English	1,-1,1,6			missing characters [N]←{~
Olde20.fon	20 pt. Old English	2,-1,7,5			missing characters [N]←{~
Olde40.fon	40 pt. Old English	3,-1,0,0			missing characters [N]←{~
Scrp12.fon	12 pt. Script	1,-1,5,6			missing characters [N]←{~
Scrp14.fon	14 pt. Script	1,-1,4,5			missing characters [N]←{~
Scrp18.fon	18 pt. Script	1,-1,2,3			missing characters [N]←{~
Scrp20.fon	20 pt. Script	2,-1,7,3			missing characters [N]←{~
Scrp40.fon	40 pt. Script	4,-2,0,0			missing characters [N]←{~
Ff12.fon	12 pt. Special Chars	1,0,5,0			only a few characters, see the samples below
Ff20.fon	20 pt. Special Chars	2,-1,7,2			only a few characters, see the samples below
Ff40.fon	40 pt. Special Chars	4,-2,0,0			only a few characters, see the samples below

Roman style, regular face (Romn*fon)

- 8 A good character is for remembrance.
Ptah-Hotep, Instruction
- 10 A good character is for remembrance.
Ptah-Hotep, Instruction
- 11 A good character is for remembrance.
Ptah-Hotep, Instruction
- 12 A good character is for remembrance.
Ptah-Hotep, Instruction
- 18 A good character is for remembrance.
Ptah-Hotep, Instruction

40 A good character is for
PtahHotep Instructio

Roman style, italic face (Romni*fon)

- 10 *There is a spirituality about the face ... which the typewriter does not gener
Sir Arthur Conan Doyle, The Solitary Cyclist*
- 11 *There is a spirituality about the face ... which the typewriter does not generate
Sir Arthur Conan Doyle, The Solitary Cyclist*
- 12 *There is a spirituality about the face ... which the typewriter does not
Sir Arthur Conan Doyle, The Solitary Cyclist*
- 18 *There is a spirituality about the face ... which the t
Sir Arthur Conan Doyle, The Solitary Cyclist*

Roman style, bold face (Romnb*.fon)

- 10 **Boldness has genius, power and magic in it.**
 Goethe, Faust
- 11 **Boldness has genius, power and magic in it.**
 Goethe, Faust
- 12 **Boldness has genius, power and magic in it.**
 Goethe, Faust
- 18 **Boldness has genius, power and magic in it.**
 Goethe, Faust

Script (Scrp*.fon)

- 12 *Fancy may kill or cure.*
 James Kelly, Scottish Proverbs
- 14 *Fancy may kill or cure.*
 James Kelly, Scottish Proverbs
- 18 *Fancy may kill or cure.*
 James Kelly, Scottish Proverbs
- 20 *Fancy may kill or cure.*
 James Kelly, Scottish Proverbs
- 40 *Fancy may kill or cure.*
 James Kelly, Scottish Pr

Sans Serif, regular face (Sans*.fon)

- 8 *Simplicity of character is no hindrance to subtlety of intellect.*
 John, Viscount Morley of Blackburn, Life of Gladstone
- 10 *Simplicity of character is no hindrance to subtlety of intellect.*
 John, Viscount Morley of Blackburn, Life of Gladstone
- 11 *Simplicity of character is no hindrance to subtlety of intellect.*
 John, Viscount Morley of Blackburn, Life of Gladstone
- 12 *Simplicity of character is no hindrance to subtlety of intellect.*
 John, Viscount Morley of Blackburn, Life of Gladstone
- 18 *Simplicity of character is no hindrance to subtlety of intell*
 John, Viscount Morley of Blackburn, Life of Gladstone

Sans Serif, italic face (Sansi*.fon)

- 12 *Simplicity of character is no hindrance to subtlety of intellect.*
 John, Viscount Morley of Blackburn, Life of Gladstone

Old English (Olde*.fon)

- 18 **This is the sort of English
 up with which I will not put.
 Winston Churchill**
- 20 **This is the sort of English
 up with which I will not put.
 Winston Churchill**
- 40 **This is the sort of English
 up with which I will not
 Winston Churchill**

Special Characters (Ff*.fon)

Displayed as "the character you type" followed by an equal sign followed by the "character that is printed".

12	F=Œ	a=•	n=ñ	c=ç	y=y
	o=•	t=t	T=ᵀ	M=ᵐ	C=•
	R=ᵛ	*=•	↑=	{=ᵝ	
	=•	}=ᵞ			

20	F=ℱ	a=a	n=n	c=c	y=y
	o=•	t=t	T=ᵀ	M=ᵐ	A=ᵗ
	B=ᵝ	G=ᵞ	b=	g=ᵝ	h=ᵞ

40	F=ℱ	a=a	n=n	c=C	y=y
	o=O	t=t	T=ᵀ	M=ᵐ	

WordStar and other currently available personal computer text editing and word processing software, does not consider the different horizontal and vertical sizes of different fonts. Nor does it consider the widths of characters within a given font set. Therefore, the formatting features of these text editors are not too useful when printing with the *Fancy Font* system.

To overcome this deficiency in existing personal computer software, the *Fancy Font* system provides its own horizontal and vertical formatting features. However, the *Fancy Font* system does not move words from one line of text to another (the WordStar *word wrap* feature). Thus it is advisable to disable all formatting functions of your text editor other than those which assist in making line ending decisions.

The procedure for printing with *Fancy Font* is as follows:

1. Create a text file possibly containing embedded *Fancy Font* commands.
2. Print text using Pfont and possibly specifying a variety of printing features.

When creating the text file, the user must make line ending decisions, but should *not* use justification, centering, pagination or other formatting features of the text editing or word processing software.

If you are using WordStar, do the following:

1. Enter the *document* mode.
2. Turn justification *off* (Control-O J).
3. Make sure word wrap is *on* (Control-O W).
4. Set the right margin - approximate the number of characters per line based upon the Line Width (LW) and current font. 88 characters per line is a good approximation for +LW 6.5 and font Romn12; 100 characters per line for +LW 6.5 and font Romn10. For example, Control-O R 88 .
5. Use the text editor in the normal manner, however insert *Fancy Font* embedded commands to select fonts, center, justify etc. You will not actually see the results of these commands until you print the text. Do not use tabs, but use the *Fancy Font* ^t' command instead.

When finished editing, save your file in the normal manner. Print the *source* text file you just created by using Pfont. If you try to print a file created by the WordStar *Print* feature, use the Pfont -PG feature so that Pfont will not try to format your text on top of the formatting that was done by WordStar. This is not recommended since WordStar will not be able to properly paginate (remember that WordStar does not know the height or width of *Fancy Font* characters).

When WordStar hyphenates a word it places a "soft hyphen" in the document. The soft hyphen is actually ASCII code 31. All of the normal distributed fonts come with an actual hyphen installed for ASCII code 31 so hyphenated WordStar files should print properly. However, if you create your own font you must remember to install a hyphen at ASCII 31 in the new font.

GENERAL

characters in a font -- max. 128.

character size -- max. approximately 1 inch by 1 inch. See section E.3.

file names -- max. 8 characters in the root plus a 3 character extension and optional leading disk designator (e.g. 'a:a2345678.a23').

PFONT

printing parameters -- see the command and parameter summary appendix.

text file input line length -- max. 128.

text file line length after '+SU' substitutions -- max. 256.

interactive input line length -- max. 128.

CP/M command line length -- max. 128.

CFONT

scale factors -- x1 and y1 should be between -100 and +100 to avoid multiplication overflow.

<prog>.com These are executable programs. <prog> is one of Pfont, Efont, or Cfont.

<overlay>.ovl These are the overlay segments for either Pfont or Efont. These generally must reside on the CP/M logged in disk or on disk A. (Currently, the Efont overlays must reside on the logged disk.)

<file>.ff This is the conventional file name extension for *Fancy Font* (input) text files. The 'ff' extension is not required or specially recognized by any *Fancy Font* program; it is strictly an informative convention.

<file>.ffi This is the conventional file name extension for Pfont parameter files. For example, 'pfont <letter.ffi myletter.ff' would be useful if 'letter.ffi' defined the standard letter format, included a letterhead, etc.

hershey.chr This is the Hershey character database used by Cfont. If it is on the logged in disk or on drive A Cfont will find it automatically, otherwise Cfont will ask for the disk and file name specification. The Osborne distribution package provides 'hershey.chr' in two halves, 'hershey.chr' and hershey1.chr'.

fancfont.pro This is the terminal profile file. It is read in each time Pfont begins execution and can be changed by the user to configure the program for a specific terminal. The contents of the distributed file are as follows:

```
printer.type mx80,
number.of.columns.on.screen 80,
number.of.rows.on.screen 24,
backspace.key.input.code 8,
backspace.output.string 8 32 8;
```

The first parameter must be either 'mx80' or 'mx100' and needs to be changed to 'mx100' only if you need line lengths greater than eight inches. The rest of the text is just commentary and it is the order of the parameters that is important. The last parameter is a list of bytes to be output to the screen to destructively delete the previous character (the default is the character sequence: Control-H, SPACE, Control-H).

<fontname>.fon These are the font files. The 'fon' extension is assumed by Pfont if no extension is given to the arguments to the '+fo' parameter. For example, '+fo romn12' is equivalent to '+fo romn12.fon'. <fontname> can be further refined to the following form:

```
<style>{<face>}<size>{<supersub>}
```

where {<...>} indicates an optional element. <style> examples are 'romn', 'sans', 'olde', and 'scrp' for Roman, Sans Serif, Old English, and Script respectively. <face> is optional and can be 'i' or 'b' for italic and bold. <size> is the size of the font in printers points (generally the distance from the bottom of the lower case descenders to the top of the upper case letters, but sometimes adjusted for proper overall appearance). <supersub> is optional and is either 'p' for superscript or 'b' for subscript.

Appendix 6 -- Hershey Character Database

0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T		
	U	V	W	X	Y	Z	A	B	Г	Δ	E	Z	H	Θ	I	K	Λ	M	N	Ξ		
40	0	Π	P	Σ	T	Υ	Φ	X	Ψ	Ω				0	1	2	3	4	5	6		
60	7	8	9	.	,	:	;	!	?	'	"	°	\$	/	()		-	+	=		
80	x	*	.	'	'	→	#	&	π	A	B	C	D	E	F	G	H	I	J	K		
100	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	Г	Δ	E		
120	Z	H	Θ	I	K	Λ	M	N	Ξ	0	Π	P	Σ	T	Υ	Φ	X	Ψ	Ω	Δ		
140	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U		
160	V	W	X	Y	Z	a	b	c	d	e	f	g	h	i	j	k	l	m	n			
180	o	p	q	r	s	t	u	v	w	x	y	z	α	β	γ	δ	ε	ζ	η	θ		
200	ι	κ	λ	μ	ν	ξ	ο	π	ρ	σ	τ	υ	φ	χ	ψ	ω	a	b	c	d		
220	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x		
240	y	z	ℓ	∂	ε	θ	φ	ς				0	1	2	3	4	5	6	7	8		
260	9	.	,	:	;	!	?	'	"	°	\$	/	()		-	+	=	x	*		
280	.	'	'	→	#	&	□		⊥	∠	∴	♠	♥	♦	♣	♠	♣	♠	.	.	*	
300	▲	●	▲	^	∩	∪	∩	∪	,	,	,	S	∞	ℝ	9	—	/		\			
320	—	/	/		\	\	-	/		\	∩	∪	∩	∪	()	∩	>	∩			
340	∩	∪	∩	α	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩
360	+	x	*	•	■	▲	◀	▼	▶	★	†	‡	†	×	⊗	⊗	⊗	⊗	⊗	⊗	⊗	
380	△	†	‡	⊗	⊗	⊗	.	.	o	o	o	o	o	o	o	o	o	o	o	o	o	o
	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W		

1260 C D E F G H I J K L M N O P Q R S T U V
 1280 W X Y Z a b c d e f g h i j k l m n o p
 1300 q r s t u v w x y z a b c d e f g h i j
 1320 k l m n o p q r s t u v w x y z 0
 1340 1 2 3 4 5 6 7 8 9 . , : ; ! ? ' ' & \$ /
 1360 () * - + = ' " ° 0 1 2 3 4 5 6 7
 1380 8 9 . , : ; ! ? ' ' & \$ / () * - + = '
 1400 " ° A B C D E F G H I J K L M N O P Q R
 1420 S T U V W X Y Z a b c d e f g h i j k l
 1440 m n o p q r s t u v w x y z A B C
 1460 D E F G H I J K L M N O P Q R S T U V W
 1480 X Y Z a b c d e f g h i j k l m n o p q
 1500 r s t u v w x y z 0 1 2 3 4 5 6 7
 1520 8 9 . , : ; ! ? ' ' & \$ / () * - + = '
 1540 " ° H I J K L M N O P Q R S T U V W
 1560 X Y Z a b c d e f g h i j k l
 1580 m n o p q r s t u v w x y z

1260

C D E F G H I J K L M N O P Q R S T U V

1280

W X Y Z a b c d e f g h i j k l m n o p

1300

q r s t u v w x y z a b c d e f g h i j

1320

k l m n o p q r s t u v w x y z 0

1340

1 2 3 4 5 6 7 8 9 . , : ; ! ? ' ' & \$ /

1360

() * - + = ' " ° 0 1 2 3 4 5 6 7

1380

8 9 . , : ; ! ? ' ' & \$ / () * - + = ' "

1400

° Æ B C D E F G H I J K L M N O P Q R

1420

S T U V W X Y Z a b c d e f g h i j k l

1440

m n o p q r s t u v w x y z A B C

1460

D E F G H I J K L M N O P Q R S T U V W

1480

X Y Z a b c d e f g h i j k l m n o p q

1500

r s t u v w x y z 0 1 2 3 4 5 6 7

1520

8 9 . , : ; ! ? ' ' & \$ / () * - + = ' "

1540

° H B C D E F G H I J K L M N O P Q R

1560


S T U V W X Y Z a b c d e f g h i j k l

1580

m n o p q r s t u v w x y z

Part C -- Creating Fonts

The Hershey character database is a set of characters created by Alan V. Hershey for the National Bureau of Standards (Wolcott, NBS Special Publication No. 424). SoftCraft has adapted the database for use with the Epson printer and Appendix 6 displays all of these characters. The Cfont program allows the user to choose characters from the Hershey database and map them to ASCII character codes thereby constructing a font set for use with the rest of the *Fancy Font* programs (Pfont and Efont). As indicated in previous parts of the manual, a font set contains up to 128 characters, and usually will contain at least the 95 printing ASCII characters.

One simple scenario will illustrate the use of Cfont. Suppose you want to create a font containing only one character chosen from the Hershey database, say a bell character. Hershey character number 380 is a nice bell. Start up Cfont and use the *interactive mapping* command by typing 'm'. Cfont asks for the "Starting number of font set sequence". This requests the ASCII code of the character to map the bell to. In this example, we will map it to 'B' for bell and therefore type '66' as the ASCII code for 'B' (appendix 8 gives the ASCII codes for all characters). Next Cfont asks for the "Start of matching Hershey sequence". Here we type '380' to identify the bell character. Next Cfont asks for the "End of sequence (Hershey number)". Here we accept the default (380) because only one character is to be generated for the font set. (We could type a number greater than 380 and then Hershey character 381 would be mapped to 'C', 382 to 'D' and so on.) Executing the *generate* command by typing 'g' will cause the bell character to be constructed and associated with character 'B' and then Cfont will ask for the name of an output font file, call it 'bell.ff' say. The font file thus created can be used with Pfont to print the bell character in the middle of any other type of text. That is, if the Pfont command line 'Pfont text.ff -fo romn12 bell' is executed and the file 'text.ff' contains "The \f1B\f0 rang", it will print as "The  rang".

Other Cfont commands are the *scale factor* command, which allows the Hershey characters to be reduced or enlarged, the *save mapping* command, which allows interactively specified mappings to be saved on a file and subsequently loaded with the *load mapping* command, and the *print* command, which prints Hershey characters on the printer at the current scale factor for a preview of what the characters will look like when the font set is generated. The *print* command was used to print appendix 6. Some of the characters are truncated in Appendix 6 so that they fit on the line, but they can be created in full when generating a font with the *generate* command. All of the commands are described in detail below, and a summary of the commands appears in the final appendix.

C.1 COMMANDS

The Cfont command interface is exactly the same as that of Efont. All command names are single letters and do not need to be followed by a carriage return. Immediately upon typing of the command letter the program will prompt the terminal for any required parameters. The list of available commands can be obtained by typing a question mark in response to the "Command" prompt. Question mark generally does not provide help when parameters are being requested by the program and the

standard CP/M line editing characters are in effect. The command descriptions follow.

m -- Map Enter mappings from the terminal to specify the correspondence between the ASCII codes in the font to be generated and the characters in the Hershey database. Each font generated by Cfont can contain up to 128 characters corresponding to the ASCII codes 0 through 127. The *map* command allows a correspondence to be made between these ASCII codes and the characters in the Hershey database. The introduction to Part C gives a short example using the mapping command. After the 'm' is typed the program will prompt for the "Starting number of font set sequence", asking for an ASCII number in the font set to be generated. Next Cfont asks for the "Start of the matching Hershey sequence." This specifies a character number in the Hershey database (the database is listed in Appendix 6). Finally, Cfont asks for the "End of sequence (Hershey number)." Another Hershey character number is requested to indicate the sequence of characters to be mapped. The first Hershey character will be stored in the generated font as the ASCII code entered, and if the second Hershey number entered was greater than the first, then the next Hershey character stored as the following ASCII code, and so on. For example, to create a font with the upper case Old English letters mapped to the normal ASCII codes, use the mapping command with the following three inputs: (starting ASCII) 65, (starting Hershey) 1457, (ending Hershey) 1482. The mapping command can be used many times to specify multiple noncontiguous Hershey sequences. The same Hershey character cannot be mapped to more than one ASCII code. Other commands useful during the mapping phase are the *zap* command which removes mappings, and the *display* command which shows the currently specified mappings. See the *print* command for information on space characters in the Hershey database.

l -- Load Load a set of mappings from a file. A mapping specified with the *map* command can be saved to a file with the *save* command and subsequently loaded with the *load* command. If necessary, the *zap* and *map* commands can be used after the *load* command to override some of the loaded mappings. Multiple *load* commands are also permitted.

g -- Generate Generate the font. After the mappings between ASCII codes and Hershey numbers have been specified the font can be generated. Font generation can take several minutes for standard-sized fonts. Cfont will look for the file 'Hershey.chr' on the logged drive and, failing that, on drive A. If it isn't found then the terminal will be prompted for the proper 'Hershey.chr' file name. A temporary font file is constructed on the logged disk so there must be enough disk space on the logged disk to accommodate a font-sized file (approximately 10K for a 12 point, 95 character font). A period is displayed on the terminal for each character as it is constructed during font generation. After all of the characters are constructed from the Hershey database the terminal will be prompted for the name of the (final) font file. Osborne note: The 'Hershey.chr' database is larger than the capacity of a single disk, so the file has been split into two parts, 'Hershey.chr' and 'Hershey1.chr'. When 'Hershey1.chr' is on drive A or the logged drive then the continuation should happen automatically. If 'Hershey1.chr' cannot be found Cfont will prompt for a file name. At this point a new disk containing

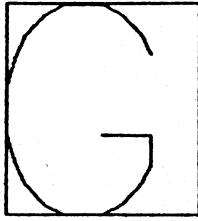
'Hershey1.chr' can be inserted in the nonlogged drive (the logged disk still contains the temporary font file so do not remove that disk).

z -- Zap Remove some of the current, previously specified mappings. The terminal will be prompted for a range of ASCII:Hershey mappings to be removed.

p -- Print Print a range of Hershey characters on the printer. The terminal is prompted for a range of Hershey characters and then these characters are printed on the printer using final draft mode and the prevailing scale factors. Appendix 6 was printed using this command with all Cfont default values. See the *generate* command for a description of the 'Hershey.chr' file access requirements. Some of the characters in Appendix 6 are space characters, e.g., 50, 51, and 52. Usually the highest numbered character in a short sequence like this is the widest space character and the smallest numbered is the narrowest. The *margins* command in Efont is generally a more convenient way to precisely specify space characters than trying to identify the appropriate space characters in the Hershey database. Depending upon the prevailing scale factors and baseline characters may be clipped at the top or the bottom when they are printed. This is to allow printing of a useful number of characters on a page. The characters will not necessarily be clipped when they are *generated*.

f -- Factors Set the scale factors. The characters in Appendix 6 can be scaled up or down (reduced or enlarged) by setting the scale factors. As characters are constructed by the *generate* command and the *print* command the horizontal components of the character are multiplied by a fraction, $x1/x2$. Similarly, the vertical components are multiplied by a fraction $y1/y2$. This command allows the user to set each of the values $x1$, $x2$, $y1$, and $y2$. For comparison, Appendix 6 used values of $x1=1$, $x2=1$, $y=2$, $y2=1$, and generally speaking, the vertical fraction should be about twice that of the horizontal fraction. For example, to make characters one and one half times as wide as, but the same height as those in Appendix 6 set $x1$ to 3 and $x2$ to 2 (i.e., there is no floating point capability in Efont). The maximum size of a character is one inch wide by a little over one inch high and the character should fall in the box described in section E.3 or else the character will be clipped. See the example in the *baseline* command description. Characters can be mirrored by setting $x1$ and/or $y1$ to negative values.

b -- Baseline Set the baseline of the characters to be generated. Any two fonts printed on the same line must have been created with the same baseline if the bases of the characters are supposed to line up. The default baseline is normally 7, but can be modified to more efficiently store a particular font, or to generate a specialized, very large font. For example, you could create a very large font (about 120 points) if it consisted of only upper case letters. After reading section E.3 you would see that a baseline of -72 might allow creation of such a font. A 120 point 'G' (Hershey character 95) appears below in the maximum character box (see E.3). This was created with baseline -72, $x1=6$, $x2=1$, $y1=34$, $y2=3$.



(The 'G' is actually 80 points high, but the size of a font measures from the bottom of the descenders to the top of upper case characters, so the 'G' would be considered to be a member of a 120 point font.)

d -- Display Display the current mappings, the scale factors, and the baseline value on the terminal.

s -- Save Save the current mappings to a file so that they can be loaded with the *load* command. The terminal will be prompted for a file name and the file thus created is a normal ASCII file and can be edited with a text editor if desired.

q -- Quit Exit Cfont.

There are only two types of data files, the font files and the Hershey character database file.

FONT FILES

The font files have the following format.

<u>bytes</u>	<u>description</u>
0-38	Id string, "Translation by Pacific Software Systems", (a division of SoftCraft responsible for font creation). No trailing zero.
39	Font file version number.
40-41	Number of "ink" data bytes in the file (see below). Low byte, high byte.
42-n	Commentary text terminated with a zero. Variable length.
n+1	A constant 2.
n+2	A constant 3.
n+3	Top blank points (see section E.2 and the Efont <i>auxinfo</i> command).
n+4	Bottom blank points.
n+5	Number of characters defined in the font, including spaces.
n+6 - n+15	Reserved but currently unused.
n+16 - n+21	These six bytes define the height, width, and relative vertical and horizontal locations of the first character in the font. Immediately following these bytes are six similar bytes for the next character, then six for the next, and so on until all characters are defined (byte n+5). The characters are arranged in increasing ASCII order. The six bytes are defined as follows:
1	ASCII number.
2	Box width (left margin + right margin + ink width).
3	Left margin (the starting location of the ink inside the box).
4	Ink width.
5	Ink height in rows (8 points, 24 dots per row).
6	Row number of ink bottom. The top row is this value plus the ink height. Section E.3 describes these parameters.

HERSHEY FILE

The Hershey character database has the following format.

The first two bytes in the file are zero and are not used. Each character in the database is defined by a list of plotting directives and is best explained in terms of a pen plotter. The first two bytes in the character description are the width bytes and are not used by Cfont except for space characters. Each subsequent plotting directive is a two-byte entry that is either the coordinate of a new pen location or a special indicator. The special indicators are octal value 40000 specifying "pen up" and 0 indicating the end of the character. Values other than these are coordinates to move the pen to, drawing a line if the pen was down, or just moving if the last value was "pen up" (any pen movement implies that the pen is lowered at the end of the movement regardless of its state before the move). The coordinates are expressed as 'y_{data}' and

'x_{data}' bytes, in that order, and are plotted by Cfont at 'x_{coord}' and 'y_{coord}' according to the following equations.

$$x_{\text{coord}} = (x_{\text{data}} - 64 * x1/x2)$$

$$y_{\text{coord}} = (68 - y_{\text{data}} * y1/y2)$$

where x1, x2, y1, and y2 are the scale factors as described in the Cfont *factors* command. x_{coord} should be centered around 0 for the best possible allowable range of scaling factors, and the margins will be automatically set to 1 (they can be changed with Efont). y_{coord} equal to zero will be set at the baseline as specified by the Cfont *baseline* command (default value of seven). The above equations are modified slightly for character numbers greater than 88. The constant, 68, in the equation for y_{coord} is as follows: characters 0 through 88, 68; 89 through 396, 73; 397 through 672, 70; 673 on, 73. The left and right margins are always set to one except as follows: characters 0 through 215, 1; 216 through 241, 0; 242 through 1075, 1; 1076 through 1101, 0; 1102 on, 1.

Appendix 8 -- ASCII Conversions

char	dec	oct	hex	char	dec	oct	hex
0	48	60	30	@	64	100	40
1	49	61	31	A	65	101	41
2	50	62	32	B	66	102	42
3	51	63	33	C	67	103	43
4	52	64	34	D	68	104	44
5	53	65	35	E	69	105	45
6	54	66	36	F	70	106	46
7	55	67	37	G	71	107	47
8	56	70	38	H	72	110	48
9	57	71	39	I	73	111	49
:	58	72	3A	J	74	112	4A
.	59	73	3B	K	75	113	4B
<	60	74	3C	L	76	114	4C
=	61	75	3D	M	77	115	4D
>	62	76	3E	N	78	116	4E
?	63	77	3F	O	79	117	4F
				P	80	120	50
				Q	81	121	51
				R	82	122	52
				S	83	123	53
				T	84	124	54
				U	85	125	55
				V	86	126	56
				W	87	127	57
				X	88	130	58
				Y	89	131	59
				Z	90	132	5A
				[91	133	5B
				\	92	134	5C
]	93	135	5D
				^	94	136	5E
				_	95	137	5F
				`	96	140	60
				a	97	141	61
				b	98	142	62
				c	99	143	63
				d	100	144	64
				e	101	145	65
				f	102	146	66
				g	103	147	67
				h	104	150	68
				i	106	161	69
				j	106	162	6A
				k	107	163	6B
				l	108	164	6C
				m	109	165	6D
				n	110	166	6E
				o	111	167	6F
				p	112	160	70
				q	113	161	71
				r	114	162	72
				s	115	163	73
				t	116	164	74
				u	117	166	75
				v	118	166	76
				w	119	167	77
				x	120	170	78
				y	121	171	79
				z	122	172	7A
				{	123	173	7B
					124	174	7C
				}	125	175	7D
				~	126	176	7E
				DEL	127	177	7F

FORMATTING INDICATORS IN TEXT FILES.

In the indicators below <1int>, <3int>, and <4int> are decimal integers of exactly 1, 3, and 4 digits, respectively. For example, to print ASCII character code 1, write '\d001'.

\f<1int>	Switch to font number <1int> (as specified by +FO).
\u	Start or stop (toggle) underlining.
\c	Center the current line.
\r	Flush the current line against the right margin.
\j	Turn on line justification.
\k	Turn off line justification.
\b	Break. Don't justify current line.
\p (fL)	Move to top of next page.
\v<4int>	Move down <4int>/72 inches from the current position.
\w<4int>	Move to vertical position <4int>/72 inches from the top of the page.
\t (fI)	Move to next horizontal tab stop (as specified by +TB).
\h	Space horizontally (right) by 1/120 of an inch.
\a<4int>	Tab to horizontal position <4int>/120 inches from the left margin.
\i<3int>	Space right <3int>/120 inches from the current position (<3int> must be between 000 and 255).
\s<1int>	Substitute string number <1int> (as specified by +SU).
\s#	Substitute the current page number.
\sf	Substitute the current file name.
\d<3int>	Print ASCII digit <3int> (in the range 000 to 127).

EFONT COMMANDS

l	Load a font set for editing or inspection.
s	Save the possibly modified font set.
p	Print a set of expanded characters to the printer.
e	Edit a set of characters by creating a text file of each.
r	Replace a set of characters with their edit file specifications.
d	Display an expanded character on the terminal screen.
m	Modify the margins for a set of characters.
t	Test the font by printing a string on the printer.
z	Zap (remove) a set of characters from the font set.
u	Move characters up or down when the 'e' command is used.
a	Display and modify the auxiliary font information.
q	Quit.

CFONT COMMANDS

m	Enter mappings from the font set to the Hershey database interactively.
l	Load a set of mappings from a file.
g	Generate a font set with the current mappings.
z	Zap (remove) previously specified mappings.
d	Display current mappings and scale factors.
p	Print a set of Hershey characters to the printer.
f	Set the scale factors for enlarging or shrinking characters during generation.
b	Set the baseline for moving characters up or down during generation.
s	Save the current mappings to a file.
q	Quit.

PFONT PARAMETERS (default values are in square brackets; tX means Control-X)

FI Names of files to print - List of up to 15 File Names - [<No Value>]
 (use filename 'tty:' for terminal input and 'loop:' to repeat files)
 FO Font file names - List of up to 10 Font File Names - [<Off>]
 TB Tab stops, explicit or an increment (one argument implies tab increments)
 - List of up to 15 Real Numbers between 0 and 8 - [0: .8]
 RD Rough draft mode - Integer between 0 and 2 (2 is roughest) - [<Off>]
 HL Heading line for top of page - String - [<Off>]
 FL Footing line for bottom of page - String - [<Off>]
 SU Strings to substitute for \s indicator - List of up to 10 Strings - [<Off>]
 LM Left margin - A Real Number between 0 and 8 - [1]
 NF No header or footer text on first page - No Argument - [<Off>]
 SP Spacing between lines - A Real Number between 0 and 150 - [.045]
 TM Top margin between page top and text
 - A Real Number between 0 and 150 - [.75]
 BM Bottom margin between text and page bottom
 - A Real Number between 0 and 150 - [.75]
 HM Heading margin between page top and header
 - A Real Number between 0 and 150 - [.25]
 FM Footing margin between footer and page bottom
 - A Real Number between 0 and 150 - [.25]
 PL Total page (paper) length - A Real Number between 0 and 150 - [11]
 LW Line width from left margin - A Real Number between 0 and 8 - [6.5]
 PP Pause between pages for insertion of paper - No Argument - [<Off>]
 SD Display on screen instead of printing - No Argument - [<Off>]
 EP Use Epson fonts - No Argument - [<Off>] ("hardwired" font mapping, i.e.,
 \f0 is normal, \f1 is compressed, \f2 is expanded, etc.)
 PF Process formfeeds (Control-L's) - No Argument - [<On>]
 PN Initial page number for \s# - An Integer between 1 and 9999 - [1]
 FP First page to begin printing - An Integer between 1 and 9999 - [<Off>]
 LP Last page to print - An Integer between 1 and 9999 - [<Off>]
 CF Concatenate files without page break - No Argument - [<On>]
 CI Command indicator - String - ["\
 IS Initialization string inserted at the beginning of each file - String - [<Off>]
 PG Enable vertical margins, pagination, headers, etc. - No Argument - [<On>]

SPECIAL CHARACTERS

? help (help is available at all levels of Pfont interaction)
 & display all of the current values
 < process a file containing parameter specifications
 <carriage return> end a list or start printing
 tV print a general help file to the screen
 tC quit
 tU line delete
 BS (tH) character delete (this can be changed by editing FancFont.pro)

PFONT "DURING PRINTING" COMMANDS

tP Abort printing of the current page and go on to the next one.
 tF Abort printing of the current file and go on to the next one.
 tC Abort printing after the current line is finished and quit.