XPRINT

TEXT FORMATTER FOR THE COLOR COMPUTER USING OS9

by Eric Dokken

Microtech Consultants Inc.

TABLE OF CONTENTS

Installation

Installing XPRINT consists of copying XPRINT itself (XP) from the XWORD disk supplied into your execution directory. In addition you must copy the program INDEX (program used to format index entries) into your execution directory. Also you must either copy one of the supplied initialization text files (which are found in the directory called INITFILES on the XWORD disk), or create one yourself, see chapter 4 for more information on initialization files.

To copy XPRINT to your system disk do the following:

For multi drive systems

Put the XWORD disk into drive 1 and your system disk into drive 0, and enter the following:

```
COPY /D1/XP /D0/CMDS/XP
```

For single drive systems

With your system disk in drive 0 type the following:

```
CHX /D0/CMDS
LOAD COPY
```

Now put the XWORD disk into drive 0 and type:

```
COPY /D0/XP /D0/CMDS/XP -S
```

Follow the prompts, putting the XWORD disk in drive 0 when the prompt asks you to insert the source disk, and putting your system disk in the drive when asked to insert the destination disk.

Now copy the file INDEX on the XWORD disk into your execution directory the same way you copied XP.

Now you must copy an initialization file from the directory called INITFILES on the XWORD disk into your execution directory. This can be done in the following manner (assume the initialization file wanted is for the Gemini-10X printer INIT.GEM):

For multi drive systems

Put the XWORD disk into drive 1 and your system disk into drive 0, and enter the following:

```
COPY /D1/INITFILES/INIT.GEM /D0/CMDS/INIT.XP
```

For single drive systems

with your system disk in drive 0 tyep the following:

```
CHX  /D0/CMDS
LOAD  COPY
```

Now put the XWORD disk into drive 0 and type:

```
COPY  /D0/INITFILES/INIT.GEM  /D0/CMDS/INIT.XP  -S
```

Follow the prompts, putting the XWORD disk in drive 0 when the prompt asks you to insert the source disk, and putting your system disk in the drive when asked to insert the destination disk.

Please read chapter 4 about editing the initialization file.

Execution

XPRINT is executed by entering a command line with the following form:

XP <filename> [options]

Where <filename> is the text file that is to be printed. XPRINT will look for <filename> in the current data directory unless a full pathlist is specified.

Options include the following:

-E Disable error checking pass. Only one pass is made through the file printing during that pass. Normally an error checking pass is made to make sure no errors are in the file before anything is printed.

-I <initfile> Use <initfile> as the initialization file. Normally INIT.XP is used as the initialization file, this option enables you to specify a different file. <initfile> will be looked for in the current execution directory unless a full pathlist is specified. <initfile> is a text file of only format lines (each line must start with a '.'). The execution bit must be set for this file, it can be set by the command ATTR <initfile> E

-O <outfile> Send the formatted text to <outfile> instead of the printer. Normally XPRINT sends output to the printer device descripter /P. This option lets you specify the output to go to a file or another device.

-Px,y Only print text from page number x to page number y. This option lets you specify a range of pages that you want printed. Either x or y may be left out, if x is left out printing will start at the beginning, if y is left out printing will continue to the end of the text.

Some examples:

-P5,8 Print pages 5 to 8.

-P13 Print starting at page 13 to the end of text.

-P,8 Print from beginning of text to page number 8.

-F=<formatline> Initial format line. This option must be the last option on the command line. <formatline> is a format line that will be evaluated after the initfile is evaluated and before any of the text file is formatted. This lets you change some of the defaults from the command line.

There are some restrictions on what you can put in <formatline> because of the OS-9 shell. For example you can not use the characters '#' '>' '<' ';' '!' or the shell will intrepret these characters and cause unpredictable results.

Here are some examples using the -F option:

-F=lm=20,rm=50,lf=j

Sets left and right margins, and sets line format to justified.

-F=fi="FILE"

Read include file named FILE before doing rest of text.

Do not use the OS-9 memory option (#x) when executing XPRINT. XPRINT will automatically expand its data memory when it needs it, specifying more memory will only give it less memory to expand to.

Here are some example command lines and the results of them:

XP REPORT

Format file called REPORT and print formatted results to printer.

XP FILEXYZ -I INIT.PRINTER2

Format file called FILEXYZ using file called INIT.PRINTER2 from the current execution directory for the initialization file.

XP TESTFILE -E -O OUTPUT

Format file called TESTFILE, do not do error checking pass, send formatted output to file called OUTPUT in current data directory.

XP FILEZ -F=LM=20,RM=50

Format file called FILEZ, first setting the left margin to 20 and the right margin to 50

2. THE FORMAT LINE

Practically all of XPRINT's functions are controlled with commands in format lines. This makes the format line an important part of the XPRINT formatter.

This manual has divided the format commands into the following six groups. If a command wasn't easily identified as belonging to one of these groups it was put into the formatting commands group.

* Margin Commands: LM, RM, TM, BM, PL
* Formatting Commands: LF, LS, PG, OP, EP, PP, SP, FO, NL, FI, PE, HS, GH, LA
* Indenting and Tabbing Commands: IN, RI, DI, ST, TF
* Header/Footer & Page Numbering Commands: LH, RH, HE, HL, HF, PN, NT
* Printer Code Commands: N, SN, A, SA, C, SC, S, SS, PI, BS, D+, D-, U+, U-, S+, S-, B+, B-, O+, O-, E+, E-, I+, I-, DA, 0-9, S0-S9
* Proportional Spacing Related Commands: CT, PJ, CS, TT, TR, PS, FS

Some of the proportional spacing commands can be used with printers that do not have proportional capabilities, so that section should be read even if you are not using a proportional printer.

Format Line Structure

A format line is any line that starts with a period '.' in the very first column of the line. The rest of the line then has zero or more format commands, with each command separated by a comma ','.

Format commands are one or two characters usually followed by an equal sign '=' and some value. This command sets the value of some feature or function of XPRINT to the value specified. For example, LM=10 will set the left margin to 10. A couple of the commands, such as PG, do not need a value, so they do not have an equal sign or value following them.

The one or two character combinations are mnemonically oriented to make it easy to remember what each command does. For example, LM RM TM BM are the commands for left margin, right margin, top margin and bottom margin.

Format commands can be specified either in upper or lower case characters. For example, LM=10 Lm=10 lM=10 and lm=10 all set the left margin to 10. Spaces have no affect on format commands, they can be used to separate commands. For example, the format line:

```
.LM = 10 , RM=70   ,  TM  =  6,  BM = 59
```

is equivalent to the format line:

```
.LM=10,RM=70,TM=6,BM=59
```

If a format command is assigned a numeric value, that value can be either specified in decimal, or it can be hexadecimal by prefixing the number with a dollar sign '$'. For example, RM=60 RM=$3C and RM=$3c are equivalent.

Placing an asterisk '*' in a format line wherever a command could go will make the remaining line a comment, it will not be interpreted. For example:

```
.* This is a comment line
.LM=10,* Set left margin to 10
.LM=10,RM=70,TM=6,BM=59,  * set up margins
```

Note that if there are any commands on the line, there must be a comma between the last command and the asterisk.

Margin Commands

The margin commands are the commands that set the margins of the printed page. The margins are left margin, right margin, top margin, and bottom margin. In addition, the page length is set using the page length format command.

These margin commands are powerful in that they can be set differently for even and odd pages to enable you to get text formatted in book format, with wide left margins on odd pages, and wide right margins on even pages. The output from you printer can be copied onto double sided paper to give you a very professional look. That is how this manual was prepared, notice the margins are wider on the left side for odd pages and wider on the right side for even pages.

The margins are set differently for even and odd pages by putting an 'e' or an 'o' after the margin command before the equal sign. For example, LME=10 will set the left margin of even pages to 10. If there is no 'e' or 'o' after the command then both even and odd pages are set to that margin.

Since many different character sizes can be used with XPRINT, including proportional characters, margins are not measured in characters. Instead, margins are measured by tenths of an inch. For example, specifying 10 for the left margin will set the left margin at 1 inch. Similarly, LM=5 sets the left margin at 1/2 inch.

The following are the margin commands:

LM{e,o}=<0-140> Left Margin (default: 8)

Set the left margin to the value specified. If 'e' or 'o' is not specified the left margin of both even and odd pages are set to the value. If 'e' is specified the left margin of even pages is set to the specified value, if 'o' is specified the left margin of odd pages is set the specified value.

Examples:

```
LM=10
LME=8
LMO=15, LME=10
```

The first example sets the left margin of both even and odd pages at one inch. The second example sets the left margin of even pages to .8 inches. The third example sets the left margin of odd pages to 1.5 inches and the left margin of even pages to one inch.

RM{e,o}=<0-140> Right Margin (default: 71)

Set the right margin to the value specified.

TM{e,o}=<0-255> Top Margin (default: 6)

Set the top margin to the value specifed. If 'e' or 'o' is specified then only the even or odd page top margin is changed. Line numbers are numbered from 0 to page length - 1. For example, with a standard 66 lines per page, line number 0 is the first line of the page, while 65 is the last line of the page.

The default top margin is line 6 which leaves one inch (6 lines) at the top (lines 0,1,2,3,4, and 5).

BM{e,o}=<0-255> Bottom Margin (default: 59)

Set the bottom margin to the value specified. If 'e' or 'o' is specified then only the even or odd page bottom margin is changed. The default bottom margin is 59 which leaves a one inch margin at the bottom of a standard 66 lines per page page.

PL=<3-255> Page Length (default: 66)

Set the length of the page in number of lines. The default is 66 lines which corresponds to an 11 inch page with 6 lines per inch.

Formatting Commands

LF{e,o}=(l,c,r,j,a) Line Format (default: j)

Set the line format mode. You have the following options:

L - Left justified. All lines are printed flush left, with the right edge jagged.

C - Centered. All lines are printed centered on the line.

R - Flush Right. All lines are printed with their right edges flush right, and their left edges jagged.

J - Justified. All lines except for those ending in a carriage return are printed with both left and right edges flush with the margins. Spaces are added to fill the lines.

A - All lines justified. All lines, including lines ending in a carriage return are printed with both left and right edges flush with the margins.

Examples:

```
.LF=J
.LFE=L, LFO=R
```

The first example will print the lines justified. The second example will print the lines on even pages flush left and the lines on the odd pages flush right.

LS=<1-255> Line Spacing (default: 1)

Set the line spacing the the specified value. For example, LS=1 will produce single spaced output, LS=2 will produce double spaced output.

PG begin new PaGe

This command will stop printing on the current page and position itself to start printing at the top of the next page.

If XPRINT is currently on the first line of a page when this command is executed, this command will have no effect. This takes care of the problem of having a completely blank page because a page break happens to come after the last line on a page.

If you are at the top of a page and you want a blank page, put in a carriage return first to print a blank line, then put in the PG command.

OP begin new Odd Page

This command is similar to PG in that it will position itself to start printing at the top of the next odd page.

EP begin new Even Page

This command is the same as OP except text will start printing at the top of the next even page.

PP=(y,n) Page Pause (default: n)

This command will enable or disable the automatic page pause feature of XPRINT. Setting PP=N turns page pause off, each page is printed one after the other with no pausing.

Setting PP=Y will turn page pause on. If page pause is on, after each page is printed the printing will stop and the message:

Press enter to start next page

will be displayed. Printing will be continued once **<ENTER>** is pressed. Page pause should be used if your printer has to be fed its paper a single sheet at a time.

Examples:

```
.PP=Y
.PP=N
```

The first example will turn page pause on and the second example will turn page pause off.

SP=<0-255> blank SPace

This command will print the specified number of blank lines in your text. For example, SP=5 will cause a space of 5 blank lines to be created. The current line spacing has no effect, SP=5 is the same whether the text is double spaced (LS=2) or single spaced (LS=1).

If there is not enough space remaining on the page when SP is executed, a new page is started as if the PG command had been executed. For example, if printing is currently at line number 55, the bottom margin is set at 59 and SP=7 is encountered by XPRINT, then a new page will be started because only 5 lines remain on the page, lines 55-59. The extra space will not be put at the top of the new page, the printing will start right at the top.

NL=<0-255> Need Lines

This command is used to make sure there is the specified number of lines remaining on the page, if not a new page command is executed. This command is used so lines can be kept together without being split from one page to another. For example, NL=3 can be used before a paragraph so at least three lines of that paragraph will appear on the page, one line won't be printed with the rest of the paragraph on the next page.

This command can also be used with SP in order to make room for figures and tables within your text without having it break from one page to the next. For example, if you need 10 lines of space for a figure, with a one line caption saying "Figure 1" you could have the following:

```
.NL=11
.SP=10
.LF=C
Figure 1
```

The NL=11 is set to 11 to account for 10 lines of space and 1 line of caption. The SP=10 spaces 10 lines of space for the figure, LF=C sets the line format to centered to center the caption, "Figure 1" is the caption.

FI="<filename>" File Include

This command enables you to include other files within the file currently being formatted. When this command is encountered XPRINT will start reading the text from the specified file. When the end-of-file of that file is reached, the text is read from the original file where it left off.

This command must be the last command on a line, no format commands can follow it on the same line. <filename> is taken from the

current data directory unless a full pathlist is specified.

File includes can be nested, a file that is included in another file can include another file. The limit to nesting is only limited by the data memory available and OS-9's limit to the number of open files. There is no limit to the number of sequential file includes in a file

Example:

```
.FI="headings"
.FI="chapter1"
```

These two commands will cause XPRINT to operate on the contents of a file called HEADINGS and then on a file called CHAPTER1.

HS=<character> Hard-Space character (default: |)

This command allows you to change the character that represents a hard-space.

A hard-space is character that when output to the printer will be printed as a space, but within XPRINT it will not be treated as a space. Extra spaces will not be added at hard-spaces to fill out lines, and lines will not break at a hard space like they do at normal spaces.

Hard-spaces are used when you want a certain space between words and you don't want any extra spaces inserted between the words. For example, if you have the name John Smith in your text and you only want one space between the first and last name you would use a hard-space because with a normal space, space might be inserted there to fill a line for justification. A hard-space will make the name John|Smith appear to XPRINT as one word so it won't be broken, but the '|' will be printed as a space.

You can not specify the hard-space character to be a space, A-Z, a-z, or a ','. If you don't want a hard-space character you can follow 'HS=' immediately with a ',' or a carriage return.

Examples:

```
.HS='
.HS=@
.HS=
```

The first example sets the hard-space character as '', the second sets the hard-space character as '@', the third example specifies that there is to be no hard-space character.

GH=<character> Ghost Hyphen character (default: ~)

This command allows you to change the character that represents the ghost hyphen.

A ghost hyphen is a character put within a word that will appear as a hyphen if the word has to be broken up because it is too long to fit on the line. But if the word does not have to be broken up the ghost hyphen will not appear at all. More than one ghost hyphen may appear in a word, with at most on of them actually appearing.

You can not define a ghost hyphen to be a space, A-Z, a-z, or a ','. If you don't want a ghost hyphen to be defined, follow GH= with ',' or a carriage return.

Examples:
.GH='
.GH=@
.GH=

The first example sets "'" as the ghost hyphen character. The second example sets '@' as the ghost hyphen character and the third example says there will be no ghost hyphen character.

LA=(y,n) Line Adjust (default: n)

Line Adjust is mainly used if you are formatting text that was created with a line editor. With a line editor each line is terminated by a carriage return. But XPRINT expects just to see one long string of characters with a carriage return at the end of each paragraph, and it formats those characters into the proper lines.

Line Adjust solves the problem by converting carriage returns to spaces, and thereby converting lines into a sequence of characters. Carriage returns are converted to spaces except when the following cases occur:

1. If the first character following a carriage return (first character of the next line) is a carriage return, space, or period, then that carriage return is left alone, it is not converted.

2. If a carriage return is the only character on a line, then it is not converted to a space.

FO=<n,a,c,p) change FOnt (default: n)

This command changes the character font. There are four choices: N-normal, A-alternate, C-condensed, and P-proportional.

Indenting & Tabbing Commands

XPRINT has powerful indenting and tabbing features. These features let you indent or tab to a specified point on a line that can be an absulute value measured by distance from the margin, or by a relative value from the current indent value.

.IN=<tab/indent value> INdent

This command indents in from the left margin the amount specified. The <tab/indent value> can be a numeric value (tenths of inches), a previously set tab stop (t#, #=0-9), a specified number of dot widths (x<value> dot widths are size of small space), a size of a specified string ("<string>"), or a combination of additions and subtractions of these types of values. The <tab/indent value> can be relative to the current indent value by prefixing it with a plus or minus sign, otherwise the value will be an absulute value as measured from the left margin.

Some examples:

.IN=10 indent 1 inch from left margin
.IN=5 indent 1/2 inch from left margin
.IN=+5 indent 1/2 inch more than previous indent
.IN=-10 indent 1 inch less than previous indent
.IN=T0 indent to value of tab stop number zero
.IN=+"ABC" increase indent by the size taken up by the characters A,B,C
.IN=30-"xyz" indent 3 inches minus size taken up by the characters x,y,z
.IN=25+x5 indent 2.5 inches plus 5 dot widths

The <tab/indent value> must be greater than or equal to zero. You can not indent out left of the left margin.

.DI=<tab/indent value> Delayed Indent
.DI

Delayed indent is how hanging indents are acheived in XPRINT. The value assigned to DI is the indent value that will take effect after one line is printed. For example, specifying DI=+10 will cause the next line to be printed at the original indent setting, but the line after it and all remaining lines will be indented 1 inch more, until the indent is changed again.

The value assigned to DI is actually assigned to IN after 1 line. Specifying an indent using IN will disable a delayed indent waiting to take effect. So the command line

.DI=20,IN=5

will cause an indenting of 1/2 inch, the DI has no effect because the IN

disables it. But the line

```
.IN=5,DI=20
```

will cause the next line to be indented 1/2 inch but all lines after it will be indented 2 inches.

The main use of DI is for creating hanging indents. For example, if you have some numbered points, you would want the numbers to hang out to the left with the text lined up to the right of the numbers. For example, the following text

```
.IN=5,DI=+"1. "
1.|This is point number one. Notice how the second line lines up with the
first line with the number "hanging" out to the left.
.IN=5,DI=+"2. "
2.|This is the second point. Notice how the indent value had to be reset to
bring the hanging indent back out.
.IN=5,DI=+"3. "
3.|This is the third point. Again the indent and delayed indent had to be
reset.
```

when formatted will look like:

1. This is point number one. Notice how the second line lines up with the first line with the number "hanging" out to the left.
2. This is the second point. Notice how the indent value had to be reset to bring the hanging indent back out.
3. This is the third point. Again the indent and delayed indent had to be reset.

Notice that a hard-space is put into the spaces separating the numbers and the text. This is because spaces added to right justify the text might make the hanging indents not line up correctly. So you should use hard indents up to the delayed indent point when using justified text.

For hanging indents the indent and delayed indent have to be reset each time. Since they are usually reset to the same previously set values, a short hand way of resetting them is to just specify 'DI' with no equal sign and no <tab/indent value>. This will reset both IN and DI to the values they had when the last DI was executed. For example, the previous example could have been specified as:

```
.IN=5,DI=+"1. "
1. This is point number one. Notice how the second line lines up with the
first line with the number "hanging" out to the left.
.DI
2. This is the second point. Notice how the indent value had to be reset to
bring the hanging indent back out.
.DI
3. This is the third point. Again the indent and delayed indent had to be
reset.
```

The second and third DI's reset IN to 5 and DI to +"1. ".

RI=<tab/indent value> Right Indent

This command will indent the right margin the specified amount. The larger the value the more towards the center of the line the margin goes. A RI of 0 sets the right indent at the right margin.

ST#=<tab/indent value> Set Tab stop

This command will set the specified tab stop to the specified <tab/indent value>. The '#' character should be replaced with a digit '0' to '9' to specify one of the ten tab stops.

For example:

```
.ST0=10
.ST5=20-"abc"
.ST9=t0+5
```

The first example sets tab stop zero to one inch from the left margin. The second example sets tab stop five to two inches minus the size of "abc". The third example sets tab stop nine to tab stop zero plus 1/2 inch.

TF=<character> Tab Fill character (default: ' ')

This command will change the tab fill character. The tab fill character is the character that is printed when a tab is made. The default is a space. Notice there are no quotes or anything around the character.

Some examples:

```
.TF=.     set tab fill character to '.'
.TF=-     set tab fill character to '-'
.tf=      set tab fill character to ' '
```

Header/Footer & Page Numbering Commands

XPRINT has extensive header and footer capabilities. You may define up to 10 header/footer lines with up to 3 on one line (left justified, centered, and right justified). These headers and footers have their own margins so they don't have to line up with the normal text margins. These margins can be different for even and odd pages, just as with the normal text margins.

LH{e,o}=<0-140> Left Header margin (default: 8)

This command sets the left margin for header/footer lines. If the 'e' is specified then the left header margin is set to the specified value for even pages, if the 'o' is specified then the left header margin is set to the specified value for odd pages, if 'e' nor 'o' is specified then the left header margin is set to the specified value for both even and odd pages.

Examples:

```
.LH=10
.LHE=15
```

The first example sets the left header margin to one inch for both even and odd pages. The second example sets the left header margin of even pages to 1 1/2 inches.

RH{e,o}=<0-140> Right Header margin (default: 71)

This command sets the right margin for header/footer lines. If the 'e' is specified then the right header margin is set to the specified value for even pages, if the 'o' is specified then the right header margin is set to the specified value for odd pages, if 'e' nor 'o' is specified then the right header margin is set to the specified value for both even and odd pages.

HE#="<header-string>" HEader

This command sets the header specified to the specified string. Ten header/footers can be defined HE0 - HE9.

In <header-string> the '#' character is used to insert page numbering in the header. A string of '#' characters will print the page number in a field at least as large as the size specified.

For example, if '###' is included in a header, then the page number will be printed in a field at least 3 spaces wide. If the page number is less than 3 characters, the remaining space will be filled with spaces and the page number will be printed flush right in the field. If the page number is more than 3 characters wide, the field will be expanded to fit the page number.

The <header-string> can include imbedded printer codes such as \d+, and \c. A restriction to this is you should turn off any printer features turned on. For example, if you want a header in double width characters you could have:

.he0="\d+This is a header\d-"

Notice the \d- must be there to turn off the double width. If special features are not turned off, unpredictable results will occur.

Headers cannot use the indexing (\x) or tabbing (\t) imbedded commands within them. You can not have a header defined as: HE1="Header\t20 tabbed 2 inches" or HE1="Heading number 1\xheading 1\"

If proportional characters are defined (with the P= command, see printer code commands), then headers and footers will be printed in proportional characters. Otherwise, headers and footers will be printed in normal characters (\n). The character size can be changed though using the imbedded commands in the header (\n, \a, \c, \p).

HL#=<0-255> Header Line (default: 0 for all headers 0-9)

This command tells XPRINT what line number the specified header is to be printed on. For a header/footer to be printed the line number should be less than the top margin or greater than the bottom margin. If the line specified is between the top and bottom margins, no error will occur, but the corresponding header will never be printed.

More than one header may be on the same line, (up to 3 per line). To have multiple headers appear on the same line they must have a different header format (see HF). If two headers are on the same line with different header formats and they overlap, only the first header will be printed.

Examples:

```
.HL5=0
.HL9=64
```

The first example sets it so header number five will be printed on line number zero. The second example defines header number 9 to be printed on line number 64

HF#{e,o}=(l,c,r,n) Header Format (default: n for all headers)

The header specified by the '#' (0-9) will be printed in the indicated format. The formats are as follows:

L - Print header left justified.
C - Print header centered on the line.
R - Print header right justified.
N - No print. Do not print header.

PN=<0-32767> Page Number

This command sets the page number to the specified value. Initially, when XPRINT first starts the page number is set to one. For each page that is printed, the page number is automatically incremented.

NT=(d,u,l) Number Type (default: d)

This command sets the page number type. Three types are available:

D - Decimal
L - Lower-case roman numerals
U - Upper-case roman numerals

If roman numerals are used (U,L), the limit to the page number that can be used is 5000, otherwise page numbers can go to 32767.

Examples:

```
.NT=L
.NT=d
```

The first example set the page number type to lower case roman numerals. The second example sets the page number type to decimal.

Printer Code Commands

Printer code commands are the commands related to defining the codes needed for all of the special printer features for the particular printer being used. These commands are mainly used in the initialization file, and once set usually don't need to be changed.

Many of these commands are assigned a <printer-code>. A <printer-code> is the values that are sent to the printer to turn features on or off. It is defined as a series of byte values and ASCII strings, with each value or string separated by a semicolon ';'. The byte values can be in decimal, or it can be specified as hexadecimal by prefixing the number with a dollar sign '$'. An ASCII string is a series of characters enclosed within quotes '"'. These characters are converted to their corresponding ASCII values.

Some examples:

```
1;2;3     The numbers 1 2 3 make up the printer code
"ABC"     The characters A B C make up the printer code
```

"ABC" is equivalent to 65;66;67 and $41;$42;$43 and "A";66;$43

If your printer does not have the capability for any of the following features, the corresponding command should set the printer code to null. This can be done by using a pair of quotes with nothing in them (""), or by following the equal sign of the command immediately by a comma or a carriage return. For example, to set the double width on command to null you could use one of the following:

```
.D+=""
.D+=,
.D+=
```

N=<printer-code> Normal characters

This command sets the printer for normal characters. Normal characters is usually the character set of the printer that has 10 characters per inch, but you can set the printer code to any character set your printer has.

Example:

```
.N=27;15
```

Will set the printer code for normal characters to 27;15.

SN=<1-255> Size of Normal characters

This command sets the size of normal characters. It is set by setting SN to the number of characters that fit on an 8 inch line. For normal characters this is typically 80, 10 characters per inch * 8 inches = 80.

A=<printer-code> Alternate characters

This command sets the printer code for the alternate character font. This is usually the elite characters (12 characters/inch) if your printer has it. For example, if the codes 27,40 sent to your printer turn on the 12 characters/inch font then you would use the command:

```
.A=27;40
```

SA=<1-255> Size of Alternate characters

This command sets the size of the alternate character set. It is set by setting SA to the number of characters that fit on an 8 inch line. If you are using alternate characters as elite characters (12 characters/inch) then you would set SA to 96 (12 characters/inch * 8 inches = 96).

C=<printer-code> Condensed characters

This command sets the printer code for the condensed character set.

SC=<1-255> Size of Condensed characters

This command sets the size of the condensed character set. It is set by setting SC to the number of characters that fit on an 8 inch line. For example, if your printer can print 132 characters per 8 inch line when in condensed mode, then you would have the command SC=132.

S=<printer-code> Small space

This command sets the printer code for a small dot space. This code is mainly used for proportional spacing, if you are using proportional spacing this code must be defined. If you are not using proportional spacing and your printer does not support carriage movement by a dot space, you should set the small space command to null by either of the following:

```
.S=""
.S=,
```

If your printer does not have proportional characters but it does have a small space, you should define it. If it is defined centered lines will be better centered and margins will be better aligned if you are mixing different character sizes.

SS=<1-255> Size of Small space

This command sets the size of the small space by setting SS to the number of small spaces that would fit on an 8 inch line. For example, if the small space that your printer supports and you defined using the S= command is 1/60th of an inch, then you would set SS to 480 (60 spaces/inch * 8 inches) using the command SS=480.

PI=<printer-code> Printer Init

This command will send <printer-code> directly to the printer. This is to initialize your printer if you have to send it certain codes to initialize it.

D+=<printer-code> Double width on
D-=<printer-code> Double width off

These codes are the codes that are sent to the printer to turn double width on and off.

U+=<printer-code> Underline on
U-=<printer-code> Underline off

These codes are the codes that are sent to the printer to turn the underline on

and off.

If your printer does not have any automatic underlining capability, then you should set U+ to null (N+=""). In this case XPRINT will underline by automatically backspacing after every word and print underscores "_" under each character of the word. Your printer must be able to backspace if you will be using underlining in this way. The backspace character is normally the code 8, but it can be changed with the BS command (to be explained later).

O+=<printer-code> Overstrike on
O-=<printer-code> Overstrike off

These codes are the codes that are sent to the printer to turn overstrike on and off.

If your printer does not have any automatic overstrike capability, then you should set O+ to null (O+=""). In this case XPRINT will overstrike by automatically backspacing after every word and print the word again, overstriking it. The number of times the word is overstruck can be controlled with the DA command.

DA=<1-255> DArkness of overstrike

This command sets the number of times overstriking will occur if you are using the XPRINT's overstriking. Setting DA to 1 will overstrike each word once, so each word will be printed twice. This command only has effect when XPRINT is doing the overstriking (by setting O+ to null). If you are using printer codes to tell your printer when to overstrike, this command will have no effect.

S+=<printer-code> Superscript on
S-=<printer-code> Superscript off

These commands set the codes for turning superscripts on and off. Turning superscripts on is the same as doing a reverse half line feed. Turning superscripts off is the same as doing a foward half line feed.

B+=<printer-code> suBscript on
B-=<printer-code> suBscript off

These commands set the codes for turning subscripts on and off. Turning subscripts on is the same as doing a foward half line feed. Turning subscripts off is the same as doing a reverse half line feed.

E+=<printer-code> Emphasize on
E-=<printer-code> Emphasize off

These commands set the codes for turning emphasized characters on and off.

I+=<printer-code> Italics on
I-=<printer-code> Italics off

These commands set the codes for turning italics characters on and off.

(0-9)=<printer-code> set user defined code (0-9)

This command sets up a user defined code. Up to 10 user defined codes can be defined (0-9). These codes are used to print non-ASCII codes or to "build" multi-character characters. These user defined codes are printed by using the imbedded command \(0-9), a printer escape character followed by one digit. For example, \3 in your text will print user defined code number 3.

For example, if your printer has special characters in the range 128-255, and you want to print the character with code number 194 you could define a user defined code as follows:

```
.0=194
```

Then when \0 is encountered in your text, code 194 will be sent to your printer. Or if you want to define a slashed zero you could define:

```
.5="0";8;"/"
```

This code would be printed when\5 is encountered. This code prints a "0" then a backspace, then a "/" on top of the zero.

In addition to defining the code you must also set the size of the character, see next command (S(0-9)).

S(0-9)=<byte-value> set Size of user defined code

This command sets the size of a user defined character. The size of a character must be set before it is used. This size is used so XPRINT can keep track of linelengths correctly.

The size is measured differently for mono-spaced characters and proportional characters. For mono-spaced characters the size is specified in the number of characters the user defined code prints out as. For example, for the slased zero code in the previous example, the final outputed characters take up one column of space so the size would be set to one (S5=1).

For proportional characters the size of the defined code is the number of dot widths the character takes up. For example, if you define a code as a graphics character (1=183 for example) and graphics code 183 is 8 dots wide, then you would set the size of defined code 1 as 8 (S1=8).

P=<printer-code> Proportional characters

This command sets the printer code for the proportional character font. Size of proportional characters does not have to be defined because it is defined by CT and size of small space.

BS=<printer-code> Back Space character

This command defines the backspace character. If you are in monospaced mode <printer-code> is printed once to backspace over 1 character.

If you are in proportional mode, then <printer-code> cna include the '#' character. If it does then '#' is replaced with the size of the character being backspaced when it is backspaced. If <printer-code> does not have a '#' in it then <printer-code> is printed X times, where X is the size of the character being backspaced over.

Proportional Spacing Related Commands

CT="<96-character-string>" Character size Table
CT<ascii-value>=<0-255>

This command sets up the character sizes for the ASCII characters (values 32 - 127). The first method is using a 96 character string enclosed in quotes. The 96 characters correspond to the 96 ASCII characters from space (32) to ASCII 127. Each character may be between A-Z in upper or lower case. A corresponds to 1 dot width, B=2 dot widths, C=3 dot widths, ... Z=26 dot widths. For example, for the first 5 ASCII characters (space,!"#$) if the sizes are 4,3,5,6,3 dot widths respectively, then the command would be CT="DCEFC.....". There must be exactly 96 characters between the quotes or an error will occur. Using this method to define the character size table you must be careful to make sure you get all the characters right.

The second method assigns a width to each character separately. <ascii-value> can either be numeric, or you can specify the actual character by prefixing the character with a "'". For example, 65, $41, and 'A all specify the same ASCII value. For example CT'A=5 will set the width of "A" to 5 dot widths, CT65=5 and CT$41=5 do the same. So you can define the characters singly with:

.CT' '=4, CT'!=3, CT'"=5, CT'#=6, CT'$=3

and do this for all 96 characters.

PJ=(c,s) Proportional Justification mode (default: c)

This command defines what way proportional justification is accomplished. There are 2 ways: C - character, S - Space.

Justification by character means small spaces are put into the spaces between the characters as well as in the spaces. This makes the line more even because the spaces don't get huge with the character spacing staying the same. Small spaces are only put between the characters when the line has to be expanded a large amound.

Justification by space means extra spaces are only added to the spaces. The spacing between characters stays the same.

Example

.PJ=S

will set proportional justification mode to spaces.

CS=<0-255> Character Spacing (default: 0)

This command will allow you to put extra spaces between each and every character. This allows you to put more space between the characters. For example, CS=1 will put one extra dot space between every character. CS=0 will not put any extra spaces between characters.

TT="<96-character-string> define Translation Table
TT<ascii-value>=<ascii-value>

This command lets you define a translation table. This translation table will define what character will actually be sent to the printer for each character in the text. Some daisy wheel printers need this because the characters printed are not always what is sent to the printer.

The <96-character-string> are the characters that are actually sent to the printer. The characters are in ASCII order, starting with space (32) up to ASCII code 127. For example if the first part of the command is

.TT="YE$HR....

The characters "YE$HR" correspond to the first five ASCII characters space,!,",#,$. Whenever a space is printed, a "Y" is sent to the printer. Whenever a "#" is printed, a "H" is sent to the printer.

The <96-character-string> must have exactly 96 characters or an error will occur.

The second method of defining a translation table is by setting the translated value for each character one at a time. <ascii-value> can either be numeric or it can be the ASCII character itself by prefixing it with a "'". For example TT'R='V will send a "V" to the printer whenever a "R" is to be printed.

TR=(y,n) TRanslation on/off (default: n)

This command turns the translating on and off. If TR=Y, then translation is done according to the translation table defined with TT. If TR=N, no translation is done, the translation table is not used.

PS=(0,1) Proportional Spacing (default: 0)

This command turns proportional spacing on and off. If proportional spacing is

off (PS=0), regular spaces are inserted at the spaces to justify the line.

If PS=1, proportional spacing is on, small dot spaces are inserted at the characters and/or spaces (depending on what PJ is set to) to justify the line.

3. IMBEDDED COMMANDS

While most of XPRINT's commands are given through format lines, there are some commands that must be imbedded within the text. These commands include printer commands, indexing, tabbing, and pausing in the middle of a line.

These commands are imbedded by using a character called the printer escape character. It is defined as the back-slash character "\" but may be changed with the PE format command. In this manual all examples assume the "\" character is the printer escape character.

The printer escape character followed by one or more additional characters make up an imbedded command. The characters following the printer escape character can be in either upper or lower case, \E+ is the same as \e+. If you want to use the printer escape character just as an ordinary character, putting two of them together will output as one of them, and will not be interpreted as a printer escape character.

For example, the text

The back-slash "\\" is the printer escape character.

will output as

The back-slash "\" is the printer escape character.

Printer Escape Codes

Most of the imbedded commands using the printer escape character are used for commands for the printer. There are 4 types of these printer codes: changing fonts (character sizes), changing styles (underline, double width, etc.), user defined printer codes (\0 - \9), and imbedded codes to be sent to the printer (Z-codes).

FONTS

\N Normal characters
\C Condensed characters
\A Alternate characters
\P Proportional characters

The four imbedded commands change the character size in the middle of a line. Some printers are not able to change font type in the middle of a line. If this is the case with your printer, you will have to use the FO command in a format line to change fonts.

The imbedded commands \N, \C, \A, \P is the same as the FO format command except the imbedded commands do not change the PS (proportional spacing) command, while the FO command changes the PS command to 0 for N, C, and A (to turn off proportional spacing), and to 1 for P (to turn on proportional spacing).

STYLES

\U+ \U- Underline on/off (no spaces underlined)

These imbedded commands turn underlining on and off. The underline printer commands must have been defined with the format line commands U+= and U-=. With this command no spaces will be underlined. If you want spaces underlined use the \V+ and \V- commands.

Example:

> Here is a test of \U+the underlining capability of XPRINT. Notice how spaces are not underlined and the margins are not underlined. Underlining is turned off \u- with u-.

when formatted will output as:

> Here is a test of the underlining capability of XPRINT. Notice how spaces are underlined and the margins are not underlined. Underlining is turned off with u-.

\V+ \V- Underline on/off (spaces underlined)

These imbedded commands turn underlining on and off. The underline printer commands must have been defined with the format commands U+= and U-= (V does not have its own definition). With this command spaces will be underlined on the condition that the printer does have underlining capability. Spaces will not be underlined if underlining is done by backspacing over words and printing underscores (see page 23).

Example:

> Here is a test of \v+the underlining capability of XPRINT. Notice how spaces are underlined and the margins are not underlined. Underlining is turned off \V- with V-.

when formatted will output as:

> Here is a test of the underlining capability of XPRINT. Notice how spaces are underlined and the margins are not underlined. Underlining is turned off with V-.

\E+ \E- Emphasize on/off

These commands turn emphasized characters on and off. Your printer must

have emphasizing capability and the emphasize printer codes must have been defined with the E+= and E-= format commands.

Example:

To create \e+emphasized \e- characters use the e+ and e- imbedded commands.

when formatted will output as:

To create **emphasized** characters use the e+ and e- imbedded commands.

\I+ \I- Italics on/off

These commands turn italics characters on and off. Your printer must have italics capability and the italics printer codes must have been defined with the I+= and I-= format commands.

\D+ \D- Double width characters on/off

These commands turn double width characters on and off. Your printer must have double width capability and the double width printer codes must have been defined with the D+= and D-= format commands.

Example:

You can create \d+BIG \d- characters using the d+ and d- imbedded commands.

when formatted will print the word "BIG" in double width characters.

\O+ \O- Overstrike on/off

These commands turn overstriking on and off. The overstrike printer codes must have been defined with the O+= and O-= format commands. These imbedded commands may work even if your printer does not automatically do overstriking, by setting O+="" in the format command (see page 34).

\S+ \S- Superscript on/off

These imbedded commands turn superscripts on and off. The superscript printer codes must have been defined with the S+= and S-= format commands.

Example:

Superscripts can be used for formulas such as E=MC\S+2\S-.

will output as:

Superscripts can be used for formulas such as $E=MC^{2}$.

\B+ \B- Subscript on/off

These imbedded commands turn subscripts on and off. The subscript printer codes must have been defined with the B+= and B-= format commands.

Example

H\B+2\B-O is the formula for water.

will output as:

$H_{2}O$ is the formula for water.

USER DEFINED CODES

\0 \1 \2 \3 \4 User defined codes
\5 \6 \7 \8 \9

These imbedded codes are codes that were previously defined with the 0=, 1=,... 9= format commands (see page 25). The size of these codes also must be defined with the S1=, S2=,... S9= format commands. These imbedded commands can be used to print characters outside of the ASCII range (32-127) or to print muliple character symbols, such as the slashed zero example given on page 25.

IMBEDDED Z-CODES

\Z<printer-code> Z-code

This imbedded code allows you to imbed printer codes inside your text. This command is for inserting commands to your printer that XPRINT does not support. <printer-code> is the same as the <printer-code> as defined in the printer code commands section, see page 21.

The <printer-code> must be ended with a printer escape character. Defining a very long Z-code may cause a "Line buffer overflow" error to occur. If this happens you must shorten the Z-code (by using user defined printer codes for example). If the final printer escape character ending the Z-code is left out, a "Line buffer overflow" error will occur because the Z-code is never ended.

Example:

Use the Z-code imbedded command to send
\Z15;13;"AB"\ codes to the printer.

will send the values 15,13,"A","B" to the printer just before the word "codes".

Indexing

\X<indexed word> Put <indexed word> into index

This imbedded command will put <indexed word> into the index. <indexed word> may include any characters except for the printer escape character and carriage returns. It can be up to 100 characters long.

When this imbedded command is executed XPRINT will put <indexed word> and the current page number into a file called <filename>.X where <filename> is the name of the file being formatted.

After XPRINT is done formatting, it automatically executes the program called INDEX. INDEX takes the file <filename>.X, reads in all the data, sorts the words and page numbers, and outputs another file with <filename>.NDX as its name. This file is a text file with XPRINT commands in it that can be edited or printed with XPRINT just as any standard text file.

After INDEX is done and you have the file <filename>.NDX you can either print it by using the command "XP <filename>.NDX" or you can edit it using XED to make any changes in the format you want. INDEX will make the file with three inch lines, this way you can put two columns per page by pasting the second column in. This is how the index was done for this manual.

After XPRINT is done and INDEX is done and you have printed the <filename>.NDX file, you will have the two extra files <filename>.X and <filename>.NDX in your data directory. You can delete these files if you wish.

You can merge <filename>.X index data files together from different XPRINT runs and execute INDEX on the large file to create an index from more than one file. For example, if you XPRINT a file called FILE1, then XPRINT a file called FILE2, and then XPRINT a file called FILE3, where each of these files are printed to different pages, you will end up with FILE1.X, FILE2.X, FILE3.X as the index data files. In addition you will have the files FILE1.NDX, FILE2.NDX, FILE3.NDX from the output of INDEX. These last three files may be deleted since you only want one index covering all three files. The three index data files can then be merged into one file with the command:

```
MERGE >BIGFILE.X FILE1.X FILE2.X FILE3.X
```

You can then manually execute INDEX using the file BIGFILE.X to create an index with the entries from all three files with the command:

```
INDEX BIGFILE.X
```

The file specified in the command must end in a .X suffix. The output from this command will be a file called BIGFILE.NDX which is an index of all the index entries from all three files which can be edited or XPRINTed.

Tabbing

\T<tab/indent value> Tab over to <tab/indent value>

This imbedded command will allow you to tab over to a specified spot on the line. <tab/indent value> is as defined in the indenting & tabbing commands section on page .

This command allows you to create columns in your text, even if you are using proportional characters. The <tab/indent value> is relative to the left margin, unless it is preceded by a plus sign '+'. If a plus sign is preceding it then it is relative to the current cursor position which enables you to make lines of a specified size.

Example:

```
.tf=_
Name \t+40\
Address \t+50\
```

will output as:

```
Name____________________________________________
Address_________________________________________________________
```

The t+40 tab command will tab four inches from the current cursor position. The .tf command makes the tab fill character an underscore character '_' so four inches of underscores are printed.

Example:

```
col 1\t10\col 2\t20\col 3\t30\col 4
```

will output as:

```
col 1        col 2        col 3        col 4
```

The "\t10\" tabs in one inch from the left margin, "\t20\" tabs in two inches from the left margin, and so on.

Example:

```
.tf=.
.st0=50
Chapter 1\tt0\1
 Section 1\tt0\2
 Section 2\tt0\3
Chapter 2\tt0\4
Chapter 3\tt0-"1"\10
```

will output as:

This example shows how powerful the tabbing feature can be. Notice that the page numbers at the right are perfectly alligned even though proportional characters are used. Notice the line for "Chapter 3", it tabs out to position T0-"1", the -"1" is to make room for another digit so the two digit number 10 will be alligned with the other one digit numbers. It could also have been done by defining tab stop 1 as ST1=T0-"1" and then using \tt1\ to tab out.

Pausing in the Middle of a Line

\w"<string>" Display <string> and pause (wait)

This command allows you to stop printing in the middle of a line, display a message to the screen, and wait for **<ENTER>** to be pressed before continuing on. This command is mainly for daisy-wheel type printers so you can change print wheels in the middle of a line. This command will not work properly if you are using a dot matrix line printer.

Example:

Change wheels to \w"Place bold wheel in printer, press <enter>"Emphasize characters.

when formatted will print the characters "Change wheels to ", and then printing will stop and the message "Place bold wheel in printer, press <enter>" will be displayed on the screen. Once **<ENTER>** is pressed printing will continue.

4. THE INITIALIZATION FILE

In order for XPRINT to work you must have an initialization file for your printer in your execution directory. The initializaion file sets up the codes for your printer for the various printer functions, such as character sizes and underlining and so forth. Also in the initialization file can be format commands to set your own defaults.

The initialization file is a text file that you can easily write yourself, or you can modify an existing one.

Many initialization files are included with XPRINT. They are in the directory called INITFILES on the XPRINT diskette. Read the READ.ME file in that directory to find out what printers the various files are for. If one of the included initialization files is for your printer you can just copy that file to your execution directory and use it. Otherwise you will have to modify one of the other files or write one yourself.

When writing an initialization file, or modifying one, there are a few things you must keep in mind. They are the following:

1. Unless otherwise specified with the -I option in the command line, XPRINT will look for an initialization file called INIT.XP in the execution directory. So the initialization file you use the most should be named INIT.XP.

2. The initialization files attributes must have the execution bit set for XPRINT to read it. Editors, including XED, will not set the execution bit for text files, so you must set it yourself by executing the command:

   ```
   ATTR /D0/CMDS/INIT.XP E
   ```

3. The initialization file can only contain valid XPRINT format lines. This means each line must start with a period '.'.

What is Needed

The following items should be included in an initialization file:

1. Printer codes and sizes for the the different character fonts: normal, alternate, condensed, and proportional characters (N=, A=, C=, SN=, SA=, SC=, P=). Printer code for small space and size (S=, SS=).
   If your printer does not have alternate, or condensed characters, you should define them the same as the normal characters. If your printer does not have proportional characters, you should set the proportional code to null (P="").

2. Printer codes for styles: double width, underline, superscript, subscript, overstrike, emphasized, and italics. Both the feature on, and feature off must be defined. If your printer does not have one or more of these features, set the associated printer codes to null.

3. Other codes that need to be defined for your printer, such as backspace.

4. Character size table if the printer supports proportional characters. Character translation table if needed.

5. Default format commands. Format commands that will take effect when a file is formatted. Useful for setting default margins, headers, etc.

Sample Initialization File

Here is a sample initialization file. It is for a Gemini-10X printer or any other printer that uses the same codes.

```
.* init file for gemini-10x
.n=$12,sn=80,  * normal characters 10 char/inch, 80/8inch line
.a=$1b;$42;$2,sa=96, * alternate characters, 12 char/inch
.c=$f,sc=136,  * condensed characters, 136/line
.p="",  * no proportional characters
.d+=$e,d-=$14, * double width on/off
.u+=$1b;$2d;1, u-=$1b;$2d;0, * underline on/off
.s+=$1b;$53;0, s-=$1b;$54, * superscript on/off
.b+=$1b;$53;1, b-=$1b;$54, * subscript on/off
.o+=$1b;$47, o-=$1b;$48, * overstrike on/off
.e+=$1b;$45, e-=$1b;$46, * emphasize on/off
.i+=$1b;$34, i-=$1b;$35, * italics on/off
.s=$1b;$4b;1;0;0, ss=480, * small space, 480 per line
.bs=8, * backspace
.
.* Put default settings here, can be any format commands.
.* Especially useful for setting default margins, headers,
.* and so forth.
```

FORMAT COMMANDS

Command		Description	Page References
LF e,o	= l,c,r,j,a	Line Format	10
LM e,o	= 0-255	Left Margin	08
TM e,o	= 0-255	Top Margin	09
BM e,o	= 0-255	Bottom Margin	09
RM e,o	= 0-255	Right Margin	09
IN	= <tab/indent value>	INdent	15
RI	= <tab/indent value>	Right Indent	17
DI		Delayed Indent (reset)	15
DI	= <tab/indent value>	Delayed Indent (set)	15
PN	= 0-32767	Page Number	20
PE	= <character>	Printer Escape character	
HS	= <character>	Hard-Space character	13
PL	= 3-255	Page Length	09
PG		start new PaGe	10
OP		start new Odd Page	11
EP		start new Even Page	11
PP	= y,n	Page Pause	11
LS	= 0-255	Line Spacing	10
LH e,o	= 0-255	Left Header margin	18
RH e,o	= 0-255	Right Header margin	18
HE#	= "<string>"	HEader	18
HL#	= 0-255	Header Line	19
HF# e,o	= l,c,r,n	Header Format	20
SN	= 0-5000	Size of Normal character (# of char/8 in. line)	22
SC	= 0-5000	Size of Condensed characters	22
SA	= 0-5000	Size of Alternate characters	22
SS	= 0-5000	Size of Small space	23
FI	= "<filename>"	File Include	12
NT	= d,l,u	Number Type	20
CT	= "<96 character string>"	Character size Table	27
CT	<ascii value>= 0-255		
PJ	= c,s	Proportional Justification mode	27
CS	= 0-255	Character Spacing	28
NL	= 0-255	Need Lines	12
SP	= 0-255	blank SPace	12
PI	= <printer code>	Printer Init	23
FO	= n,c,x,p	FOnt type	14
BS	= <printer code>	BackSpace character	26
DA	= 1-255	DArkness of overprint	24
TT	= "<96 char string>"	Translation Table	28
TR	= y,n	TRanslate	28
PS	= 0,1,2	Proportional Spacing	28
FS	= <printer code>	Forward Space	29
TF	= <character>	Tab Fill character	17
LA	= y,n	Line Adjust	14
ST#	= <tab/indent value>	Set Tab stop	17
D+,D-	= <printer code>	Double width on/off	23
U+,U-	= <printer code>	Underline on/off	23
S+,S-	= <printer code>	Superscript on/off	24
B+,B-	= <printer code>	suBscript on/off	25
O+,O-	= <printer code>	Overstrike on/off	24
E+,E-	= <printer code>	Emphasized on/off	25
I+,I-	= <printer code>	Italics on/off	25
N	= <printer code>	Normal characters	21
C	= <printer code>	Condensed characters	22
A	= <printer code>	Alternate characters	22
P	= <printer code>	Proportional characters	26
S	= <printer code>	Small space	23
0-9	= <printer code>	programmed character 0-9	25
S0-S9	= 0-255	Size of programmed character 0-9	26

INDEX