PolyText

USERS GUIDE

PolyData
microcenter

TABLE OF CONTENTS

1. Introduction to PolyText 2

2. Operating PolyText 3
   2.1 PolyText functions 3
      2.1.1 Load text 3
      2.1.2 Save text 4
      2.1.3 Edit text 4
      2.1.4 Clear textbuffer 4
      2.1.5 Format text 4
      2.1.6 New page 6
      2.1.7 Output parameters 6
      2.1.8 Quit 7

3. The PolyText formatter 8
   3.1 Formatter commands 8
      3.1.1 Formatting mode commands 9
      3.1.2 Underline commands 11
      3.1.3 Margin control commands 11
      3.1.4 Line and page control commands 12
      3.1.5 Header and footer definition commands 12
      3.1.6 Text register commands 13
      3.1.7 Various commands 14

Section 1

Introduction to PolyText

PolyText is an advanced text processing system for use on a NASCOM 1 or 2 running under PolyDos. PolyText gives you the ability to create letters, manuals, articles, etc. using the editor, and to have them printed using the formatter.

Through commands embedded in your manuscript you can switch between five different formatting modes, including even margining and text centering, define headers and footers, for all pages as well as even/odd pages, change both left and right margins, and turn on and off underlining. Page numbering is supported through a page number register which can be inserted anywhere in your text or in headers and footers, and text register commands provide you with a method of varying selected parts of your document, for instance a letterhead, without changing the manuscript itself. Since formatting is done while your text is being printed you can easily experiment with different line and page lengths without altering the manuscript. Furthermore you can select seperate pages only to be output. Printing can be done either to a printer or to the screen.

To ensure readability on both versions of PolyDos (G805 and G809/G815), PolyText comes on a single density disk. The program itself is contained in a machine code file called PTXT.GO. To run PolyText you will have to copy this file onto your system disk using a COPY command (if you a running on a single drive system use COPY in connection with an 'S' option).

This manual was created using PolyText, and printed on a NEC 3515 spinwriter.

Section 2

Operating PolyText

Before you invoke PolyText make sure that your system disk contains the PolyEdit overlay (Edit.OV) as PolyEdit is used by PolyText. To invoke PolyText enter the command line:

$PTXT

On running PolyText the screen is cleared and the main menu is displayed:

```
PolyText 2.0    Main menu

L. Load text
S. Save text
E. Edit text
C. Clear textbuffer
F. Format text
N. New page
O. Output parameters
Q. Quit

Which function?

aaaaa bytes in use, bbbbb free.
```

To invoke a function enter the character to the left of the function name. Input lines are always ended by <ENTER>. To correct mistakes you may use the <BACK> key which backspaces one character position.

2.1 PolyText functions

In the following sections the functions offered by the main menu is described in detail. Whenever a function completes its task you are returned to the main menu. When (or if) an error occurs, for example an attempt to load an unexisting file, an error message is displayed:

>>>> I can't find that file. Press <SPACE>

On pressing <SPACE> you will be returned to the main menu.

2.1.1 Load text

The load text function is used to load text files into the text buffer. On invoking this function you are prompted:

Load from which drive?

to which you must answer by entering a valid drive number (invalid numbers will return you directly to the main menu). Following this, PolyText reads into memory the directory of that drive and displays the names of all text files (files with the

.TX extension) on that disk. The list is followed by the prompt:

Load which file?

to which you must answer by entering the file specifier of the file to be loaded. The default extension is .TX, and the default drive is the drive you have selected above. When a file is loaded, it is merged to the end of the text already in the text buffer. If loading the file would overflow the text buffer PolyText will display:

>>>> There is not enough memory. Press <SPACE>

The file will not be loaded, but the text already in the text buffer is unchanged.

2.1.2 Save text

The save text function is used to save the text currently within the text buffer into a text file. On selecting the save text function the screen is cleared and you are prompted:

Save text as?

to which you must answer by entering a file specifier. The default extension is .TX, and the default drive is the master drive.

2.1.3 Edit text

The edit text will enter the editor. On selecting this function PolyText will invoke the PolyEdit overlay. For a description of the features offered by the editor, please refer to the PolyEdit Users Guide. When you press CTRL/^ from the editor you are returned to the main menu in PolyText.

2.1.4 Clear textbuffer

This function is used to erase the text currently held within the text buffer. On selecting the clear textbuffer function you will be prompted:

Are you sure?

If you answer 'Y' the text is erased. Anything else will return to the main menu with no changes made to the text.

2.1.5 Format text

The format text function will invoke the PolyText formatter. This section only describes the format text command. For a full desctiption of the text formatter and its commands, please refer to section 3. On invoking the format text command, you will be prompted:

Output device (P/V)?

to which must must answer 'P' or 'V'. 'P' instructs the formatter to direct its output to the printer, i.e. the printer output routine contained in the information file area. 'V' will output to the screen. Following this you are asked:

Format all text (Y/N)?

If you answer 'Y' all of your text will be printed, whereas 'N' allows you to select a section of text to be printed. Selection is done on page basis, i.e. one or more pages of text can be selected. If you answerred 'N' to the above question you will be prompted:

First page number?

to which you must answer by entering the number of the first page you want printed. Following this comes yet another question:

Last page number?

to which you must answer by entering the number of the last page you want printed. Note that when you select only a part of your text to be printed, the formatter still processes the rest, since this is the only way of determining where pages begin and end. However, pages before the first page or beyond the last page are not sent to the output device. The last question to be answerred before the formatter goes to work is:

Input file name (CR for KBD)?

The inputs we are talking about here are the possible inputs required by the document to be formatted, i.e. inputs to one or more of the text registers. Usually, the above question is answerred by pressing <ENTER> directly, thus instructing the formatter to obtain its inputs from the keyboard. However, if you enter a file specifier the file will become the input device instead of the keyboard. The above question may be seen upon as a way of doing a local activation of the command file mode which is deactivated as soon as you return to the main menu. The aspects of this is described in more detail in section 3.1.6. When the above question has been answerred the formatter starts processing. If the screen was selected for output, the formatter will stop and blink the cursor each time 14 lines has been printed. Press any key to continue. At any time you may press ESC which causes the formatter to stop and prompt:

>>>> Operator abort. Press <SPACE>

On pressing <SPACE> you will be returned to the main menu. If an error is detected by the formatter, i.e. an invalid formatting command, the formatter goes:

>>>> Format error. Press <SPACE>

On pressing <SPACE> the editor is invoked, and the cursor is moved to the command in error. When the formatter reaches the end of your document it prompts:

Print again?

to which you must answer 'Y' or 'N'. 'Y' will repeat the printout (however it will not redo the leading questions), whereas 'N' will return you to the main menu.

2.1.6 New page

The new page function will send a form-feed to the printer, causing it to move to the top of a new page.

2.1.7 Output parameters

The output parameters function allows you to, as the name suggests, change the output parameters used by the formatter. On selecting this function the screen is cleared and a submenu is displayed:

```
PolyText 2.0    Output parameters

F. Form size? 72
B. Bottom margin? 8
C. Characters per line? 72
O. Offset from left edge? 8
N. First page number? 1
P. Pause after each page? N

Edit which field?
```

Note that the above parameters values are not necessarily the ones that appear on your screen. The F, B, C, and O fields defines the forms used in your printer. The N field defines the starting value of the page counter register of the formatter. The P field is a boolean variable (either 'Y' (yes) or 'N' (no)). If P is set to 'Y', the formatter will stop its output flow each time a page has been written, and ask you to press <SPACE> before it continues, thus enabling you to print on single sheets. If fan-fold paper or paper rolls are used the P field should yield 'N', as no pauses are required here.

To edit a field, enter the first character of the field name (i.e. the character before the period). Assuming a correct field selector is entered, the cursor moves to the question in that field, allowing you to enter a new value. Illegal values are ignored by deleting the entry, thus prompting for a new value. Once a valid value has been typed, the cursor moves back to the 'Edit which field' question. To terminate the output parameters function enter an empty line.

When PolyText is invoked some default values are put into the output parameters. The values stored in F, B, C, and O are fetched from the corresponding fields in the information file area (addresses 0C210H-0C213H, symbolically known as PLPP, PBMG, PCPL, and PLMG. Please refer to the System Programmers Guide for further informations on this subject). The N field defaults to 1 and the P field defaults to 'N'.

2.1.8 Quit

The quit function is used to terminate PolyText. On selecting the quit function the screen is cleared and you are returned to the command level in PolyDos. Following a quit function call you can reenter PolyText using the command line:

```
$PTXT;W
```

However, this requires that no vital memory areas has been overwritten.

Section 3

The PolyText formatter

The formatter is the part of PolyText that formats your text, that is, it performs various textual functions to produce the finished document. The process of formatting a text does in no way affect the actual text in memory. Formatting is done while the text is being printed.

The input text appears to the formatter as a number of paragraphs, each paragraph separated from the others by a carriage return. The formatter will apply textual rearrangements to the text within a paragraph only and not to the paragraph delimiters, i.e. the carriage returns. Thus, whenever a carriage return appears in the text it will also appear in the finished document. The manuscript presented to the formatter should not contain graphic characters (if it does, such characters are printed as the corresponding ASCII character but with underlining).

On processing a paragraph, the formatter will subdivide the paragraph into a number of lines. To extract a line, the formatter repeatedly reads words into a line buffer until the buffer is full (which happens when the next word is longer than the number of characters left in the buffer), or until a carriage return is met. Once extracted, the line is formatted according to the current formatting mode, and finally the finished line is sent to the output device. A word is any number of contiguous non-blank characters, and each word is separated from other words by at least one blank (space) or TAB (tabulate character). When a line is read, the separators between words are included as well as the words. Words are never broken except in that seldom case where a word is longer than the current line width. To arrive at the first character in the next line, the formatter keeps on skipping blanks (or TABs) until a non-blank character is met.

The calculation of the line with involves several parameters, global as well as local. Global parameters are the parameters defined by the output parameters function from the main menu (see section 2.1.7). Local parameters are the current left and right margins, defined by commands embedded in the text. To arrive at the exact line width, the formatter subtracts from the global line with (i.e. the 'Characters per line' field less the 'Offset from left edge' field) the current left and right margins.

3.1 Formatter commands

Commands to the formatter consists of command words enclosed in angular brackets (<>). Some commands require or permit a parameter value to be specified, in which case the value must follow the command word. Several commands can be specified within the brackets if separated by commas. Examples of formatter commands:

<bp> <ce,ul> <lm4> <fi,nu,lm6,rm10>

The formatter does not care whether upper or lower case letters are used in command words. To include the opening angular bracket in the text, type it twice for each occurrance desired, thus to produce '<<' type '<<<<'. A special command to the formatter is the use of the underline character (ASCII 5FH). The underline character is looked upon by the formatter as any other non-blank character, i.e. two words seperated by a an underline will appear to the formatter as one word. However, when an underline character is encountered by the output routine, a blank (space) character is substituted.

The formatter commands are divided into 7 groups:

Formatting mode commands
Underline commands
Margin control commands
Line and page control commands
Header and footer definition commands
Text register commands
Various commands

Each of these groups are described in the following sections.

3.1.1 Formatting mode commands

The formatting mode commands are used to define which type of formatting to apply to subsequent lines. If the command appears in front of the first character of a line, that line (and of course the following lines) are affected. Otherwise, the current line will be formatted according to the current formatting mode, and the new formatting mode will start at the next line.

<le> Left justify. In this mode no formatting is done to the lines extracted, since they are already adjusted to the left margin.

<ri> Right justify. Enough blanks are inserted in front of the line to make the last character correspond to the right margin.

<ce> Center justify. The extracted line is output at the center of the print line.

<fi> Line fill. In this mode the extracted line will be adjusted to the right margin by the insertion extra blanks where blanks already appear. However, if the current line is the last line of a paragraph, no filling is done, corresponding to a left justification. Line fill is the default formatting mode, that is the mode selected on entering the formatter.

<ex> Line expand. The line expand mode is equivalent to the line fill mode, except that the last line of a paragraph is also filled.

To demonstrate the above commands some examples follow, showing what each command does to the same piece of text. The text used

in each example is listed below, just as it would appear on the screen:

```
PolyDos is a high-level disk operating system de
signed specially for NASCOM 1 and 2 with NAS-SYS
 1 or NAS-SYS 3 monitor. The basic concept of Po
lyDos is that it is totally compatible with exis
ting software written for NAS-SYS and NASCOM ROM
 BASIC. The PolyDos package includes the PolyDos
 controller ROM, the PolyDos system files, the P
olyEdit on-screen editor, the PolyZap disk assem
bler, the PolyDos DISK BASIC extension to the NA
SCOM ROM BASIC, and a number of utility programs
 for formatting, editing, and copying disks.
```

Note that the text contains no carriage-returns, thus being a paragraph as explained earlier.

LEFT JUSTIFICATION (<le>):

```
PolyDos is a high-level disk operating system designed
specially for NASCOM 1 and 2 with NAS-SYS 1 or NAS-SYS 3
monitor. The basic concept of PolyDos is that it is
totally compatible with existing software written for
NAS-SYS and NASCOM ROM BASIC. The PolyDos package
includes the PolyDos controller ROM, the PolyDos system
files, the PolyEdit on-screen editor, the PolyZap disk
assembler, the PolyDos DISK BASIC extension to the
NASCOM ROM BASIC, and a number of utility programs for
formatting, editing, and copying disks.
```

RIGHT JUSTIFICATION (<ri>):

```
PolyDos is a high-level disk operating system designed
specially for NASCOM 1 and 2 with NAS-SYS 1 or NAS-SYS 3
monitor. The basic concept of PolyDos is that it is
totally compatible with existing software written for
NAS-SYS and NASCOM ROM BASIC. The PolyDos package
includes the PolyDos controller ROM, the PolyDos system
files, the PolyEdit on-screen editor, the PolyZap disk
assembler, the PolyDos DISK BASIC extension to the
NASCOM ROM BASIC, and a number of utility programs for
formatting, editing, and copying disks.
```

CENTER JUSTIFICATION (<ce>):

```
PolyDos is a high-level disk operating system designed
specially for NASCOM 1 and 2 with NAS-SYS 1 or NAS-SYS 3
monitor. The basic concept of PolyDos is that it is
totally compatible with existing software written for
NAS-SYS and NASCOM ROM BASIC. The PolyDos package
includes the PolyDos controller ROM, the PolyDos system
files, the PolyEdit on-screen editor, the PolyZap disk
assembler, the PolyDos DISK BASIC extension to the
NASCOM ROM BASIC, and a number of utility programs for
formatting, editing, and copying disks.
```

LINE FILL (<fi>):

PolyDos is a high-level disk operating system designed specially for NASCOM 1 and 2 with NAS-SYS 1 or NAS-SYS 3 monitor. The basic concept of PolyDos is that it is totally compatible with existing software written for NAS-SYS and NASCOM ROM BASIC. The PolyDos package includes the PolyDos controller ROM, the PolyDos system files, the PolyEdit on-screen editor, the PolyZap disk assembler, the PolyDos DISK BASIC extension to the NASCOM ROM BASIC, and a number of utility programs for formatting, editing, and copying disks.

LINE EXPAND (<ex>):

PolyDos is a high-level disk operating system designed specially for NASCOM 1 and 2 with NAS-SYS 1 or NAS-SYS 3 monitor. The basic concept of PolyDos is that it is totally compatible with existing software written for NAS-SYS and NASCOM ROM BASIC. The PolyDos package includes the PolyDos controller ROM, the PolyDos system files, the PolyEdit on-screen editor, the PolyZap disk assembler, the PolyDos DISK BASIC extension to the NASCOM ROM BASIC, and a number of utility programs for formatting, editing, and copying disks.

3.1.2 Underline commands

Through the underline commands you can select parts of your text to be underlined. If a line contains text to be underlined the formatter will, when the line itself has been printed, output a carriage-return, thus moving the print head to the beginning of the line, followed by a line of blanks and underline characters (ASCII value 5FH).

 Start underlining. Subsequent characters will be underlined when output.

<nu> End underlining. Restores normal mode of operation.

Since the underline character is printed as a blank, you must use the underline commands to produce underline characters in your document. To produce a single underline character in the output text use:

_<nu>

3.1.3 Margin control commands

The margin control commands are used to control the margins. Note that the margin control commands does not affect the global line width parameters (the parameters set using the output parameters function from the main menu). Also note that it is not possible to specify negative margins. If the command appears in front of the first character of a line, that line (and of course the following lines) are affected. Otherwise, the command will take effect next time the formatter begins a new line.

<lmX> Set left margin. X is a decimal number defining the number of spaces to indent from the left margin. If X is omitted, zero is assumed. Thus, you may use a <lm> command to cancel a left margin.

<rmX> Set right margin. This command is equivalent to the above, except that it affects the right margin, i.e. defines the number of spaces to indent from the right margin.

<dmX> Set double margin. A <dmX> command corresponds to <lmX,rmX>, thus defining the number of spaces to indent from both margins. If X is omitted, both margins are reset.

If you try to define margins that would result in a line width smaller than 24 characters a format error occurs.

3.1.4 Line and page control commands

The commands in this group are used to control skipping of lines and pages.

<slX> Skip lines. If X is specified the command evaluates into X CR/LF sequences. If X is omitted, a single carriage-return is output, thus retuning the print head to the first character column on the current line.

<bpX> Begin page. If X is specified, the formatter will begin a new page if there are less than X lines remaining on the current page. This is useful for assuring that a table or a diagram will fit on a page. If X is not specified, the formatter will begin a new page if it is not already at the top of a page.

<np> New page. This command will move to the top of a new page even if the formatter is already at the top of a page.

The PolyText manual itself gives a good example of these commands in use. Throughout the manual a <bp5> command is issued before the heading of each section to assure that the heading is not placed at the bottom of a page with the section itself starting at the next page. Furthermore, a <sl,lm8> command is used to indent the description of each command. For instance, the description of <np> above looks like this on the screen:

<<np><sl,lm8>New page. This command will move to
the top of a new page even if the formatter is
already at the top of a page.<lm>

3.1.5 Header and footer definition commands

These commands are used to define headers and footers, i.e. texts that are to be printed on the top and at the bottom of each page. Commands are provided to define headers/footers for all pages, odd pages, and even pages (a page is odd if its page number is odd, and even if its page number is even). Headers for

all pages are printed before headers for odd/even pages, and footers for all pages are printed after footers for odd/even pages. The text following the command (i.e. following the ending angular bracket) until the next carriage-return gives the header/footer. To include CR/LFs in a header or a footer, the definition should use the <slX> command (see section 3.1.4). On printing a header or a footer, the formatter defaults to the line fill formatting mode with margins reset.

<ha> Define header for all pages.

<ho> Define header for odd pages.

<he> Define header for even pages.

<fa> Define footer for all pages.

<fo> Define footer for odd pages.

<fe> Define footer for even pages.

The PolyText manual itself gives a good example of these commands in effect. Consider the headers on top of each page. Note that the headers differ from odd to even pages, since.<ho> and <he> are used. The definitions used look like this on the screen:

```
<ho><ex>PolyText_Users_Guide -<pn>-<sl3>
<he><ex>-<pn>- PolyText_Users_Guide<sl3>
```

Note the use of the line expand mode and non-expandable blanks to provide line numbers at one margin and text at the other margin. Below is shown a footer definition which will cause line numbers to be printed at the bottom center of each page:

```
<fa><sll,ce>-<pn>-
```

3.1.6 Text register commands

The text register commands provide a method of varying selected parts of your manuscript, for instance a letterhead, without altering the manuscript itself. The <rdT> command will read from the keyboard (or from a text file) a string of up to 48 characters into one of the text registers. The <wrT> command will, when formatted, expand into the text contained in the designated text register. There are ten text registers, numbered from 0 to 9.

<rdT> This command will read a string into text register T, where T is a text register number (0-9). The text following the ending angular bracket until the next carriage-return is printed on the screen as a prompt string. The maximum length of your input is 48 characters. Note that the input string cannot contain formatter commands.

<wrT> This command will, when formatted, expand into the contents of text register T, where T is a text register number (0-9).

When you select the format text function from the main menu, the formatter will, among other things, prompt you for an input file name. Typing <ENTER> directly in response to this question causes the formatter to obtain text input, i.e. input to text registers requested from the <rdT> command, from the keyboard. However, entering a file specifier (TX is the default extension, and the master drive is the default drive) causes that file to function as the input source. Each time a <rdT> command appears, the next line of the input file will be read into the register. When formatting ends, the formatter asks if it is to do a repeat. This question, which should be answerred by either 'Y' or 'N' followed by a carriage-return, will be fetched from the input file as well. Thus you can create an input file which, apart from supplying the proper text register values, supplies a 'Y' after each record, except after the last one, in which case an 'N' should appear.

3.1.7 Various commands

<pn> Insert page number. When formatted this command expands into the current page number. Usually <pn> is used in connection with headers and/or footers to provide page numbering. The initial value of the page number register can be changed using the output parameters function from the main menu.

<chN> Control character output. This command is used to output control characters to the printer directly without affecting line/character counters maintained by PolyText. N is a constant between 1 and 255 giving the number of characters that follow. The characters must immediately follow the ending angular bracket.

<co> Display comment. The text following the ending angular bracket until the next carriage-return is displayed on the screen as a comment to the operator.