PolyPascal-86 Supplementary Documentation
Installation Notes

PolyPascal is designed in such a way that it can be installed to match the hardware of any microcomputer system. If you have bought an "uninstalled" copy of PolyPascal, i.e. a copy which is not set up for any specific computer system, then before you can use it, you must install it.

To install PolyPascal you may use the INSTALL program supplied on the distribution disk. The INSTALL program is fully self-explaining, but to answer the questions asked by it you will probably have to consult your system documentation. If you wish, you may also use the INSTALL program to customize an already installed copy to suit your individual needs.

Through the INSTALL program you may view and/or modify almost all configuration parameters. However, some features, for instance user written machine code drivers, cannot be accessed, and usually need not be accessed. Should you wish to modify and/or install user written machine code drivers, you must use the DDT86 debugger supplied with your CP/M-86 operating system.

The descriptions that follow specify the address and the length of each field in the configuration table. The address is in hex and the length is in decimal. Note that when PolyPascal is loaded into DDT86 (using the R command), the first $80 bytes of the file make up a command file header. You must therefore add 8 to the load segment address to find the segment address of the configuration table.

Many fields in the configuration table are strings. The first byte of a string contains its length and the following bytes contain the actual characters. When the length of a string field is 9, the string can at most contain 8 characters.

Basic terminal data

0020 32 Terminal name. A string of up to 31 characters defining the name of the installed terminal. A length of zero indicates an unconfigured copy.

0040 1 Screen width. The value contained in this location defines the width of your video display, i.e. the number of characters per line. If a character cannot

be written to the last position on the bottom line without scrolling the screen, then set this field to the actual screen width less one.

2

0041 1 Screen height. The value contained in this location defines the heigth of your video display, i.e. the number of lines on the screen. The value must be exact.

Cursor positioning data

If a user defined GOTOXY routine is not installed, these fields must be filled in for PolyPascal to operate correctly.

0042 9 GOTOXY lead-in sequence. The string sent before the coordinates in a cursor addressing sequence.

004B 5 GOTOXY separator sequence. The string sent between the coordinates in a cursor addressing sequence. This string is usually empty, but some standards (e.g the ANSI standard) requires a separator.

0050 5 GOTOXY terminator sequence. The string sent after the coordinates in a cursor addressing sequence. This string is usually empty.

0055 1 GOTOXY coordinate order. Zero in this location indicates that the Y coordinate (row) is sent before the X coordinate (column). Non-zero means X before Y.

0056 1 GOTOXY coordinate offset. This location defines the offset value to be added to the coordinates before they are sent. Unbiased coordinate values start at 0. A very common offset value is 32 (decimal).

0057 1 Coordinate format. Zero in this location indicates that coordinates are to be sent as single characters in binary format (i.e. the ASCII values of the characters sent correspond to the coordinate values plus an offset as defined above). Non-zero values (2 or 3) indicate that coordinates are to be sent as strings of numeric characters ('0'-'9'), and the value defines the length of such strings.

Terminal function sequences

The CLRHOM sequence must be installed for PolyPascal to operate correctly. All other sequences are optional. All fields in this section are strings.

0058 9 CLRHOM sequence. Clears the screen and places the cursor in the upper left corner.

0061 9 CLREOS sequence. Clears all character locations from the cursor to the end of the screen.

006A 9 CLREOL sequence. Clears all character locations from the cursor to the end of the current line.

3
0073 9 INSLIN sequence. Inserts a blank line at the current line and scrolls all lines below it down.

007C 9 DELLIN sequence. Deletes the current line and scrolls all lines below it up. A blank line must appear at the bottom of the screen.

0085 9 Reverse on sequence. Subsequent characters will be printed in reverse.

008E 9 Reverse off sequence. Turns off the reverse attribute.

0097 9 Intensify on sequence. Subsequent characters will be printed in increased (or decreased) intensity.

00A0 9 Intensify off sequence. Turns off the intensify attribute.

00A9 9 Underline on sequence. Subsequent characters will be underlined.

00B2 9 Underline off sequence. Turns off the underline attribute.

00BB 9 Blink on sequence. Subsequent characters will blink.

00C4 9 Blink off sequence. Turns off the blink attribute.

00CD 9 All attributes off sequence. Turns off all character attributes. Some standards (e.g. the ANSI standard) does not allow for selective deactivation of character attributes. Instead they provide a single sequence to turn off all attributes. If this applies to your

terminal, install empty strings in the four deactivation sequences above, and install the clear attributes sequence in this field.

Editor Attribute Definitions

The fields that follow are used by the PolyPascal editor to determine the character attributes used to display different kinds of texts on the screen. The attribute definitions depend on whether a direct write string routine is installed or not (address 00F3). If a direct write string routine is not installed, the four least significant bits of each field determine which attributes are to be active for that specific kind of text:

Bit 0 = Reverse.
Bit 1 = Intensify.
Bit 2 = Underline.
Bit 3 = Blink.

A bit value of 1 indicates active, and 0 indicates inactive. If a direct write string routine is installed, the values contained in the attribute definition fields are passed directly to the driver with no interpretation. In that case, the values installed are typically the attribute values to be stored in the video RAM to achieve the desired effects.

00D6 1 Text attributes. The attributes used to display ordinary text.

00D7 1 Status line attributes. The attributes used to display the status line.

00D8 1 Prompt line attributes. The attributes used to display prompt lines.

00D9 1 Error message attributes. The attributes used to display error messages.

00DA 1 Overflow marker attributes. The attributes used to display a '+' at the end of lines that are wider than the screen.

00DB 1 Block attributes. The value stored in this field defines the bit pattern to XOR the current attribute value with, when displaying text within a block. In other words, this field defines the attributes to

switch to the opposite state when displaying text within a block.

00DC	1	Control character attributes. The value stored in this field defines the bit pattern to XOR the current attribute value with, when displaying control characters.

GOTOXY routine

If the GOTOXY function cannot be implemented using the GOTOXY configuration table entries, you must write your own driver in machine code. Install the code in the patch area and store the entry address in the vector shown below. When a non-zero address is stored in the vector, it automatically overrides all GOTOXY strings. Registers BP, CS, DS, and SS must be preserved, and the routine must end with a RET (within segment) instruction. (X,Y) is passed in (DL,DH).

00DD	2	GOTOXY driver address. Zero indicates that no driver is installed.

Low-level I/O routines

PolyPascal allows you to write your own low-level machine code drivers for character device input and output. The code should be installed in the patch area and the entry addresses in the vectors shown below. If a driver is not installed, PolyPascal will use a CP/M-86 BIOS call instead. An uninstalled vector should be set to 0. Registers BP, CS, DS and SS must be preserved, and the routines must end with a RET (within segment) instruction. Output routines receive the character in DL, input routines must return a character in AL, and status routines must return AL=$FF if character is ready or AL=$00 if not.

5

00DF	2	Console status driver address. Used by the KEYPRESS standard function. Zero causes PolyPascal to use the CONST routine in the BIOS.
00E1	2	Console input driver address. Used by the CON:, TRM: and KBD: devices. Zero causes PolyPascal to use the CONIN routine in the BIOS.
00E3	2	Console output driver address. Used by the CON:, TRM:

and KBD: devices. Zero causes PolyPascal to use the CONOUT routine in the BIOS.

00E5 2 List output driver address. Used by the LST: device. Zero causes PolyPascal to use the LIST routine in the BIOS.

00E7 2 Auxiliary output driver address. Used by the AUX: device. Zero causes PolyPascal to use the PUNCH routine in the BIOS.

00E9 2 Auxiliary input driver address. Used by the AUX: device. Zero causes PolyPascal to use the READER routine in the BIOS.

Entry and exit routines

The entry and exit routines may be used to initialize and de-initialize the terminal. The code should be installed in the patch area and the entry addresses in the vectors shown below. If a routine is not installed, its vector should be zero. Registers CS, DS and SS must be preserved, and the routines must end with a RET (within segment) instruction.

00EB 2 PolyPascal entry routine address. If installed, this routine is called when PolyPascal is started from the operating system.

00ED 2 PolyPascal exit routine address. If installed, this routine is called just before PolyPascal returns to the operating system.

00EF 2 Editor entry routine address. If installed, this routine is called when the editor is started.

00F1 2 Editor exit routine address. If installed, this routine is called just before the editor returns to the PolyPascal command mode.

Memory mapped video screens

If your computer system is equipped with a memory mapped video screen or another type of "direct drive" screen, you can speed up the PolyPascal editor by implementing the routines outlined be-

low. The code should be installed in the patch area and the entry addresses in the corresponding vectors. If a routine is not installed, its vector should be zero. Registers CS, DS and SS must be preserved, and the routines must end with a RET (within segment) instruction.

00F3 2 High-speed write string routine. If this routine is installed the editor will use it for updating the screen instead of passing characters one by one to the console output driver. On entry CX contains the length of the string to write, and DX contains the screen coordinates. (DL,DH) corresponds to (X,Y) with the upper left corner at (0,0). DS:SI points to a string of words, each of which define a character and an attribute. The least significant byte of each word contains the ASCII value of the character, and the most significant byte contains its attribute value. The attribute values are obtained from the table stored at addresses 00D6-00DC.

00F5 2 Insert line routine. This routine should insert a blank line at a specified position on the screen and scroll the remainder of the screen down one line. On entry DH contains the line number with 0 corresponding to the top line. If this routine is not installed, the editor will use the INSLIN sequence (if available) to insert a line.

00F7 2 Delete line routine. This routine should delete a specified line on the screen and scroll the remainder of the screen up one line. A blank line must appear at the bottom of the screen. On entry DH contains the line number with 0 corresponding to the top line. If this routine is not installed, the editor will use the DELLIN sequence (if available) to delete a line.

User patch area

The user patch area is not used by PolyPascal but left open for you to install user written drivers in it.

00F9 640 User patch area. The address of the last available byte is $0378.

Editor configuration table

The address of the editor configuration table may be different from one release of PolyPascal to another. To find the start address, examine locations $0001 and $0002. They contain the base address (least significant byte first) of the editor configuration table. In the descriptions below, the address of each field is an offset address from the base address of the table.

7

0006 2 Replace prompt delay. This entry is used only by the find/replace function in the editor. It defines the delay used when moving the cursor alternately between the replace prompt and the text. Experiment to find a suitable value (typical values range from 200 to 2000).

0008 256 Alternate editor keyboard table. See below for a description.

0108 1 Side scroll step value. This location defines the step size used by the editor when it scrolls sidewards. The minimum value is 1 and the maximum value is the screen width less one.

0109 1 Backup flag. This location must contain either $00 or $FF. If it is set to $FF, duplicate files will have their type changed to 'BAK' when the SAVE command is used to save a file. Otherwise such files are simply deleted.

010A 1 INSERT mode initial value. This location defines the initial state of the INSERT mode. $FF means on and $00 means off.

010B 1 AUTO mode initial value. This location defines the initial state of the auto-indent tabulator. $FF means on and $00 means off.

010C 1 TABS mode initial value. This location defines the initial state of the TABS mode. $FF means on and $00 means off.

010D 1 Error message mode. The error message mode may be either $00, $01 or $02. $00 means that the error message file should never be loaded. $01 means that the error message file should only be loaded if the user confirms it (by answering yes to the prompt when PolyPascal is started), and $02 means that the error

message file should always be loaded.

010E	12	Help file name. Contains the name of the PolyPascal help file in FCB format. The first byte defines the disk drive. 0 means the default drive, 1 means A, 2 means B, etc. The next 8 bytes define the file name, and the last 3 bytes define the file type. Unused positions in the name and type fields should be set to blanks. If bit 7 is set in the drive number, PolyPascal will scan user number 0 in addition to the current user number when looking for the help file.
011A	12	Error message file name. Contains the name of the PolyPascal error message file name in FCB format as described above.
0126	12	Default workfile name. Contains the default workfile name in FCB format.
0132	3	Default text file type. Three characters which define the default type used by the LOAD, SAVE and NAME commands. Unused positions should be set to blanks.
8		
0135	3	Default backup file type. Three characters which define the type used by the SAVE command to create backup files.
0138	3	Default backup file type. Three characters which define the default type used by the PROGRAM command.
013B	3	Default object file type. Three characters which define the default type used by the OBJECT command.

Alternate editor keyboard table

To be able to benefit fully from the special keys offered by your terminal's keyboard, PolyPascal allows you to define alternate keys to invoke selected editor functions. A typical example of alternate keys is defining your cursor arrows to do the same as ↑S, ↑D, ↑E and ↑X.

The alternate editor keyboard table consists of a list of strings each defining an alternate character (or sequence of characters) to invoke a specific editor function. The first byte of each string defines the number of the function it invokes. The second byte defines the length of the sequence and the following bytes

contain the actual characters of the sequence. The strings are totally contiguous, i.e. immediately following the last character of a sequence comes the next definition. The table is ended by a zero byte. The maximum length of the table is 256 bytes. The function numbers, in hex, are:

01 ↑J	02 ↑S	03 ↑D	04 ↑A
05 ↑F	06 ↑Q↑S	07 ↑Q↑D	08 ↑E
09 ↑X	0A ↑Q↑E	0B ↑Q↑X	0C ↑R
0D ↑C	0E ↑Q↑R	0F ↑Q↑C	10 ↑H
11 ↑G	12 ↑Q↑H	13 ↑Q↑Y	14 ↑T
15 ↑Y	16 ↑M	17 ↑I	18 ↑N
19 ↑P	1A ↑V	1B ↑Z	1C ↑B
1D ↑W	1E ↑Q↑F	1F ↑Q↑A	20 ↑L
21 ↑K↑B	22 ↑K↑K	23 ↑K↑Y	24 ↑K↑C
25 ↑K↑V	26 ↑K↑P	27 ↑K↑R	28 ↑K↑W
29 ↑K↑H	2A ↑K↑D	2B ↑K↑X	

Below follows an example of an alternate editor key table (the numbers are in hex):

```
08 02 1B 41 09 02 1B 42 00
```

This table defines ESC-A ($1B,$41) as an alternate for ↑E (function $08) and ESC-B ($1B,$42) as an alternate for ↑X (function $09). Note that the first character of a function sequence may be any character, but the following characters must be stored as values between $20 and $5F or $80 and $FF. When the editor inputs a function sequence from the keyboard, all control characters and lower case letters following the first character of the sequence are converted to upper case letters, i.e. characters between $00 and $1F or $60 and $7F are converted to characters between $40 and $5F. For instance, this means that ↑Q↑H may also be entered as ↑Q "H" or ↑Q "h".

Note that it is actually possible to override the standard key sequences. If you for instance define ↑J to be an alternate key for the ↑X function, the original ↑J function will become inaccessible unless you define an alternate sequence for it as well.