Text Minikernel User's Guide

The text minikernel is a minikernel that allows bB games to display multiple short (12 character) text messages in games, loaded from a data table.

How to Use

Currently, the text minikernel may be used with any of the provided bB kernels. Most of this document is written with the standard kernel in mind. Differences when using this minikernel with the DPC+ and multisprite kernels will be covered in separate sections at the end of this document.

A project using the text minikernel will need to include the assembly files. For a multibank project, they need to go in the last bank:

```
    inline text12a.asm
    inline text12b.asm
```

The user will need to dim a variable for the TextIndex variable. As one might guess, this controls the color of the text:

```
    dim TextIndex = z
```

text_strings: The actual text data is defined with a data statement in a table named text_strings. In the table, letters and numbers are represented by two underscores followed by the letter or number (capitals only). Symbols such as punctuation are represented by one underscore followed by a two-letter designation, e.g. "_cm" for comma. The available symbols are:

Symbol	**Meaning**
_sp	space
_pd	period

_qu	question mark
_ex	exclaimation point
_cm	comma
_hy	hyphen
_pl	plus
_ap	apostrophe
_lp	left parenthesis
_rp	right parenthesis
_co	colon
_sl	slash
_eq	equal
_qt	quote
_po, _ht	pound sign

A single-line table that holds the line "HELLO, WORLD" would look like this. Note that all 12 characters on a line need to be defined. Extra space can be filled in with the space character ("_sp"):

```
    data text_strings
    __H, __E, __L, __L, __O, _cm, _sp, __W, __O, __R, __L, __D
end
```

TextIndex: The minikernel defines this variable for you. This controls what part of the text data to display. A value of 0 will display the first line of text, a value of 12 will display the second line, etc. A maximum 21 lines may be displayed unless the "extendedtxt" option is used. It can be helpful to assign constants for TextIndex values to make for more readable code:

```
    const hello_message = 0
    const goodbye_message = 12
    const yousuck_message = 24
```

```
    TextIndex = goodbye_message
```

COLUBK: The text area itself always has a background color of black. If a different background color is used in your game, then you will need to make sure that COLUBK is set in your game loop.

scorebkcolor: The score will normally have whatever background color you set for your game, but if you wish to have it match the text area instead, or set it to a completely different color, set this constant to the desired color:

```
    const scorebkcolor = $50
```

noscoretxt: Set this constant to 1 to turn off the score when using this minikernel instead of using the standard "noscore" option:

```
    const noscoretxt = 1
```

extendedtxt: Set this constant to 1 to turn on "extended text" mode. This allows for more than 21 messages, but it adds complexity and makes the minikernel use one more line due to extra processing, so it is not enabled by default. This mode is discussed in a later section.

```
    const extendedtxt = 1
```

textbank: Set this constant to place most of the minikernel code and its supporting data in a different bank. This is useful to avoid taking up limited space in the kernel/graphics bank. Place "text12b.asm" and the text_strings table in the bank specified by this constant, leaving only "text12a.asm" in the last bank:

```
    const textbank = 7
```

Concerns About Cycles

Using any minikernel takes cycles away from your game, and since this minikernel uses more lines than most, it is an even bigger concern. Here are a few possible remedies.

Shrink the Game Area: Decreasing the height of the game area slightly with some combination of pfres and/or pfrowheight can compensate for the lines added (and cycles lost) by this minikernel.

Turn off the Score: Turning off the score if it is not needed will save a number of cycles. To turn off the score when you are using this minikernel, use "noscoretxt" instead of the usual "noscore":

```
    const noscoretxt = 1
```

Shrink the Score: If the provided score_graphics.asm is used with the project, and the score is set to "SQUISH", this will save 3 lines:

```
    const fontstyle = SQUISH
```

Extended Text Mode

This option allows for more than 21 text messages at the cost of pushing the ext down one additional line due to extra processing, and adding complexity to the minikernel. If you use this option, you will also almost certainly want to use the "textbank" option to relocate most of the code and all of the data to a different bank.

TextIndex: In this mode, the TextIndex stands for the message number instead of the character position, so 0 would be the first message, 1 would be the second message, etc. In theory, this could go up to 255, but even with a dedicated bank for the minikernel and data, the bank would fill up after around 240 messages.

text_strings: The format of this table is the same in this mode, but you can go beyond 21 messages. There is one additional complication, however: due to data alignment issues, every 22nd line needs to be filled with filler data, and not used in the code. This is true every 2 out of 3 times, so lines 21 and 42 need to be skipped, but not 63, then 84 and 105, but not 126, etc.

The table template below can be used to help with the skipped lines:

```
    data text_strings
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 0
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 1
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 2
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 3
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 4
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 5
```

```
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 6
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 7
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 8
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 9
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 10
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 11
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 12
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 13
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 14
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 15
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 16
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 17
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 18
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 19
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 20
; *** 21 is skipped ***
      0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0 ; 21
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 22
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 23
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 24
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 25
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 26
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 27
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 28
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 29
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 30
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 31
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 32
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 33
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 34
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 35
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 36
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 37
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 38
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 39
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 40
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 41
; *** 42 is skipped ***
      0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0 ; 42
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 43
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 44
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 45
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 46
```

```
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 47
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 48
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 49
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 50
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 51
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 52
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 53
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 54
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 55
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 56
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 57
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 58
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 59
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 60
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 61
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 62
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 63
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 64
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 65
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 66
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 67
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 68
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 69
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 70
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 71
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 72
; *** 73 is NOT skipped ***
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 73
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 74
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 75
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 76
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 77
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 78
    _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp, _sp ; 79
;
; Next skipped lines are 84, 105, (not 126), 147, 168, (not 189)
;
```

DPC+ Kernel Differences

There are a number of special considerations when using this minikernel, as the built-in support for minikernels in the DPC+ kernel is currently incomplete. Also, the memory model is different, so the minikernel uses 4 more variables with the DPC+ kernel.

Extra Variables: This minikernel uses 4 extra variables as temp variables with the DPC+ kernel. By default, it uses var5 - var8. Alternately, It can be set to use any persistent variables, and they do not need to be consecutive. The minikernel sets these variables back to zero when it is done with them, so these can be set to use any of NUSIZ2 - NUSIZ9 for any sprites that don't need to change their NUSIZ value from 0, or if you set their values manually within your game loop. Here's an example of these being set to NUSIZ values:

```
    const textoffset8  = NUSIZ6
    const textoffset9  = NUSIZ7
    const textoffset10 = NUSIZ8
    const textoffset11 = NUSIZ9
```

Screen Size: Due to the design of the DPC+ kernel, the size of the screen can not be altered by the user via bB code. If you use this minikernel, the screen will be reduced by 16 lines to make room for the text minikernel.

Cycles: Due to the built-in screen size reduction, using the text minikernel with a DPC+ kernel will not take away any cycles from your project.

text12DPCplus.asm: This is the minikernel file to use with the DPC+ kernel. It may be inclided in any bank (2-6) where it will fit:

```
    inline text12DPCplus.asm
```

DPCplus_kernel.asm and **DPCplusbB.h**: These are customized versions of standard bB files needed for the text minikernel. Place these in the same directory as your project, and bB will automatically use these files instead of the standard ones when compiling.

textbank: This is the bank where you have included the text minikernel. If this is nort set correctly, the minikernel will not function properly.

score: This minikernel does not affect the score in any way when using the DPC+ kernel, so the **scorebkcolor** and **noscoretxt** options are also not used. Also note that due to the design of the DPC+ kernel, turning off the score does not reduce the size of the screen at all.

Multisprite Kernel Differences

The text minikernel uses the same files as it does with the standard kernel, and its usage is the same. The differences are explained below:

multisprite_kernel.asm: As of this writing, there is a bug in the multisprite kernel that affects the text minikernel, so a patched version of the kernel is included with the distribution. Put this file in the same directory as your project, and it will use the patched version automatically when you compile.

Cycles: You can save cycles as described in the "Concerns About Cycles" section by shrinking the score or turning off the score. Shrinking the screen can also be done, but it works differently with the multisprite kernel. The screen height is normally 88, but it can be set via the **screenheight** constant to 84 or 80. 80 may be used if the **pfheight** variable is set to less than 8, and 84 may be used if the **pfheight** variable is set to less than 4. Finally, if these options are used, then a playfield needs to be defined, even if it is not used in the game (in which case it will be blank). This is seen in the included 9objtxt demo:

```
   const screenheight = 80
   pfheight = 7

   playfield:
   ................
   ................
   ................
   ................
   ................
   ................
   ................
   ................
   ................
   ................
```

```
    . . . . . .
end
```

Troubleshooting

One user reports that running out of space in the bank where this minikernel resides may result in a large number of label mismatch errors.

Acknowledgements

Many thanks to Mike Saarna ("RevEng") for help with the initial design and development advise for this minikernel. Thanks also to Duane Alan Hahn ("Random Terrain") for proving the demo programs that I hacked to use as demos for this minikernel. Finally, I would like to thank those who have tested this code and provided valuable feedback.