sparkfun ELECTRONICS
WEBSITE: sparkfun.com
6175 LONGBOW DRIVE, SUITE 200 BOULDER, COLORADO USA ZIP CODE: 80301
[303] 284.0979 [GENERAL] P
443.0048 F

Graphic LCD Serial Backpack Datasheet
2009.01.26

Overview

The Serial Graphic LCD backpack interfaces to 160x128 pixel “Huge” Graphic LCD (sku: LCD-08799) and provides the user a simple serial interface to a full range of controls. Besides writing text, the backpack allows the user to draw lines, circles and boxes, set or reset individual pixels, erase specific blocks of the display and control the backlight. There's also a reverse mode that allows the screen to operate blue on white instead of white on blue.

All source code for the ATMega168 processor is compiled using the free WinAVR compiler and is free for downloading on the product description page.

- Voltage: 6V – 7V DC
- Current: 220mA (backlight at 100%)

  Input: 0-5V, 115,200bps (adjustable), 8 data bits, 1 stop bit, no parity
- Dimensions: 4x5x0.65" (102x129x17mm)

Figure 1

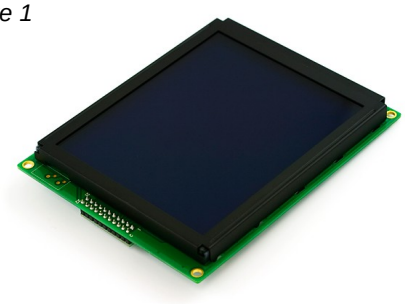
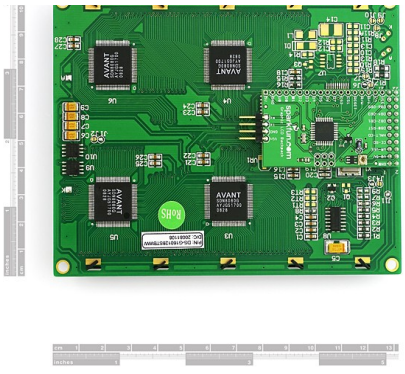


Figure 2



Setup

Setting up the display is as easy as connecting 6V to Vin, 0V to GND and a serial TX line from your source to the RX line on the backpack. The TX line from the backpack has been left in the final design for future code revisions, debugging and user development, but it is not currently utilized as of this writing.

Voltages of up to 9V may be used to power the backpack, however care should be take to reduce the backlight duty cycle in such cases to reduce the chance of overloading the voltage regulator on the backpack.

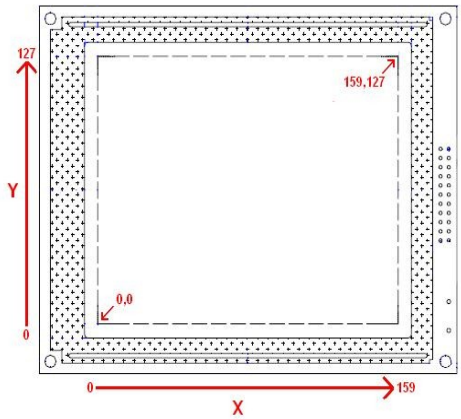
The backpack is primarily intended for embedded applications, but it can easily be connected to a computer and written to with a terminal emulator (the FT232RL USB to Serial Breakout, SKU:BOB-00718, is very good for this – just connect grounds and TX on the break out board to RX on the backpack).

If you're running on a terminal emulator, just open the port to 115,200bps, 8 data bits, 1 stop bit, no parity, and start typing.

Note: There is a small potentiometer on the backpack that allows for contrast adjustments. This should already be adjusted for you, but if text is not readily apparent or otherwise does not suit your needs, feel free to adjust to your liking.

Operation

The graphic LCD is mapped out in Cartesian coordinates as shown in the following picture:



ASCII characters are printed to the screen with respect to two user-changeable settings, x_offset and y_offset. These two settings define the top-left corner bit of a character space, which is 6x8 bits. By changing x_offset and y_offset the user can place test anywhere on the screen.

Printing characters to the screen happens left to right, top to bottom, without adjusting x_offset and y_offset. Further, changing the offsets will change the entire frame of the text, meaning that writing to the end of one line and onto the next will happen seamlessly as the text has no predefined locations where it can or can't be written (except for locations close to the left and bottom edges of the display). Backspace is also functional and tries to maintain the reference frame as set by the user.

The backpack has an input buffer of 416 bytes, the number of characters that fill the screen. Streaming more than 416 characters at a time will likely cause a buffer overrun, and the time it takes for the firmware to process the entire buffer is highly dependent on what combination of characters and special commands that the user sends. Some experimentation may be necessary to find the best operation.

Special Commands

All commands are preceded with "|", or ASCII decimal 124 (0x7C). This tells the display that a command sequence follows. Before any of the following commands are given, they must be preceded with "|". The actual character "|" is not (and cannot) be printed to the screen.

Clear Screen:

Sending "<control>@ (0x00)" clears the screen of all written pixels. If you're operating in normal mode, all pixels are reset. If you're operating in reverse mode, all pixels are set (white background). An example of a "clear screen" command would be to send 0x7C 0x00, or from a keyboard send "|" and <control>@.

Demo code:

Sending "<control>d (0x04)" runs demonstration code. This is in the firmware just as an example of what the display can do. To see the demonstration, send 0x7C 0x04, or from a keyboard send "|" and <control>d.

Reverse Mode:

Sending "<control>r (0x18)" toggles between white on blue display and blue on white display. Setting the reverse mode causes the screen to immediately clear with the new background. To set the reverse mode, send 0x7C 0x18, or from a keyboard send "|" and <control>r. This setting is saved between power cycles. If the display is turned off while in reverse mode, it will next power up in reverse mode.

Splash Screen:

Sending "<control>s (0x13)" allows or disallows the SparkFun logo to be displayed at power up. The splash screen serves two purposes. One is obviously to put our mark on the product, but the second is to allow a short time at power up where the display can be recovered from errant baud rate changes (see Baud Rate for more info). Disabling the splash screen suppresses the logo, but the delay remains active. To disable the splash screen, send 0x7C, 0x13, or from a keyboard send "|" and <control>s.

Set Backlight Duty Cycle:

Sending "<control>b (0x02)" followed by a number from 0 to 100 whill change the backlight intensity (and therefore current draw). Setting the value to zero turns the back light off, setting it at 100 or above turns it full on, and intermediate values set it somewhere in-between. The number setting in the command sequence is an 8-bit ASCII value. As an example, to set the backlight duty cycle to 50, send 0x7C 0x02 0x32, or from a keyboard send "|", <control>b and "2".

Change Baud Rate:

Sending "<control>g (0x07)" followed by an ASCII character from "1" to "6" changes the baud rate. The default baud rate is 115,200bps, but the backpack can be set to a variety of communication speeds:

"1" = 4800bps
"2" = 9600bps
"3" = 19,200bps
"4" = 38,400bps
"5" = 57,600bps
"6" = 115,200bps

As an example, to set the baud rate to 19,200bps, send 0x7C 0x07 0x33, or from a keyboard send "|", <control>g and "3". The baud rate setting is retained

during power cycling, so if it powers down at 19,200bps, it will next power up with that setting.

In a pinch, the baud rate can be reset to 115,200. During the one second delay at power up, send the display any character at 115,200bps. The number "115200" will be shown in the upper left corner of the display, and the backpack will revert to that setting until otherwise changed.

Set X or Y Coordinates:

Sending "<control>x (0x17)" or "<control>y (0x18)" followed by a number representing a new reference coordinate changes the X or Y coordinates. The X and Y reference coordinates (x_offset and y_offset in the source code) are used by the text generator to place text at specific locations on the screen. As stated earlier, the coordinates refer to the upper left most pixel in the character space. If the offsets are within 6 pixels of the right edge of the screen or 8 pixels of the bottom, the text generator will revert to the next logical line for text so as to print a whole character and not parts. As an example, to set x_offset to 80 (the middle of the horizontal axis) send 0x7C 0x17 0x50, or from a keyboard send "|", <control>x and "P". Attempting to set values greater than the length of each axis result in maximizing the respective offsets.

Set/Reset Pixel:

Sending "<control>p (0x10)" followed by x and y coordinates, and a 0 or 1 to determine setting or resetting of the pixel. Any pixel on the display can independently set or reset with this command. As an example, to set the pixel at (80, 64) send 0x7C 0x10 0x50 0x40 0x01, or from a keyboard send "|", <control>p, "P", "@" and <control>a. Remember that setting a pixel doesn't necessarily mean writing a one to that location, it means to write the opposite of the background. So if you're operating in reverse mode, setting a pixel actually clears the pixel and sets it apart from the white background. Resetting that pixel causes it to be white like the background.

Draw Line:

Sending "<control>l (0x0C)" followed by two sets of (x, y) coordinates defining the line's start and stop, followed by a 0 or 1 determines whether to draw or erase the line. As an example, to draw a line from (0,10) to (50,60) send 0x7C 0x0C 0x00 (x1) 0x0A (y1) 0x32 (x2) 0x3C (y2) 0x01, or from a keyboard send "|", <control>l, <control>@, <control>j, "2", "<" and <control>a. To erase the line (and leave surrounding text and graphics unchanged), submit the same command but changing the last <control>a to <control>@.

Draw Circle:

Sending "<control>c (0x03)" followed by x and y coordinates defining the center of the circle, followed by a number representing the radius of the circle, followed by a 0 or 1 determines whether to draw or erase the circle. As an example, to draw a circle at center (80, 64) with radius 10 send 0x7C 0x03 0x50 0x40 0x0A 0x01, or from a keyboard send "|", <control>c, "P", "@", <control>j and <control>a. To erase the circle (and leave surrounding text and graphics unchanged), submit the same command but changing the last <control>a to <control>@. Circles can be drawn off-grid, but only those pixels that fall within the display boundaries will be written.

Draw Box:

Sending "<control>o (0x0F)" followed by two sets of (x, y) coordinates defining opposite corners of the box, followed by a 0 or 1 determines whether to draw or erase the box. This command is exactly like the draw line command, but instead of drawing a line you get a box that exactly contains the line between the given coordinates. As an example, to draw a rectangular box around the line from (0,10) to (50,60) send 0x7C 0x0F 0x00 (x1) 0x0A (y1) 0x32 (x2) 0x3C (y2) 0x01, or from a keyboard send "|", <control>o, <control>@, <control>j, "2", "<" and <control>a. To erase the box (and leave surrounding text and graphics unchanged), submit the same command but changing the last <control>a to <control>@.

Erase Block:

Sending "<control>e (0x05)" followed by two sets of (x, y) coordinates defines opposite corners of the block to be erased. This is just like the draw box command, except the contents of the box are erased to the background color. As an example, to erase a rectangular block around the line from (0,10) to (50,60) send "0x7C 0x05 0x00 (x1) 0x0A (y1) 0x32 (x2) 0x3C (y2)", or from a keyboard send "|", <control>e, <control>@, <control>j, "2" and "<".