PCD8544 Library

This library is for monochrome graphical LCD based on the Philips PCD8544 controller (datasheet (https://github.com/carlosefr/pcd8544/blob/docs/docs/pcd8544.pdf?raw=true)) or compatibles. These displays quite cheap and are commonly found on older monochrome mobile phones, such as the Nokia 3310 (http://en.wikipedia.org/wiki/Nokia_3310) or 5110 (http://en.wikipedia.org/wiki/Nokia_5110).

This library is a very much simpified version of the Adafruit library.

- Optimized for a minimal memory footprint.
- Only SPI mode supported.
- Uses soft-SPI, does not need the SPI pins.

This is a modified version of the PCD8544 library (https://github.com/carlosefr/pcd8544) written by Carlos Rodrigues (https://github.com/carlosefr). It is ported from C++ to C syntax and is meant to be used with the sduino environment for the STM8.

This library is meant to have a minimal memory footprint. If you need graphics and other features and can spare the resources, check out the library (https://github.com/adafruit/Adafruit-PCD8544-Nokia-5110-LCD-library) from Adafruit (no sduino/STM8 port yet, though).

API

Thanks to some c preprocessor macro magic (../../developer/macro) the API syntax is very similar to the original C++ syntax. Apart from the usual name mangeling for polymorph functions replacing the dots in the method names for underscores and a small modification of the initializer code should be enough. See below for an example.

The instantiation "method" can be called only one time per sketch as this library is a singleton library. It is not possible to use more than one instance per sketch.

Arduino syntax	sduino syntax
`PCD8544 lcd`	no default pin mapping supported
`PCD8544 lcd(sclk,sdin,dc,reset,sce)`	`PCD8544 (lcd,sclk,sdin,dc,reset,sce)`
`lcd.begin()`	`lcd_begin()`
`lcd.begin(width,height)`	`lcd_begin_wh(width,height)`
`lcd.begin(width,height,chiptype)`	`lcd_begin_full(width,height,chiptype)`
`lcd.stop()`	`lcd_stop()`
`lcd.clear()`	`lcd_clear()`
`lcd.clearLine()`	`lcd_clearLine()`
`lcd.setPower(flag)`	`lcd_setPower(flag)`
`lcd.display()`	`lcd_display()`
`lcd.noDisplay()`	`lcd_noDisplay()`
`lcd.setInverse(flag)`	`lcd_setInverse(flag)`
`lcd.setContrast(level)`	`lcd_setContrast(level)`
`lcd.home()`	`lcd_home()`
`lcd.setCursor(col,line)`	`lcd_setCursor(col,line)`
`result = lcd.write(value)`	`result = lcd_write(value)`
`lcd.createChar(chr,glyph[])`	`lcd_createChar(chr,glyph)`
`lcd.drawBitmap(data[],columns,lines)`	`lcd_drawBitmap(data[],columns,lines)`
`lcd.drawColumn(lines, value)`	`lcd.drawColumn(lines, value)`

The default resolution of 84x48 fits the commons Nokia 5110 display.

Example

```
#include <PCD8544.h>

PCD8544(lcd, PC5, PC6, PC4, PC7, PD1); // sclk,sdin,dc,reset,sce

int counter = 0;

void setup() {
  lcd_begin();  // default resolution is 84x48
}


void loop() {
  // Write some text on the first line...
  lcd_setCursor(0, 0);
  lcd_print_s("Hello, World!");

  // Write the counter on the second line...
  lcd_setCursor(0, 1);
  lcd_print_u(counter);

  delay(200);
  counter++;
}
```

Hardware connections

To use this library, you must first connect your LCD to the proper pins on the STM8 board. This library uses bitbanging for the SPI data transfer, so you are not bound to the SPI pins for SCLK and MOSI.

For a Nokia 5510 display connected to a STM8S103 breakout board, the connections look like this:

Display Pin	STM8S breakout board
Pin 1 (VCC)	+3.3V (marked by a square around the pin on the silkscreen)
Pin 2 (GND)	Ground
Pin 3 (SCE)	STM8S103 pin PD1 (sduino digital pin 10)
Pin 4 (RST)	STM8S103 pin PC7 (sduino digital pin 9)
Pin 5 (D/C)	STM8S103 pin PC4 (sduino digital pin 6)
Pin 6 (MOSI)	STM8S103 pin PC6 (sduino digital pin 8)
Pin 7 (SCLK)	STM8S103 pin PC5 (sduino digital pin 7)
Pin 8 (LED)	82 Ohm resistor to 3.3V or 330 Ohm to 5V

Since the STM8S works on 3.3V you can connect the data lines directly to port lines. If you are using this display with a 5V CPU (like on most Arduino boards), you have to add extra components to connect it to the digital pins of the Arduino (not necessary if you are using a 3.3V variant of the Arduino, such as Sparkfun's Arduino Pro).

The background LEDs need only 2.8V/6mA. If you connect them to a port pin, prefer a HS (High sink) pin (see CPU datasheet, all pins except the oscillator and the I2C pin PA1/PA2 and PB4/PB5). PWM pins are the best choice.

when soldering the pin connectors, keep in mind that the upside of the display is the thicker part of the metal frame.

For a Nokia 3310 display the connections would be the following:

Display Pin	Arduino Pin
Pin 1	+3.3V Pin
Pin 2 (SCLK)	PC5, digital Pin 7
Pin 3 (SDIN/MOSI)	PC6, digital Pin 8
Pin 4 (D/C)	PC4, digital Pin 6

Display Pin	**Arduino Pin**
Pin 5 (SCE)	PD1, digital Pin 10
Pin 6	Ground Pin
Pin 7	10uF capacitor to Ground Pin
Pin 8 (RST)	PC7, digital Pin 9

For this display model, "Pin 1" is the leftmost pin when facing the back of the display with the connector on top.

Custom Symbols

The library allows the use of custom bitmap symbols (5x8), defined by an array of five bytes. Checkout the examples in the library folder for usage of this function.

To make it easy to create custom symbols, there's a graphical glyph editor available online (http://carlosefr.github.io/pcd8544/).