

AVC Competitor Registration is **now closed**! If you would like to attend as a spectator, please follow the link for spectator registration found at the bottom of [this page](#).

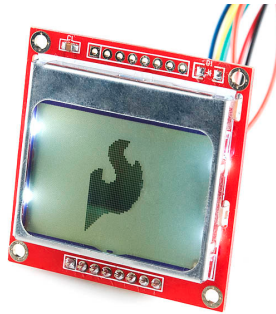


Graphic LCD Quickstart Guide

by [Hily](#) | June 30, 2011 | [16 comments](#) Skill Level: ☆ Beginner

Overview:

Welcome to the landing page for the **Graphic LCD**! These displays are nifty little monochrome graphic LCDs which can be used in your project to display both graphics and text. The display is 84x48 pixels and has LED backlighting for low light visibility. Do they look familiar? They were originally created for old Nokia phones. ([Remember these?](#))



Requirements:

Assembly of this kit will require soldering. If you need a refresher in soldering, please check out this helpful [guide](#). Additionally, you will need a few tools for assembly:

- [soldering iron](#)
- [solder](#)
- [cutters](#)
- [male headers](#) or other connection method
- (optional) [needle nose pliers](#)
- (optional) [third hand](#)

Since the Graphic LCD is simply a serially controlled LCD there are many, many ways you can use or control it. However, for this guide you will find the following helpful:

- [breadboard](#)
- [jumper wires](#)
- [Arduino](#)
- 2 x [Logic Level Converters](#) or 4 x [10k ohm resistors](#) and [1k ohm resistor](#)

What it Does:

The LCD displays black text or graphics. However, what you don't see concealed beneath the metal holder/faceplate is a tiny IC helping make control easier. The controller handles supply and bias voltages and allows easy serial communication with the screen. This is good news, because doing those things without the controller would require lots of external components and work! Of course even with the controller this display has over 4,000 pixels for you to control. Fortunately, there is a great [library](#) that makes printing text or displaying graphics a snap!

How to Use it:

With that general idea of how the LCD works let take a look at hooking it up and using it!

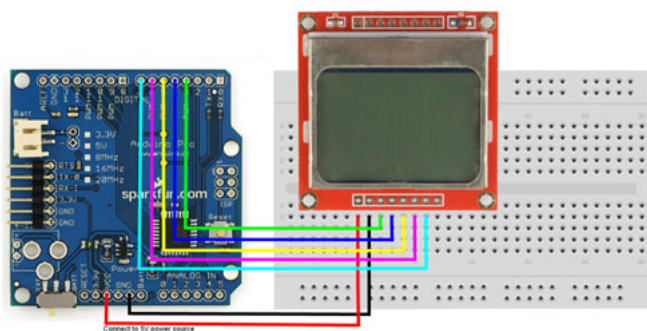
Hardware:

Now you'll want to wire up the LCD. But! Wait for it... This LCD is a 3.3V device*. This means that it should not be directly wired to a 5V microcontroller like the [Arduino](#). It can, however, be directly wired to devices like the [Arduino Pro 3.3V](#) and [Pro Mini 3.3V](#). To learn more about interfacing devices with different logic levels please check out [this tutorial](#).

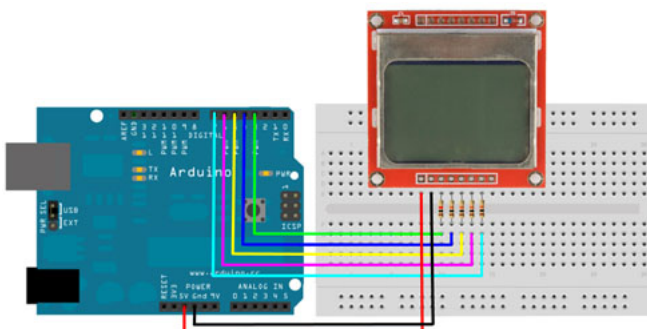
*Technically the absolute maximums of the PCD8544 controller can handle 5V logic levels and we have tested the screen successfully at these voltage. However, the life span of the device may be shortened by this.

The next thing you need to know is the nifty build-in controller (**PCD8544**) on this display allows the LCD to communicate using SPI. Learn the nitty gritty of **SPI** on [Wikipedia](#). Otherwise you should at least know SPI stands for Serial Peripheral Interface Bus and is a common type of serial data protocol used for communication between devices.

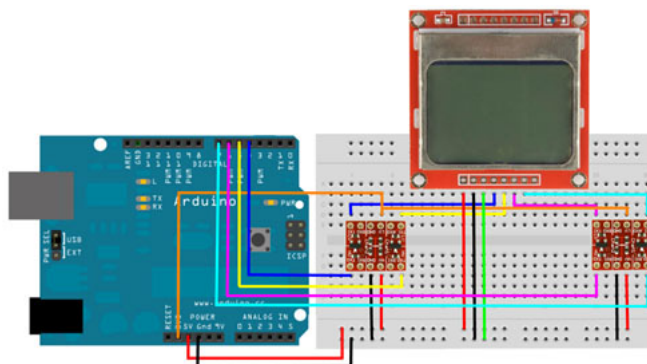
On to wiring! Choose one of the following wiring setups:



Here is how you wire the LCD directly to a **3.3V microcontroller**.



Here is how you wire the LCD using 10k ohm resistors for current limiting. With the exception of the SCE/CS pin wired in green, which uses 1k ohm.



Here is how you wire the LCD using **logic level converters**. The SCE/CS pin shown in green is tied low in this example. To use other devices your SPI bus you'll have to wire this to a logic input as well.

Note: The LEDs are not connected in these examples. The LED pin consists of 4 white LEDs which draw ~80mA total. To drive either wire directly to 3.3V or to control connect to a transistor + microcontroller pin.

Software:

After you have properly wired the device it is time to get some working code! First, download the Arduino **library** created by Adafruit. If you are trying to do something not covered by the library or working with another platform, Google around for the PCD8544. There are literally thousands of results. This particular Nokia LCD has been around for a long, long, long while, since 1998! This means there is A LOT of example code for it out there.

If you don't know how to install a library for Arduino, instructions can be found on the [Arduino website](#).

Basic Functions and Text

One of the most basic things to do with this LCD is to display text on the screen. The library makes this super easy, but there are a few things you will want to note about how it is done. The screen is divided into 6 lines for text. Each line is 8 pixels tall and 84 pixels wide. You can shift the text left and right over all 84 pixels of the screen; however, each shift up and down will jump the text up a line. Play with the example code to get the idea.

```
#include "PCD8544.h"

//Define the pins you are connecting the LCD too.
//PCD8544(SCLK, DIN/MOSI, D/C, SCE/CS, RST);
PCD8544 nokia = PCD8544(3,4,5,7,6);

void setup(void)
{
  nokia.init(); //Initialize lcd
  // set display to normal mode
  nokia.command(PCD8544_DISPLAYCONTROL | PCD8544_DISPLAYNORMAL);
  nokia.clear(); //clears the display
}

void loop(void)
{
  int pause=2000;
}
```

Drawing Basic Shapes

Another nice function of the library is the ability to draw some simple shapes. Shapes include: pixels, lines, rectangles, and circles. They are pretty self explanatory so check out the example code and try it out!

```
#include "PCD8544.h"

//Define the pins you are connecting the LCD too.
//PCD8544(SCLK, DIN/MOSI, D/C, SCE/CS, RST);
PCD8544 nokia = PCD8544(3,4,5,7,6);

void setup(void)
{
  nokia.init(); //Initialize lcd
  // set display to normal mode
  nokia.command(PCD8544_DISPLAYCONTROL | PCD8544_DISPLAYNORMAL);
  nokia.clear(); //Clear the display
}

void loop(void)
{
  // For all functions:
}
```

Displaying Bitmap Images

One of the best features of graphic LCDs is being able to display simple images on the screen. You've no doubt seen some thing of the sort on other mobile devices indicating things like battery life and signal strength. Check out the following example code to learn how to display very simple black and white images on your screen:

```
#include "PCD8544.h"
#include "Graphics.H" // This file contains all of the images

//Define the pins you are connecting the LCD too.
//PCD8544(SCLK, DIN/MOSI, D/C, SCE/CS, RST);
PCD8544 nokia = PCD8544(3,4,5,7,6);

void setup(void)
{
  nokia.init(); //Initialize lcd
  // set display to normal mode
  nokia.command(PCD8544_DISPLAYCONTROL | PCD8544_DISPLAYNORMAL);
  nokia.clear(); //Clear the display
}

void loop(void)
{
}
```

Graphic.h File:

```
#ifndef __graphic_H
#define __graphic_H

static unsigned char __attribute__((progmem)) SFE [] = {
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x7E, 0xFE, 0xFE, 0xF2, 0xE0, 0xC0, 0xC0, 0xE0, 0xC0, 0xF0, 0xC0, 0xC0, 0xF0, 0xC0, 0xF0, 0xC0,
0x00, 0x00, 0x80, 0x80, 0xC0, 0xC0, 0xE0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0,
0x78, 0xF8, 0xF8, 0xFC, 0xFC, 0xFE, 0xFF, 0xFF, 0xFF, 0xF1, 0xF0, 0xC0, 0xF0, 0xF0, 0xF0, 0xF0,
0xFF, 0xFF, 0xFF, 0xFF, 0x3F, 0x00, 0x00, 0x80, 0xC0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0,
0x83, 0x81, 0x81, 0x00, 0x00, 0x00, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80, 0x80,
0x81, 0x83, 0x83, 0x07, 0x0F, 0x1F, 0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0x01, 0x00, 0x02, 0x8F, 0xC0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0,
0x00, 0xFF, 0xFF, 0x3F, 0x3F, 0x3F, 0x01, 0x00, 0x00, 0xFF, 0xC0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0,
0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0xFF, 0xFE, 0xF0, 0xC0, 0x83, 0x83, 0x02, 0x02, 0x03, 0x03, 0x03, 0x03, 0x03, 0x03, 0x03, 0x03,
0x00, 0x00, 0x00, 0x00, 0x03, 0x03, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02, 0x02,
}
```

To make your own graphics follow these simple steps:

1) Get a good graphic. The easiest to work with graphics have no gradients or shading. For this example I will use a [SparkFun logo](#). SparkFun graphics can be found on the [About Us](#) page.

2) Now, since this display is monochrome, we need to make this image black and white. This can be done many ways but here's one example:

- i) First, get your image into [Pixlr's online photoeditor](#).
- ii) Next, use the crop tool to isolate the part of the image you want to make into your graphic. Also, If your image is multi-colored you'll want to do your best to reduce the colors as much as possible at this point.
- iii) Now, in the toolbar click on "Image" > "Image Size..." Adjust dimensions until the image will fit within the 84 by 48 pixel screen
- iv) Again in the tool bar click "Adjustment">"Threshold..." Change the level until you like the image on screen.
- v) Save as a bitmap file (BMP) to your computer and you're done!



3) With this image we simply use handy program called **LCD Assistant** that will take bitmap images and spit out the necessary data array. Settings to use: Vertical byte orientation and Little endianness.

4) LCD Assistant will produce a text file. Copy the array from this text file into Graphics.h and create some constants for your height and width

Drawing Live Graphs

Finally, here is a nifty example for drawing a live bar graph on your LCD:

```
#include "PCD8544.h"

//Define the pins you are connecting the LCD too.
//PCD8544(SCLK, DIN/MOSI, D/C, SCE/CS, RST);
PCD8544 nokia = PCD8544(3,4,5,7,6);

void setup(void)
{
  nokia.init(); //Initialize lcd
  // set display to normal mode
  nokia.command(PCD8544_DISPLAYCONTROL | PCD8544_DISPLAYNORMAL);
  nokia.clear(); //Clear the display
}

void loop(void)
{
  // nokia.drawline(x1,y1,x2,y2,color); draws a line from (x1,y1) to (x2,y2) with color
}
```

Resources:

- [Graphic LCD Product Page](#)
- [Library](#)
- [Arduino Powered 2.4 GHz Spectrum Analyzer](#) (example)


Conclusion:

Enjoy your new LCD! If you have any problems, feel free to contact SparkFun Technical Support at techsupport@sparkfun.com


▼ Comments 16 comments

Enter a comment...

[formatting help](#) [comment guidelines](#)

-  Member #301934 | about a year ago 1
reply | report

ok, this tutorial doesn't work. I only burned my led's because you said that it is save to power it up with 5V. Well it is not save.

-  CypherSoarize | about a year ago 2
reply | report

Works fine for me. And it is safe to hook up to 5V if you use power limiting resistors. Obviously you didn't read through the tutorial correctly.

-  jago lee | about 8 months ago * 2
reply | report

It is not safe to hook it to 5v. Where you using logic level converters? If you have a 3.3 Arduino Pro or mini it works great! The good thing is that this LCD is well priced, buy another try again... Power limiting resistors will work, but logic level conversion has much less ripple. But it should last hours with out burning out at 5v.

-  chopperdave | about a year ago 1
reply | report

The link for the library is broken.


Is it still available elsewhere?

Thanks!


-  jefftoaster | about a year ago * 1
reply | report

<https://github.com/adafruit/Adafruit-PCD8544-Nokia-5110-LCD-library>


don't forget you'll need their GFX library (the link is on that page) to run their example

-  Patrickkd1 | about 9 months ago 1
reply | report

Thankyou so much :)

-  Member #235520 | about 8 months ago 1
reply | report

How would i tell the lcd to start on a new line? would i say lcdstringln?


-  Doctor Who | about 7 months ago 1
reply | report

The tutorial worked fine for me, and I used an Arduino purchased about three years ago. (I used it to facilitate my wiring of it, and naturally pulled 3v3 from the connector on the board for that purpose.)


-  Member #383690 | about 5 months ago 1
reply | report

Has anyone tried this using the logic level converters? It looks from the picture that the D/C, SCLK, MOSI, and RES pins are all are connected through the TX pins on the logic level converters. Since all these pins send data from the microcontroller to the screen, shouldn't they be connected to the RX lines?

If they do need to be connected to the RX lines, then one could simply produce the same exact circuit by connecting another 10K resistor from each signal pin to ground. This would create a voltage divider exactly like the logic level converters do. (of course, the vcc pin should be connected to 3.3 V for power!). I have tried this (voltage dividers and connecting to 3.3 V) and it seems to be working pretty well. I'm new here, is there a place I can upload the circuit I am using to benefit others?

-  Member #426421 | about a month ago 1
reply | report

Hmmm. Just got one of these. Looking at the data sheet the supply voltage clearly states 2.7 – 3.3 Volts. So why are you powering these (in the above examples) with 5V. Trying to sell more :). These examples really ought to be fixed.

-  Member #430247 | about 3 weeks ago * 1
reply | report

Help very welcome for this, in spite of my best newbie efforts I get a set of errors every time I try the first example from this page. I have made sure that the library files are properly set up and will describe that setup after the errors. Error messages are:

In file included from sketch_Nokia_5110_LCD_example1.ino:1:

```
C:\Documents and Settings\Andrew\My Documents\Arduino\libraries \Adafruit_PCD8544\Adafruit_PCD8544.h:52: error: expected class-name b
sketch_Nokia_5110_LCD_example1:5: error: 'PCD8544' does not name a type
sketch_Nokia_5110_LCD_example1.ino: In function 'void setup()':
sketch_Nokia_5110_LCD_example1:9: error: 'nokia' was not declared in this scope
sketch_Nokia_5110_LCD_example1.ino: In function 'void loop()':
sketch_Nokia_5110_LCD_example1:19: error: 'nokia' was not declared in this scope
```


At present my Arduino folders are set up as follows:

```
My Documents
  Arduino
    libraries
      Adafruit_GFX
        Adafruit_GFX.cpp
        Adafruit_GFX.h
      Adafruit_PCD8544
        Adafruit_PCD8544.cpp
        Adafruit_PCD8544.h
    sketch_Nokia_5110_LCD_example1
    sketch...
  arduino-1.0.4
    drivers
    examples
    hardware
    java
    lib
    libraries
    reference
    tools
```

Some other demo sketches have worked so I know that my display and wiring are okay...it is just the compile error that is driving me mad for this one. I want fix it, though, to know what the problem was. None of the other sketches used .include statements so they are bulky code.

- o  **zeal370** | about 2 weeks ago * 1
reply | report

Hi, the example code above used an old library. In the new library, the type names and function names have changed. Your folders are set up correctly. Have a look at the example file “..\\Adafruit_PCD8544\\examples\\pcdtest\\pcdtest.pde”.


-  **Member #430247** | last week 1
reply | report

Hi zeal370, thank you for this information.


Hope the comment stays here and is found by anyone else trying to follow the quickstart example. I did get things working using other examples, most excellently with Henning Karlsen's code found here: <http://www.henningkarlsen.com/electronics/library.php?id=44> which allowed larger number fonts.

-  **thebread** | last week 1
reply | report

How do I use my multimeter to check that there is continuity in the logic converter circuits?

-  **Member #435486** | about 5 days ago 1
reply | report

Hi there, would any of you readers be so kind and tell me how can I see all the functions included in the Adafruit library? If you look at the 16x2 LCD library on the Arduino.cc website you see that they have beautifully publicized all functions the library supports plus documentation on how to implement them. I simply don't understand where to find the same thing for this library, and if I don't know that, I really don't know how am I supposed to use it at its full potential... Thank you very much!

- o  **MikeGrusin** | about 5 days ago 1
reply | report

For any Arduino library, the .h file should have the cleanest list of (hopefully commented) functions that the library provides.