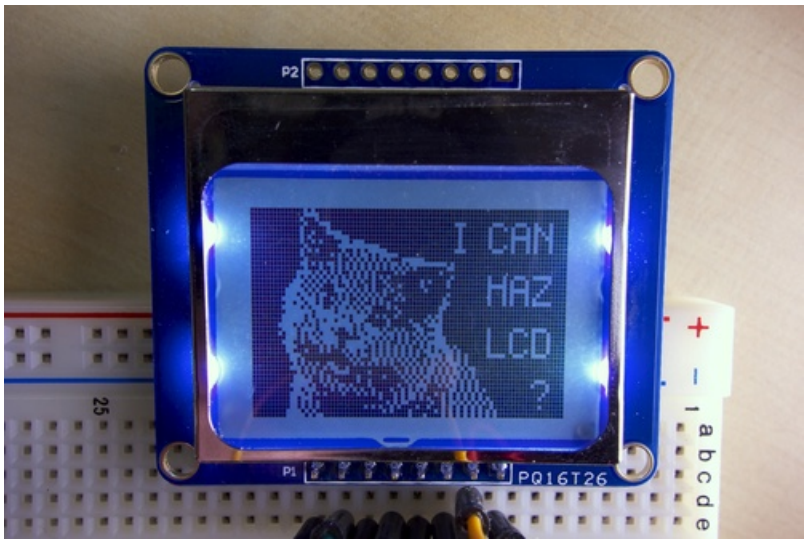
adafruit learning system

Nokia 5110/3310 LCD Python Library

Created by Tony DiCola



Last updated on 2018-08-22 03:41:45 PM UTC

Guide Contents

Guide Contents 2
Overview 3
Wiring 4
- Raspberry Pi 4
- Beaglebone Black 4

Usage 6
- Dependencies 6
- Raspberry Pi SPI Setup 6
- Beaglebone Black Device Tree Setup 6
- Usage 7

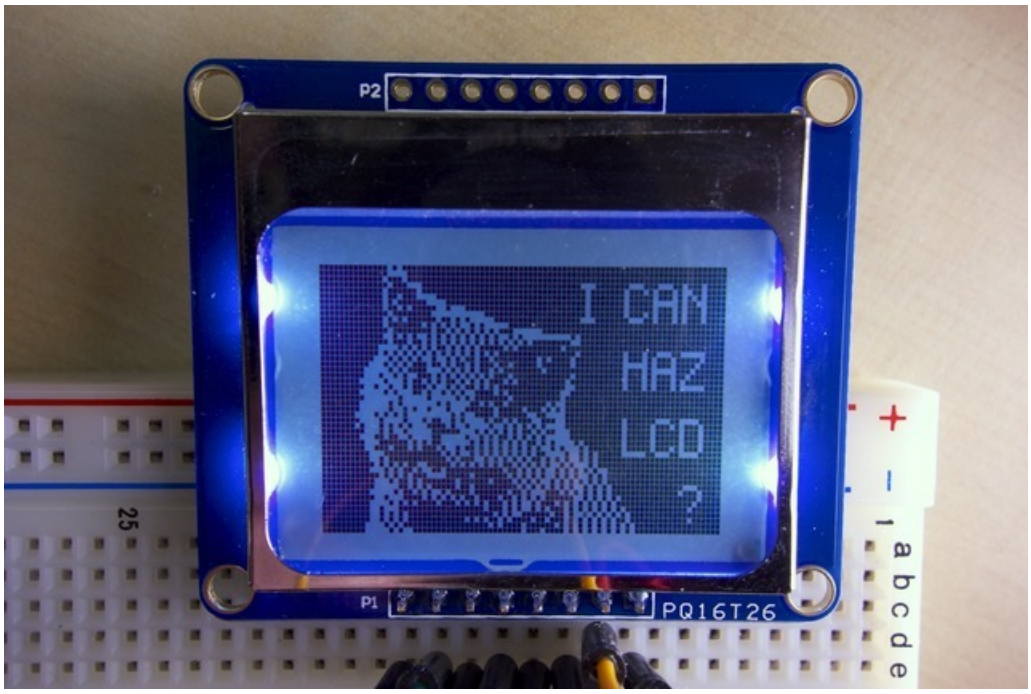
Overview

The Nokia 5110/3310 display (http://adafru.it/338) is a great inexpensive graphical display that's easy to use with an Arduino (https://adafru.it/dve). Wouldn't it be nice to use this display with a modern Linux-based development board like the Raspberry Pi or Beaglebone Black too? Now you can, with the Adafruit Nokia LCD Python library (https://adafru.it/dvf)!

This library allows you to connect the Nokia LCD to a Raspberry Pi or Beaglebone Black and display graphics using the Python programming language. Make the next great snake game, or maybe just display some stats for your Raspberry Pi/Beaglebone Black server--you're only limited by what you can imagine! Follow this guide to learn how to install and use the Nokia LCD python library.

To follow this guide you will need a Nokia 5110/3310 display and a Raspberry Pi (either model A or B) or Beaglebone Black. You will want to be running the latest Raspbian (https://adafru.it/dpb) or Raspbian-derived (https://adafru.it/dvg) operating system on your Pi, or the latest Debian image for the Beaglebone Black (https://adafru.it/dvh).

If you aren't familiar with the Nokia LCD, take a moment to read the Arduino guide (https://adafru.it/dvj) for more information on the display too.



Wiring

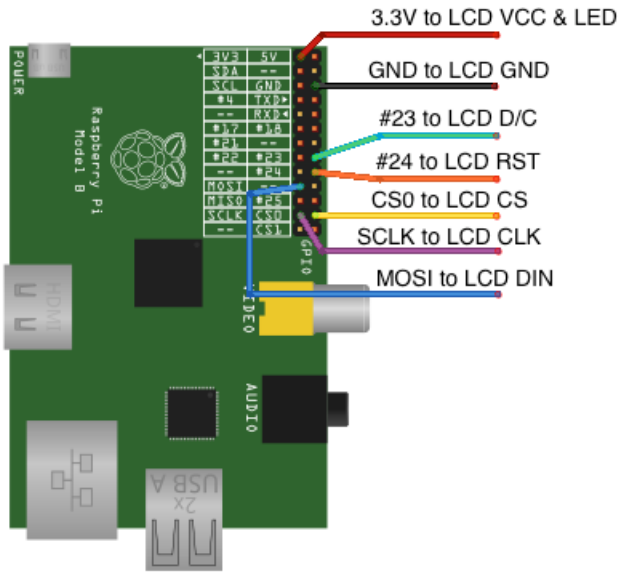
Connecting a Raspberry Pi or Beaglebone Black to the Nokia LCD is very simple **because these boards use the same 3.3 volt I/O as the LCD. There's no need to use a level converter chip like with the Arduino!**

In general you will need to connect the LCD's SCLK, DIN, and CS pins to the board's SPI pins, and the LCD's RST and D/C pins to two free digital I/O pins. With this setup you can use the very fast hardware SPI support built into your board.

However if you need more flexibility and don't require fast screen updates, the library also supports software SPI on any 5 digital I/O pins (SCLK, DIN, CS, RST, D/C). The example code on the next page describes how to enable software SPI. If you aren't sure what to use, start with hardware SPI since it's much more responsive at updating the display.

Raspberry Pi

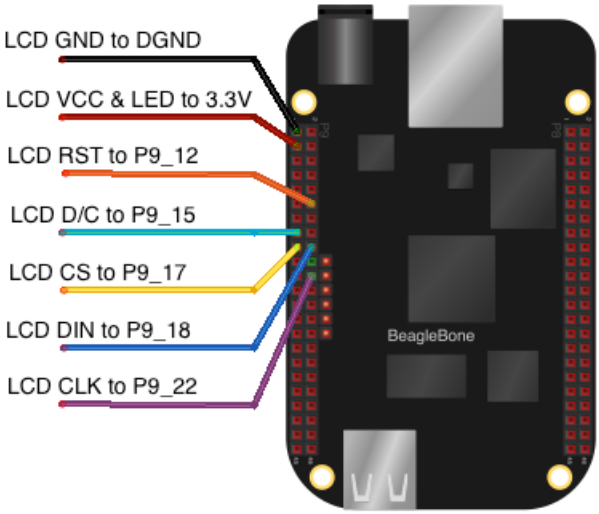
To connect the LCD to a Raspberry Pi, the Pi's hardware SPI pins and two GPIO pins should be wired to the LCD as follows:



The above wiring will support talking to the LCD over the **/dev/spidev0.0** interface.

Beaglebone Black

To connect to the Beaglebone Black, connect the SPI0 pins and two additional digital IO pins as follows:



The configuration above will enable communication with the LCD over the **/dev/spidev1.0** interface.

Also if you are new to the Beaglebone Black, take a moment to review how the GPIO pins are laid out and numbered on the board (https://adafru.it/dvk). There are a lot of pins so be careful to connect the LCD to the right ones!

Once your LCD is connected to your hardware, power on your device and confirm the LCD backlight turns on. If you don't see the backlight turn on double check your ground and voltage wiring carefully.

Usage

Dependencies

Before using the library you will need to make sure you have a few dependencies installed. Connect to your device using SSH and follow the steps below.

If you're using a Raspberry Pi, install the RPi.GPIO library by executing:

```
sudo apt-get install python-pip python-dev build-essential
sudo pip install RPi.GPIO
```

If you're using a Beaglebone Black, install the Adafruit_BBIO library by executing:

```
sudo apt-get install python-pip python-dev build-essential
sudo pip install Adafruit_BBIO
```

Finally, on both the Raspberry Pi and Beaglebone Black install the Python Imaging Library (https://adafru.it/dvB) by executing:

```
sudo apt-get install python-imaging
```

Now to download and install the Nokia LCD python library code and examples, execute the following commands:

```
sudo apt-get install git
git clone https://github.com/adafruit/Adafruit_Nokia_LCD.git (https://adafru.it/dvC)
cd Adafruit_Nokia_LCD
sudo python setup.py install
```

Raspberry Pi SPI Setup

If you haven't done so already with your Pi, make sure to edit the blacklist.conf file (https://adafru.it/dvD) to comment the line which disables SPI. Reboot your Pi and you should see the files **/dev/spidev0.0** and **/dev/spidev0.1** are now available.

Beaglebone Black Device Tree Setup

If you're using the Beaglebone Black you'll need to make sure you enable the SPI ports by manipulating the device tree (https://adafru.it/dp6). The easiest way to setup the device tree is to automatically enable a built-in SPI overlay on boot.

With the Beaglebone Black connected to your computer over USB, open the USB mass storage drive named 'boot' and edit the file uEnv.txt on the drive. Add the following line to the file:

```
optargs=capemgr.enable_partno=BB-SPIDEV0
```

NOTE: Be careful editing the uEnv.txt file on Windows, as changing the line endings can cause your BeagleBone Black not to boot and require an OS reinstall! The safest option is to connect to the BeagleBone Black in a command window andfollow the steps at the end of this page to mount and edit uEnv.txt on the BeagleBone Black (https://adafru.it/dEK).

Reboot your device and you should see the files **/dev/spidev1.0** and **/dev/spidev1.1** now exist.

Usage

Inside the examples subdirectory you'll find python scripts which demonstrate the usage of the library. To help you get started, I'll walk through the **shapes.py** code below:

```
import time

import Adafruit_Nokia_LCD as LCD
import Adafruit_GPIO.SPI as SPI

import Image
import ImageDraw
import ImageFont
```

First the necessary modules are imported to use the library. Specifically the Adafruit_Nokia_LCD and Adafruit_GPIO.SPI import statements load the Nokia LCD library and SPI library. The last three lines import the Python Imaging Library which will be used for drawing graphics and text.

```
# Raspberry Pi hardware SPI config:
DC = 23
RST = 24
SPI_PORT = 0
SPI_DEVICE = 0

# Raspberry Pi software SPI config:
# SCLK = 4
# DIN = 17
# DC = 23
# RST = 24
# CS = 8

# Beaglebone Black hardware SPI config:
# DC = 'P9_15'
# RST = 'P9_12'
# SPI_PORT = 1
# SPI_DEVICE = 0

# Beaglebone Black software SPI config:
# DC = 'P9_15'
# RST = 'P9_12'
# SCLK = 'P8_7'
# DIN = 'P8_9'
# CS = 'P8_11'
```

Next the the pins and ports which are used to connect to the LCD are defined.

The first block defines configuration for using a Raspberry Pi with hardware SPI as shown in the diagram on the previous page. Below this block is a commented out block which sets the pins for using software SPI.

Similarly the last commented sections show the pins and ports for using the Beaglebone Black, both with hardware SPI and software SPI.

Uncomment and comment the pins appropriately to match your hardware configuration.

```
# Hardware SPI usage:
disp = LCD.PCD8544(DC, RST, spi=SPI.SpiDev(SPI_PORT, SPI_DEVICE, max_speed_hz=4000000))

# Software SPI usage (defaults to bit-bang SPI interface):
#disp = LCD.PCD8544(DC, RST, SCLK, DIN, CS)

# Initialize library.
disp.begin(contrast=60)

# Clear display.
disp.clear()
disp.display()
```

Now the code creates an instance of the LCD class that uses hardware SPI by passing the DC and RST pin values, and an instance of the SpiDev class to the LCD class constructor.

Note that the SPI_PORT and SPI_DEVICE values point to a spidev hardware SPI device at /dev/spidev{port}.{device} in the filesystem, like /dev/spidev0.0 in this case. Also notice the speed of SPI communication is set to 4mhz, the maximum speed for the LCD.

The commented line that follows demonstrates how to create an instance of the LCD class using software SPI. In this case the pin values for the LCD's DC, RST, SCLK, DIN, and CS pins must be specified instead of an SPI class instance.

Once the display is created it is initialized by calling begin(), and then cleared and drawn using the clear() and display() functions.

```
# Create blank image for drawing.
# Make sure to create image with mode '1' for 1-bit color.
image = Image.new('1', (LCD.LCDWIDTH, LCD.LCDHEIGHT))

# Get drawing object to draw on image.
draw = ImageDraw.Draw(image)

# Draw a white filled box to clear the image.
draw.rectangle((0,0,LCD.LCDWIDTH,LCD.LCDHEIGHT), outline=255, fill=255)

# Draw some shapes.
draw.ellipse((2,2,22,22), outline=0, fill=255)
draw.rectangle((24,2,44,22), outline=0, fill=255)
draw.polygon([(46,22), (56,2), (66,22)], outline=0, fill=255)
draw.line((68,22,81,2), fill=0)
draw.line((68,2,81,22), fill=0)

# Load default font.
font = ImageFont.load_default()

# Alternatively load a TTF font.
# Some nice fonts to try: http://www.dafont.com/bitmap.php
# font = ImageFont.truetype('Minecraftia.ttf', 8)

# Write some text.
draw.text((8,30), 'Hello World!', font=font)

# Display image.
disp.image(image)
disp.display()
```

The remaining code demonstrates how to draw images and write text using the Python Imaging Library. First an Image (https://adafru.it/dvE) is created with the same dimensions as the LCD (84 by 48 pixels), and a 1 bit (i.e. black and white) image format. An ImageDraw (https://adafru.it/dfH) object is created with the image, and drawing commands such as rectangle, line ellipse, and polygon are made against the draw class.

To display text, a default font is loaded and text commands are made on the draw class. If you have a TrueType font available you can even load it to display a custom font.

Once drawing on an image is complete, it's displayed on the LCD by calling the image() function to copy the image data to the LCD buffer, and the display() function to send the buffer to the display.

That's all the code you need to start drawing graphics on the LCD!

To run the example execute the following command:

sudo python shapes.py

You should see your display draw a few shapes and a line of text like the image above shows. If you don't see anything, carefully check your connections to make sure the right pins are connected to the LCD. Also try adjusting the contrast value in the LCD's begin() call as it can sometimes be too low to be visible.

Examine and run the other example files such as **image.py** to see how to load and display an image file, or**animate.py** to see a simple text animation.

Enjoy using your Nokia LCD with a Raspberry Pi or Beaglebone Black! If you run into problems with the library or wish to contribute to it, feel free to raise issues and send pull requests to the library's home on github (https://adafru.it/dvf).